CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

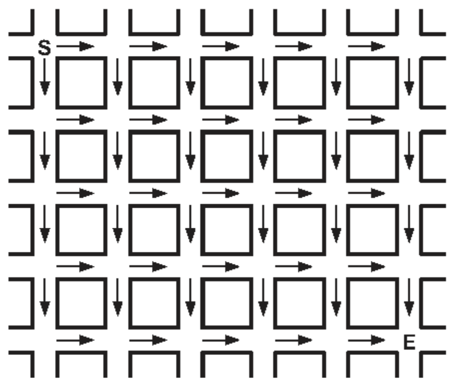

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

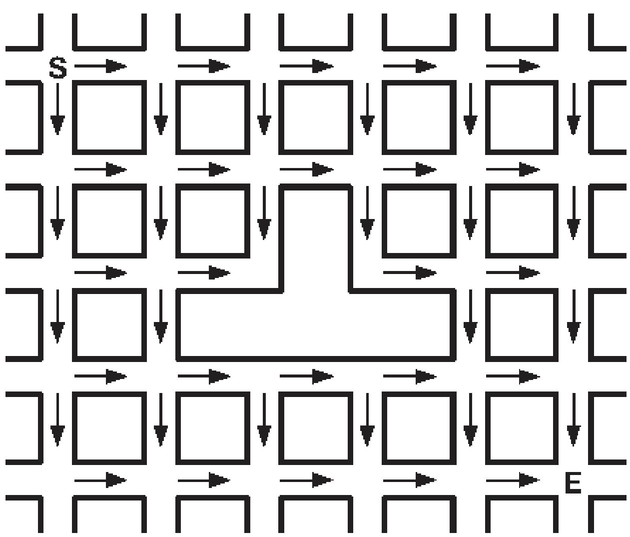

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

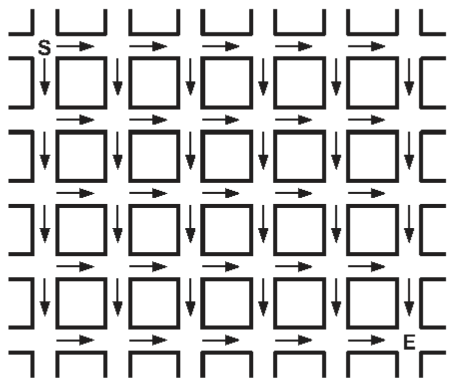

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

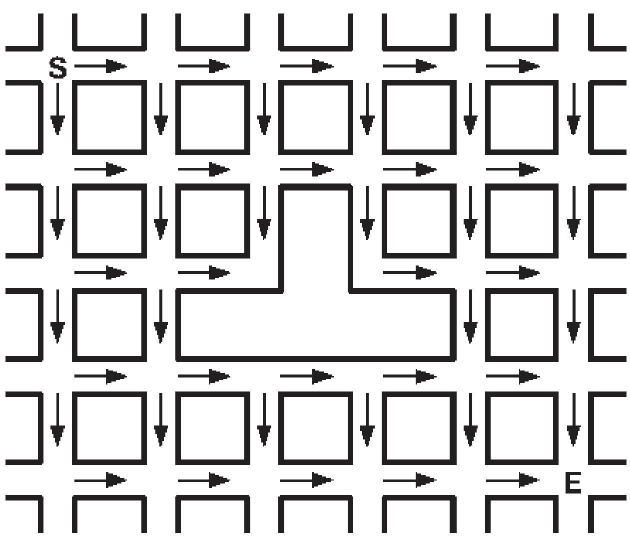

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

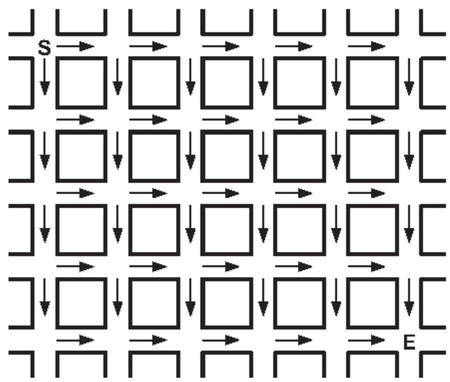

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

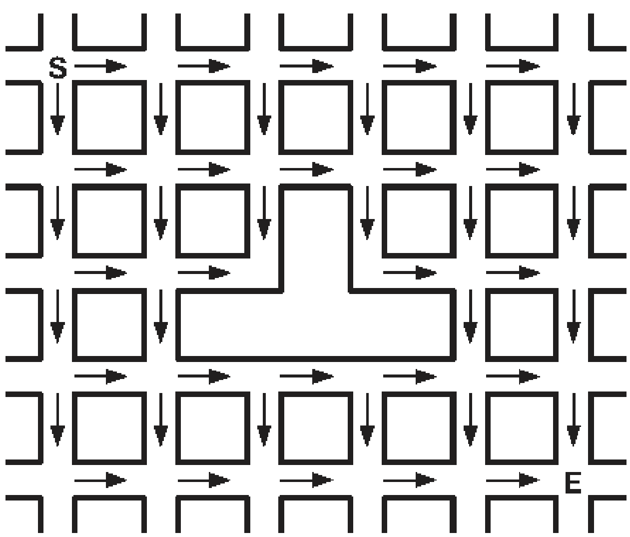

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

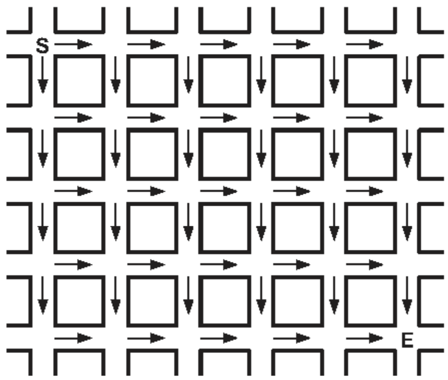

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

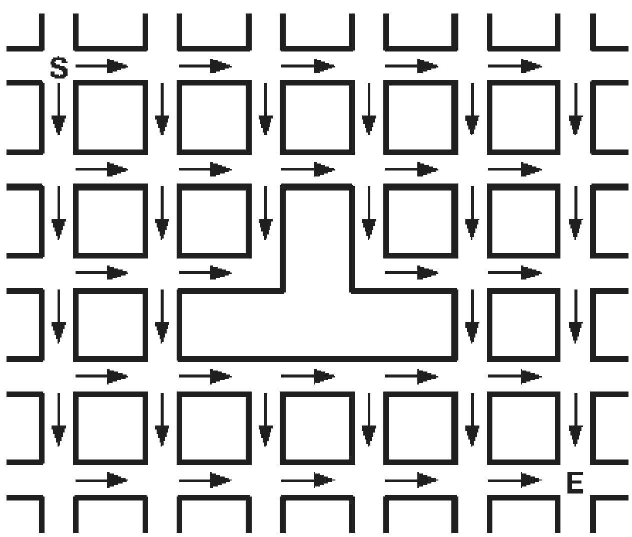

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array $A[1...n]$ of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most $O(\log n)$ pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

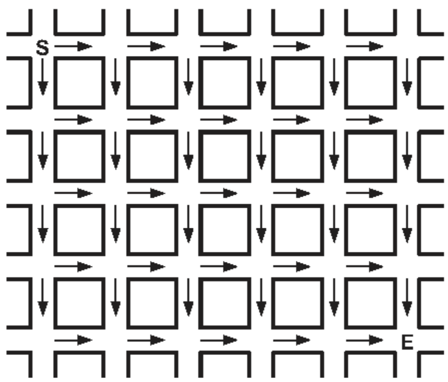

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

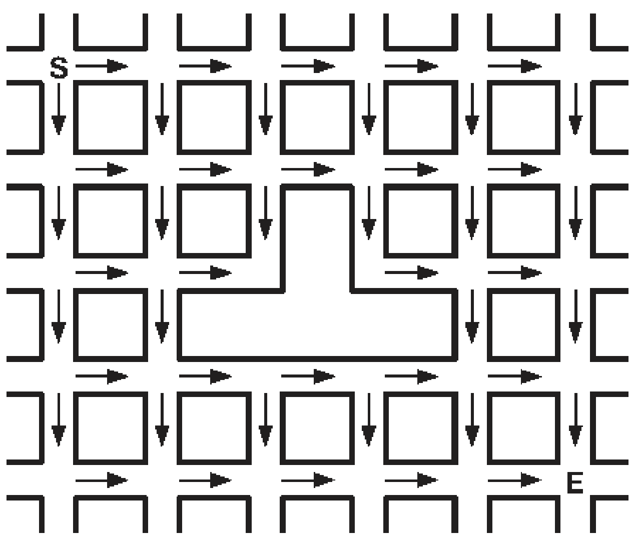

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

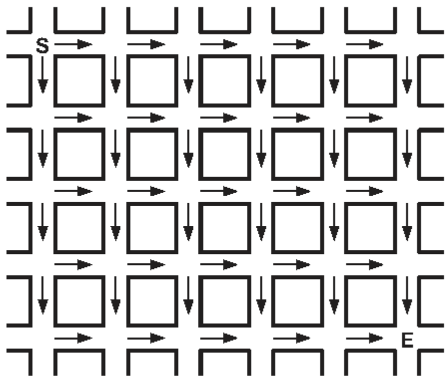

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

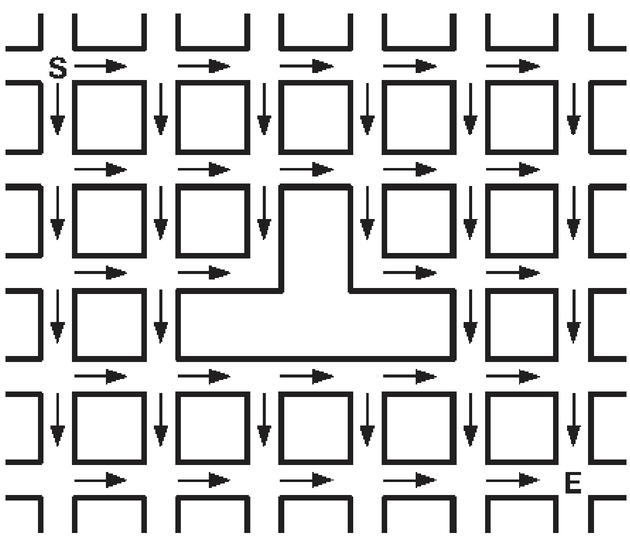

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

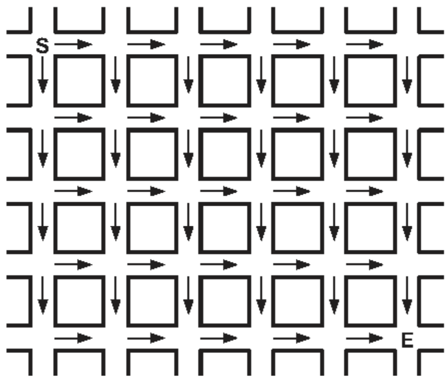

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

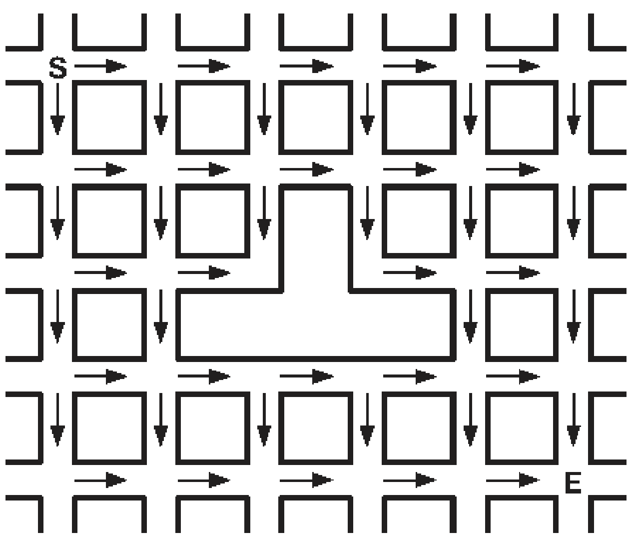

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

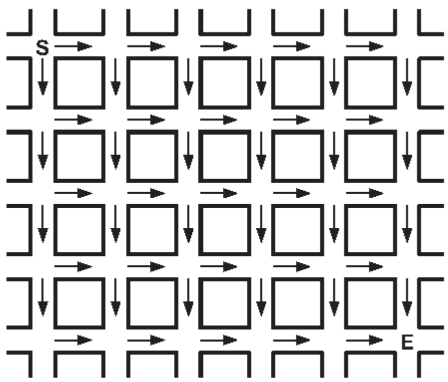

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

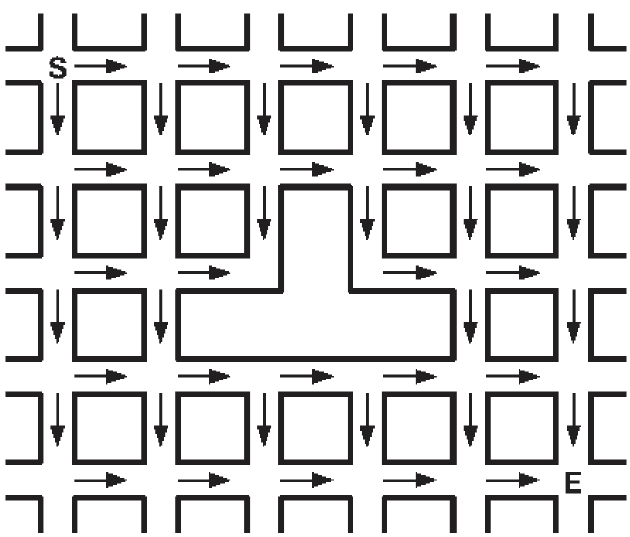

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

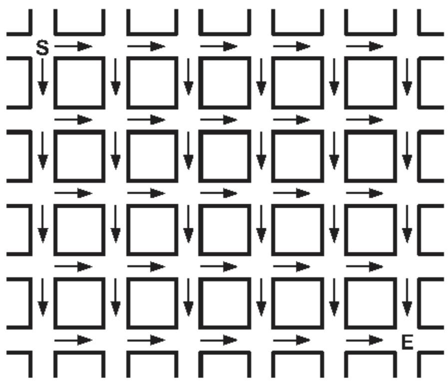

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

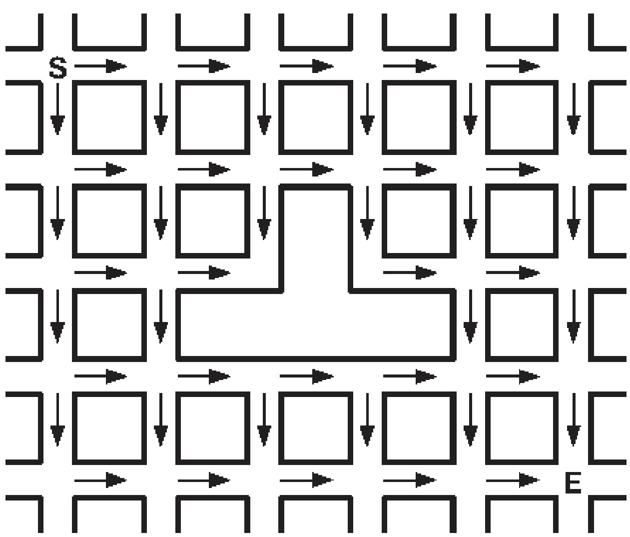

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

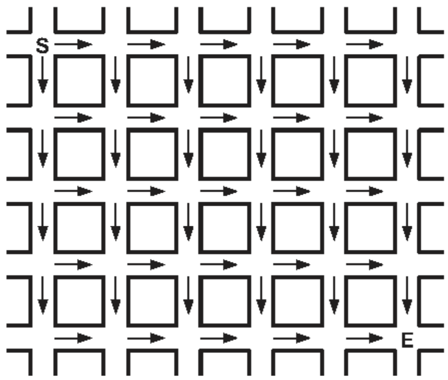

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

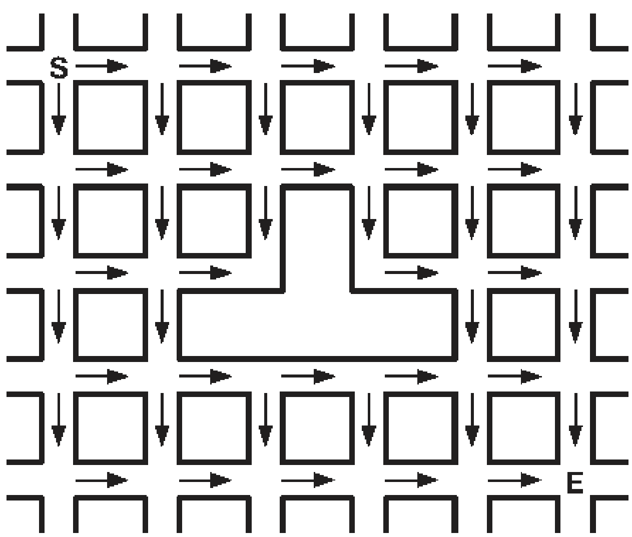

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

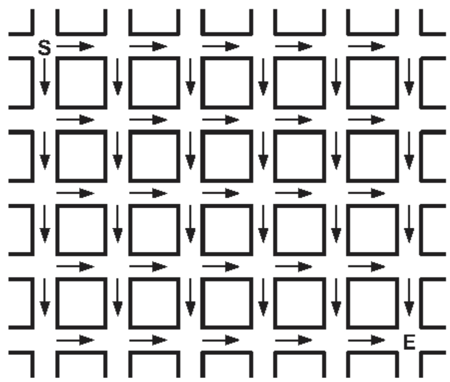

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

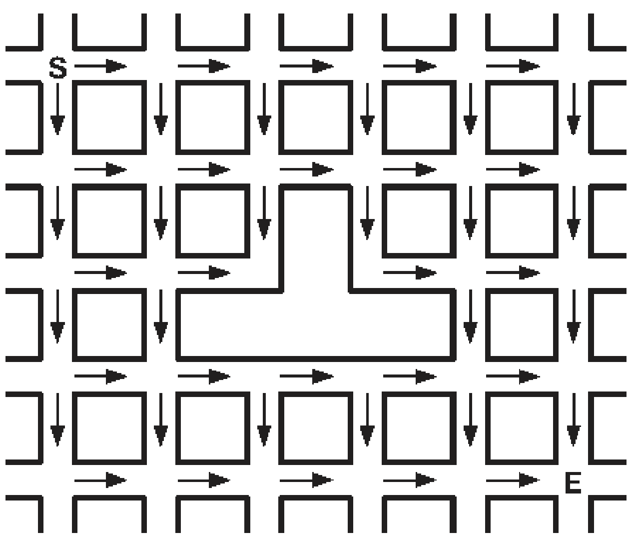

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

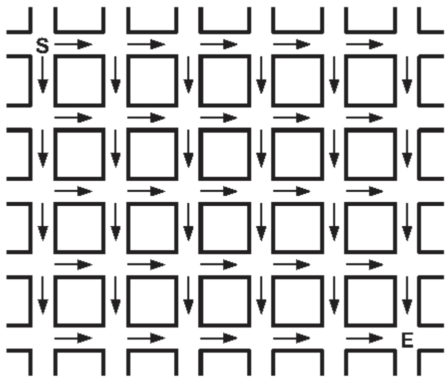

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

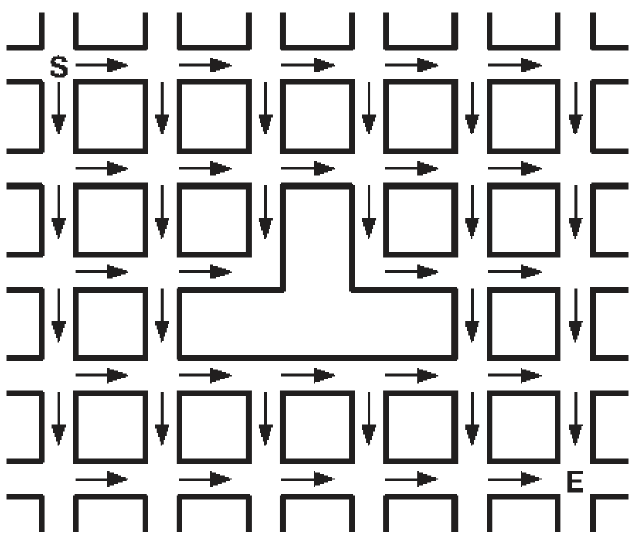

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

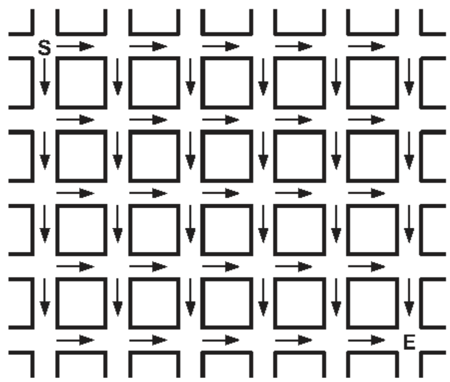

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

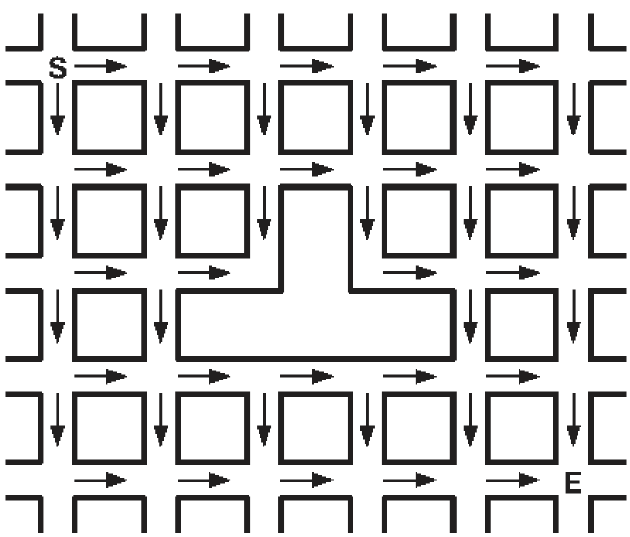

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

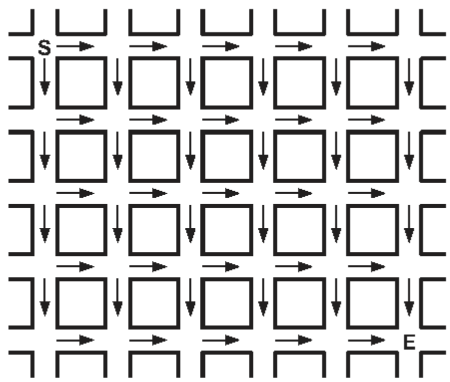

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

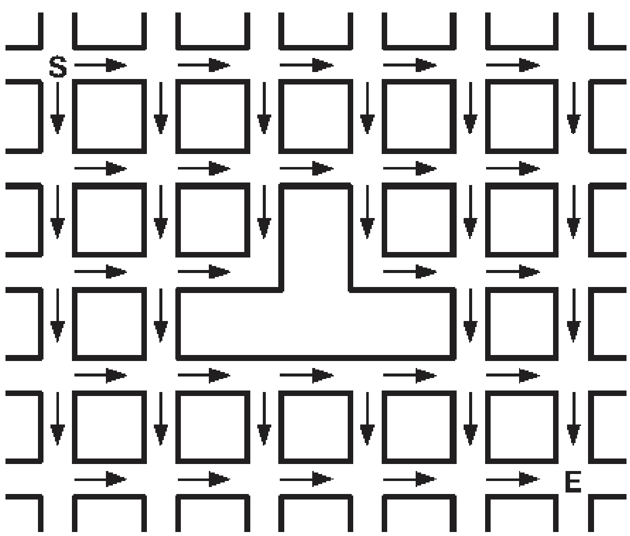

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

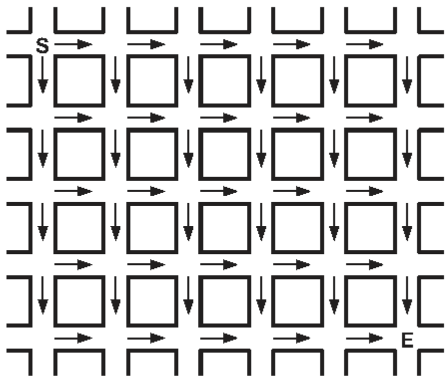

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

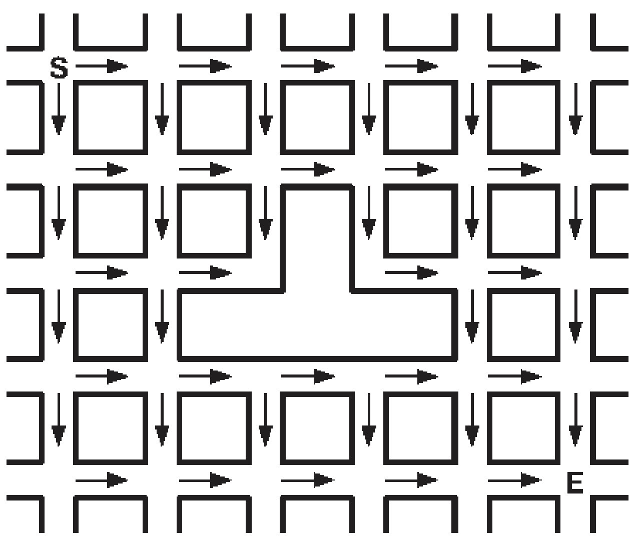

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

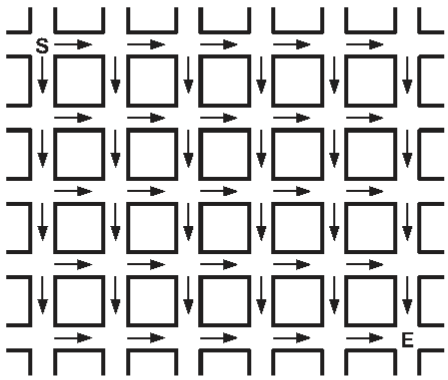

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

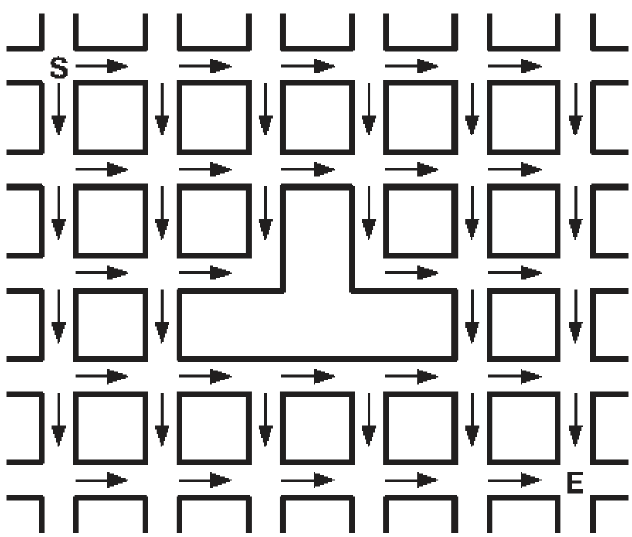

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

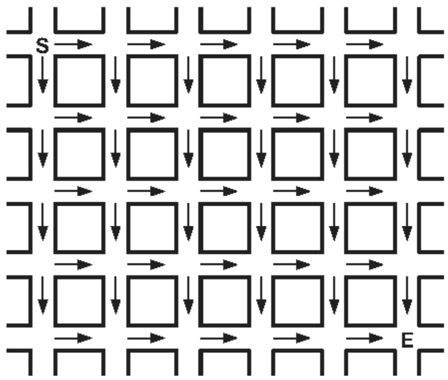

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

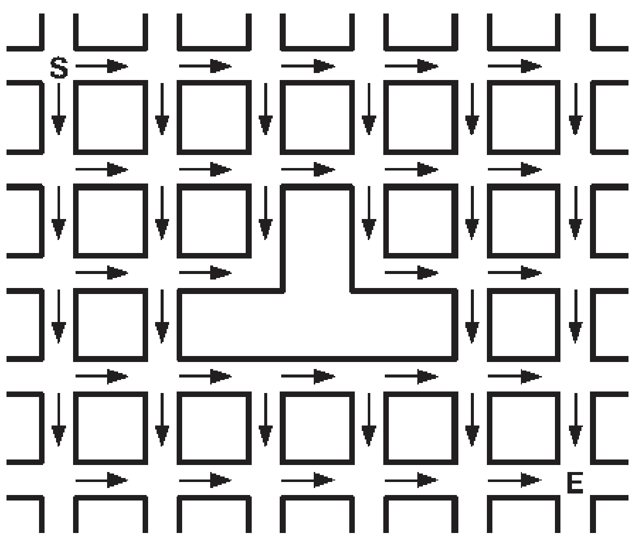

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

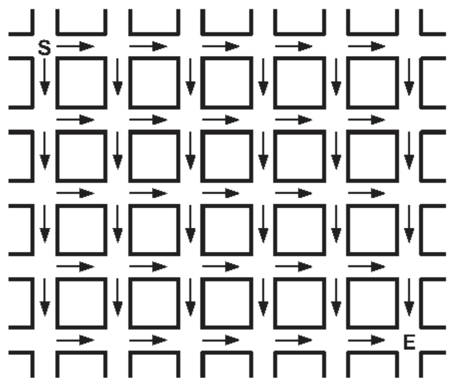

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

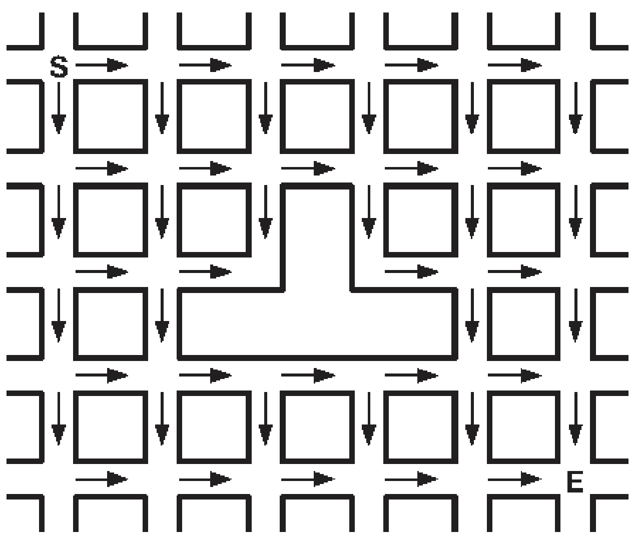

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

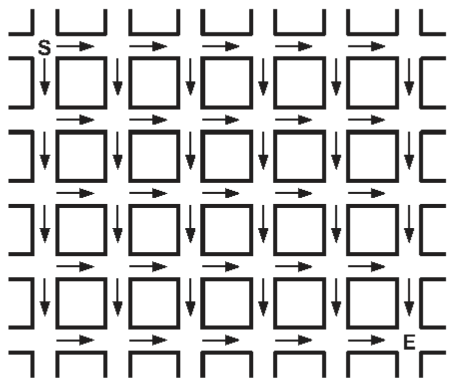

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

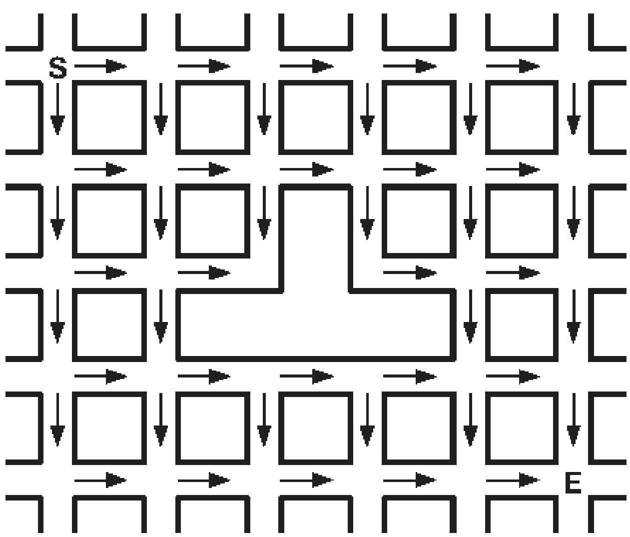

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

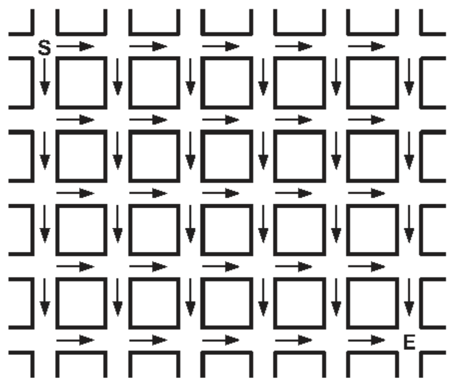

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

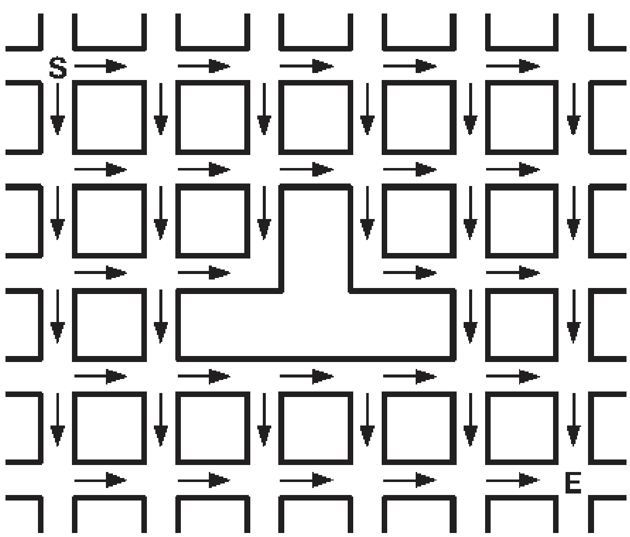

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

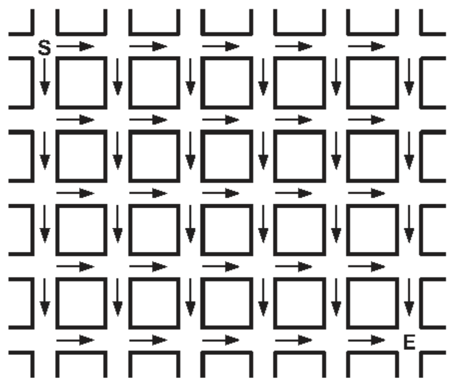

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

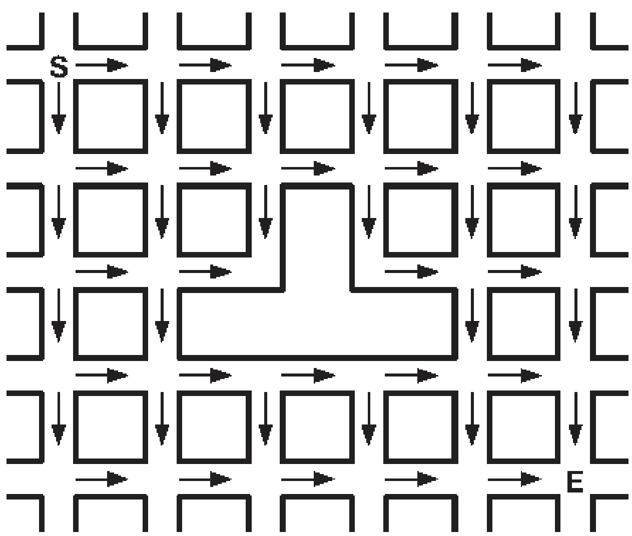

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

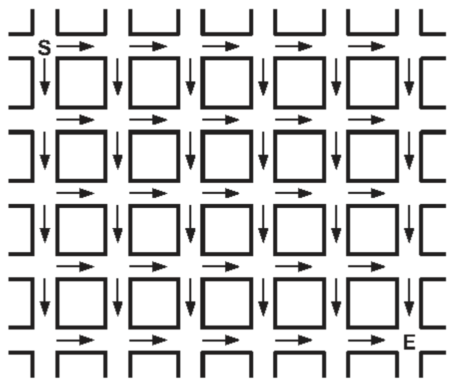

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

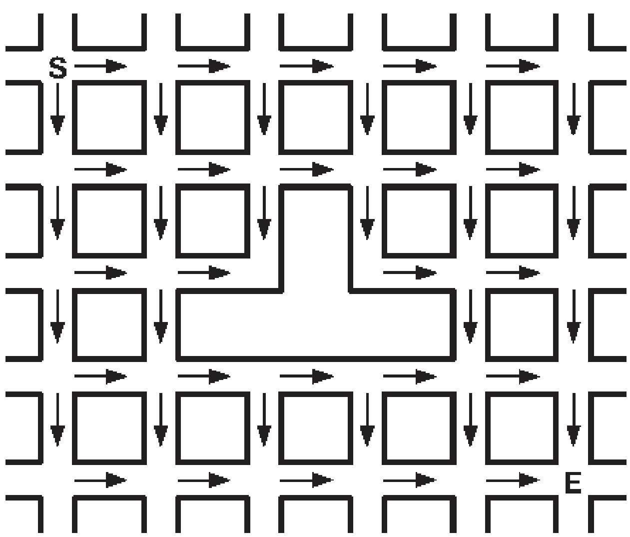

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

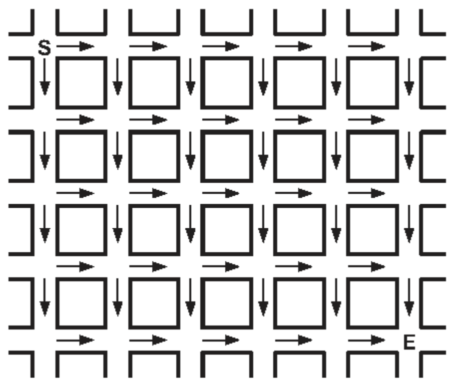

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

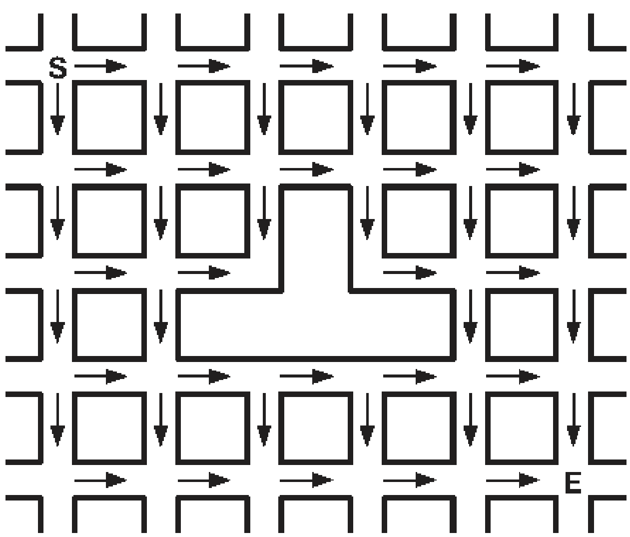

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

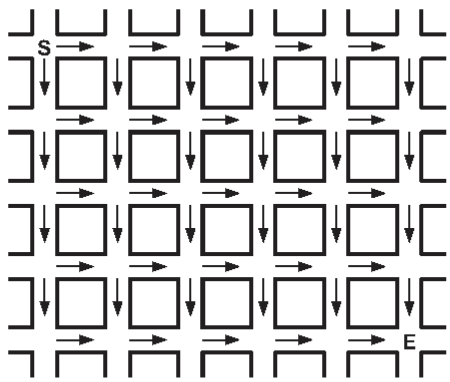

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

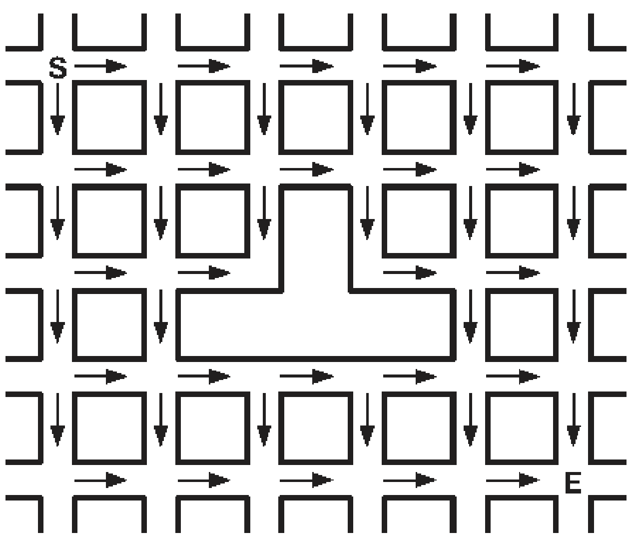

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

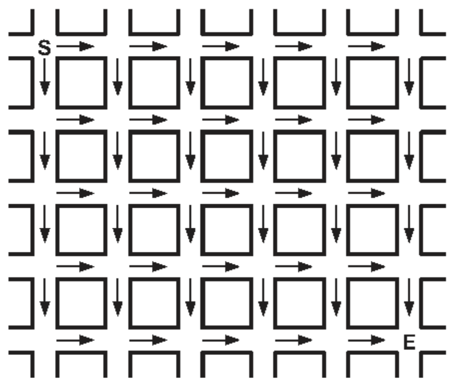

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

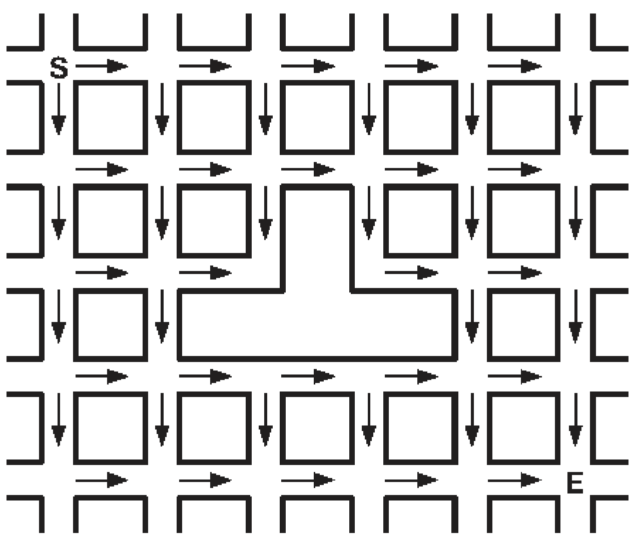

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

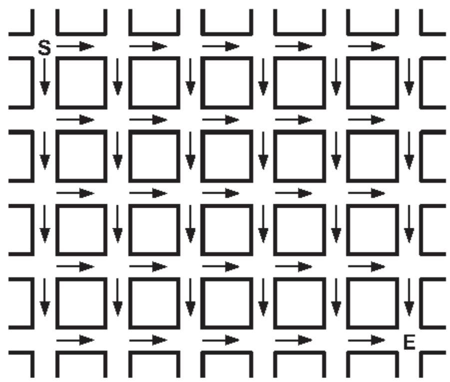

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

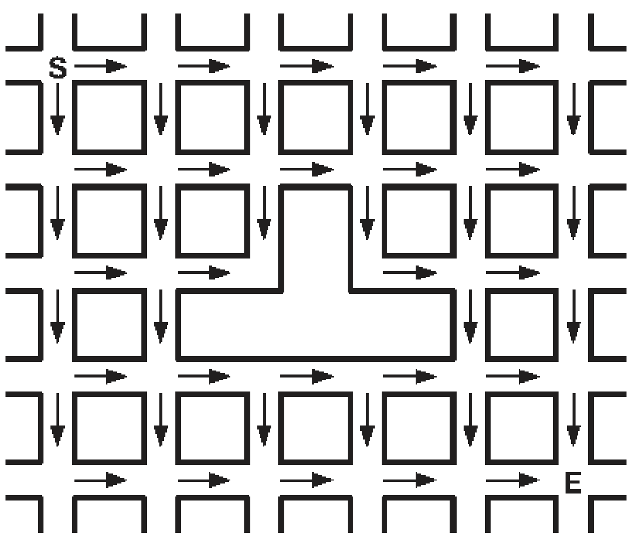

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

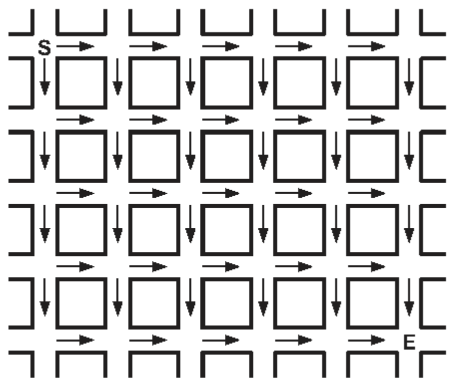

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

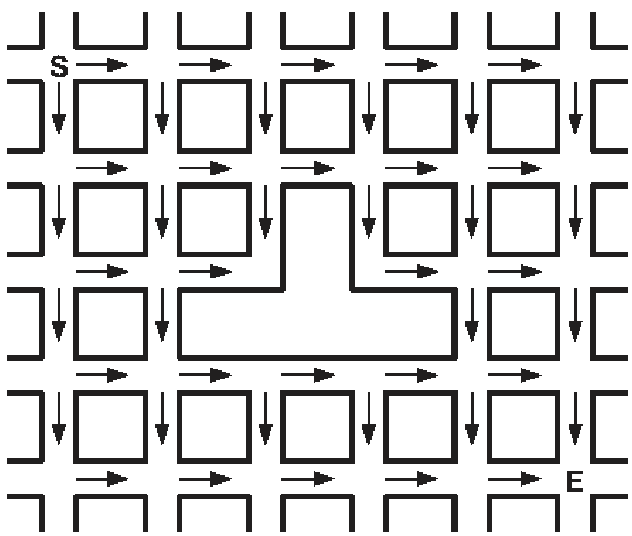

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

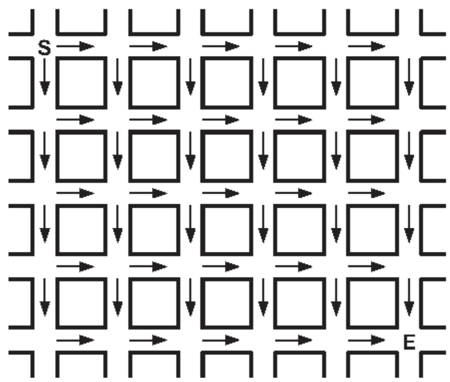

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

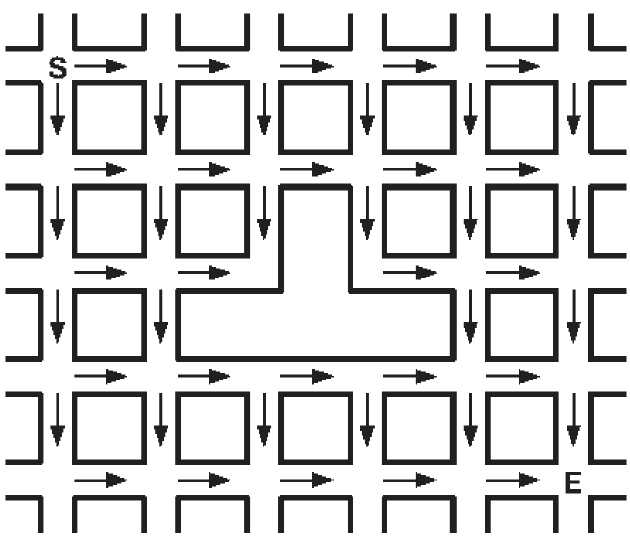

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array $A[1...n]$ of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most $O(\log n)$ pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

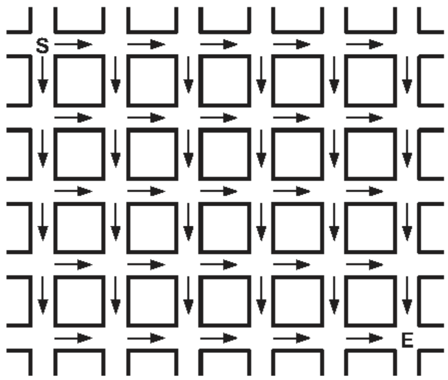

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

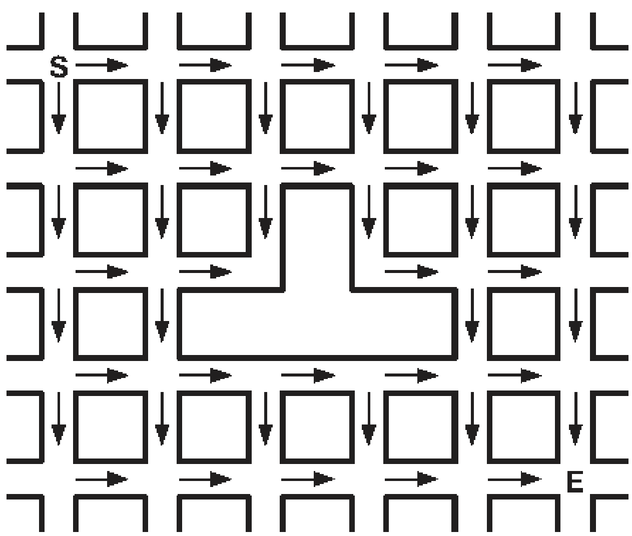

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

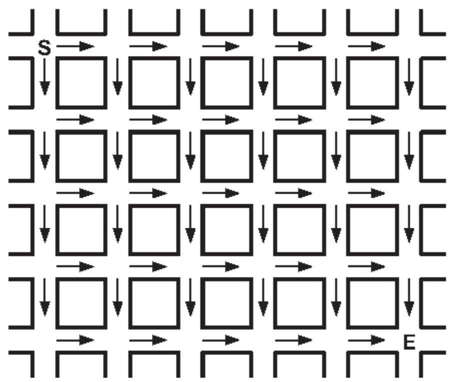

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

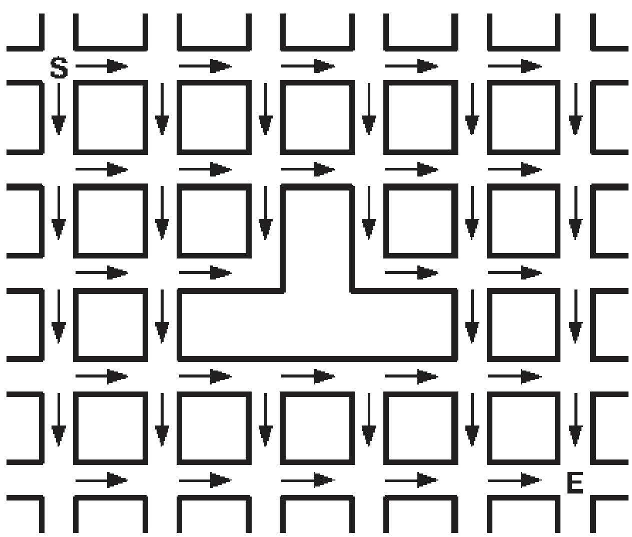

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that $n>2$, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that $1<i<n$, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

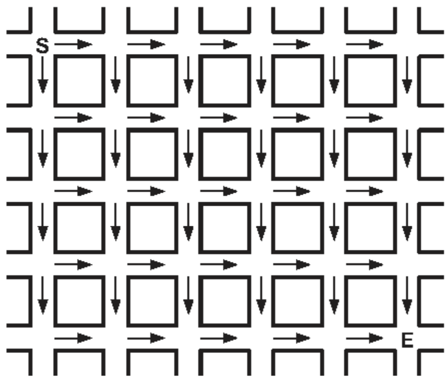

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

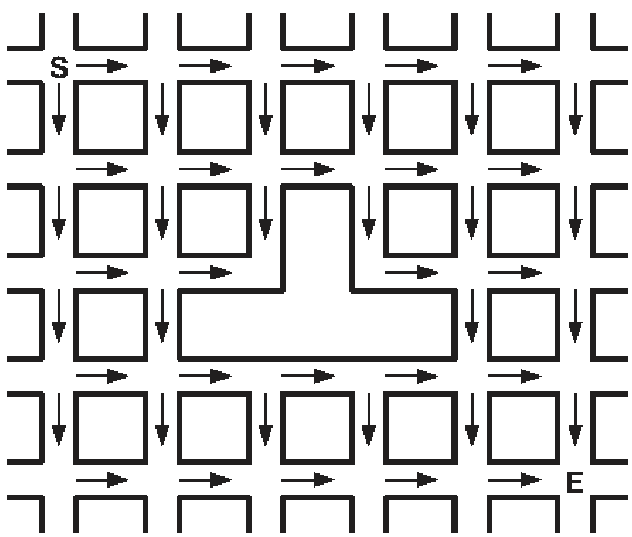

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

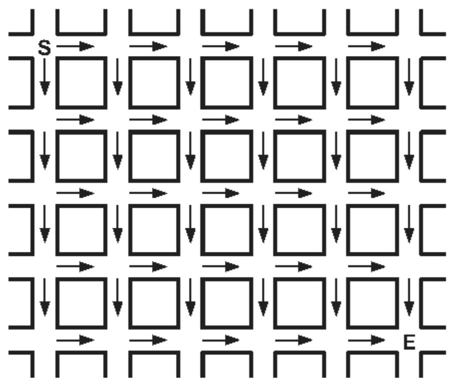

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

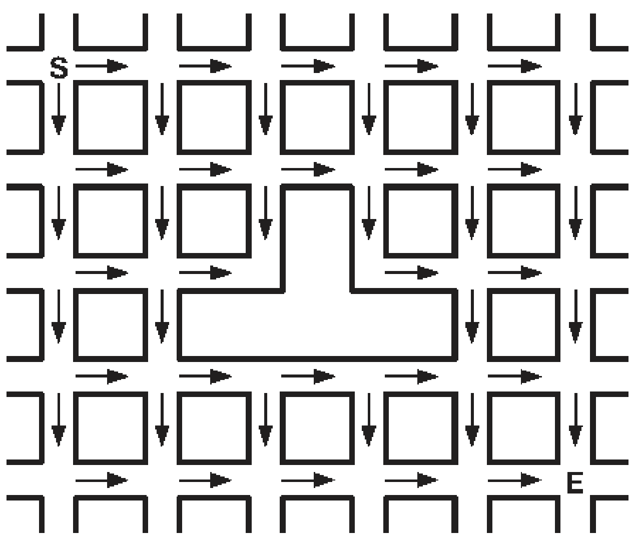

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

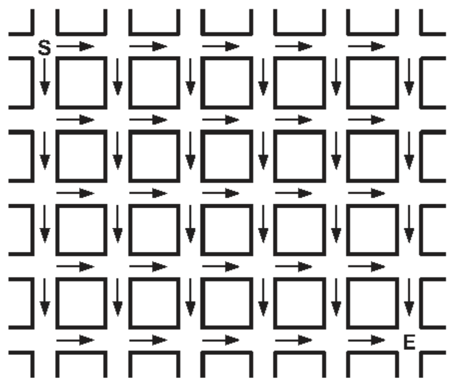

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

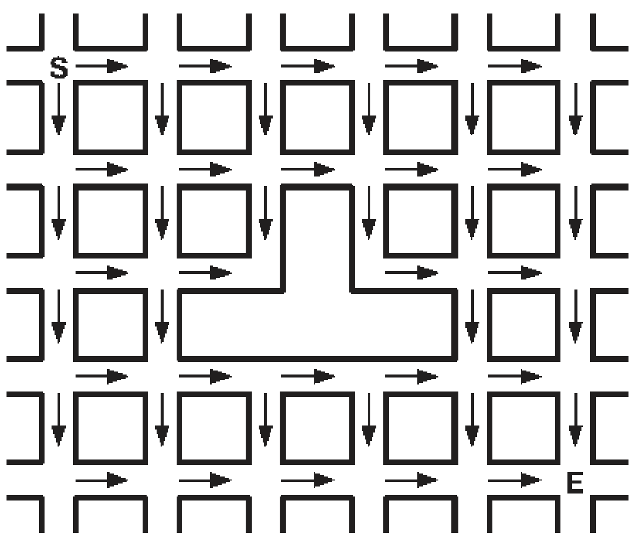

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

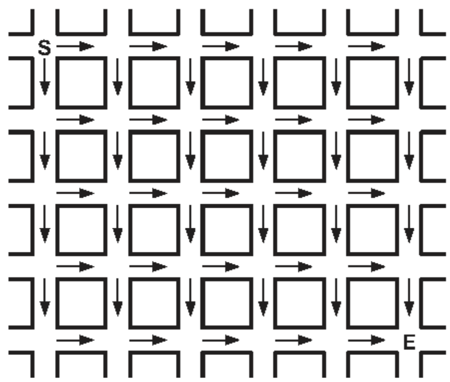

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

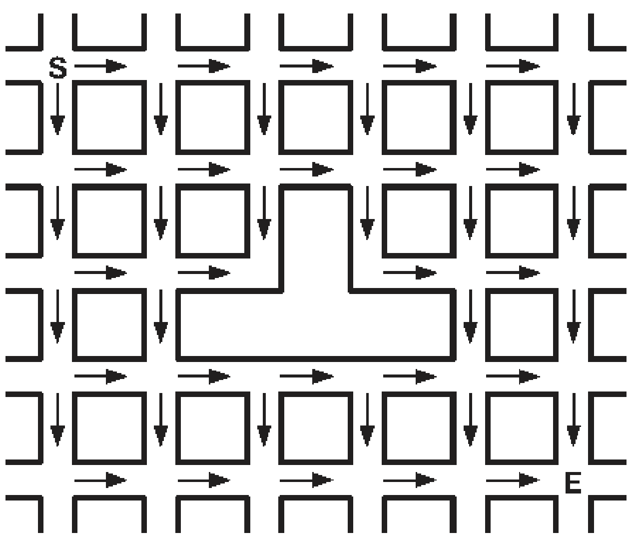

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

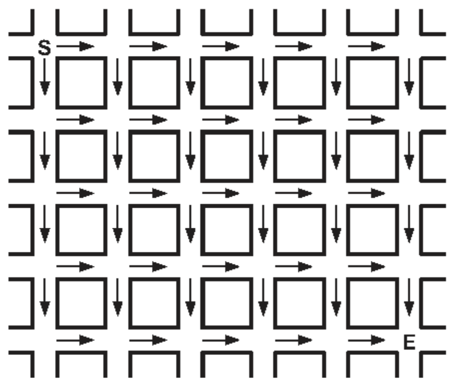

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

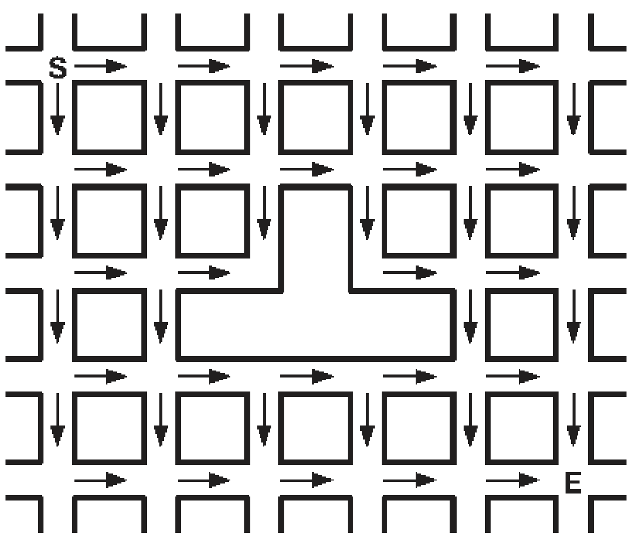

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

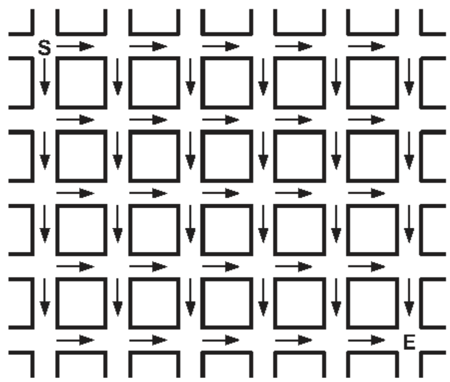

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

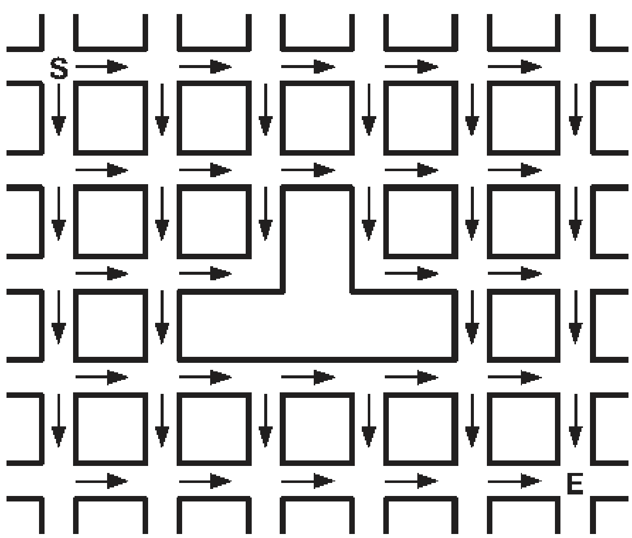

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

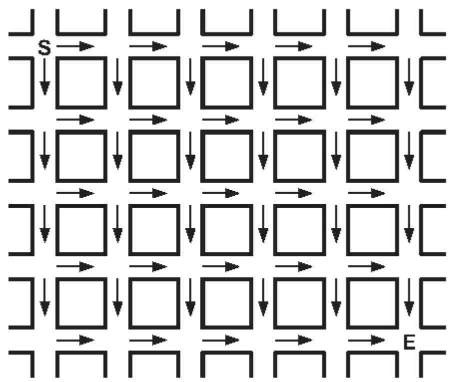

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

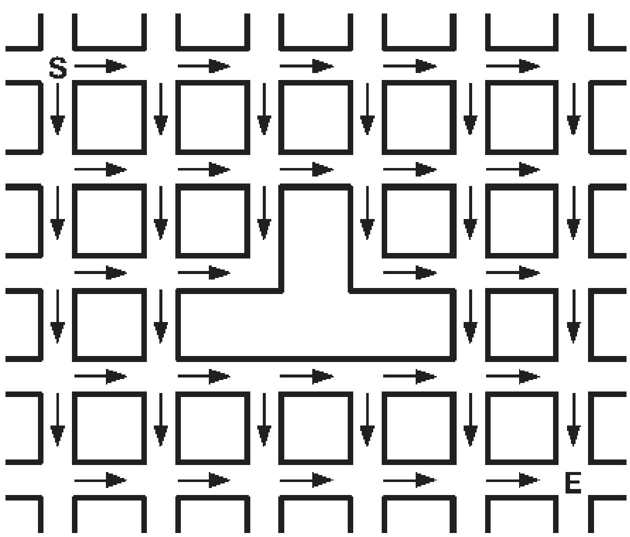

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

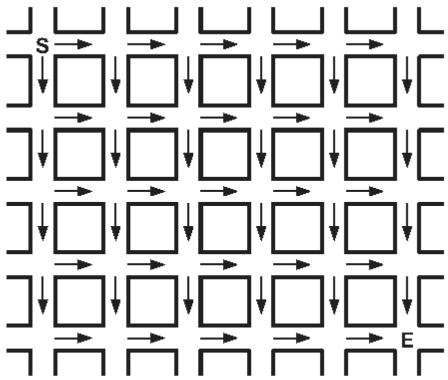

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

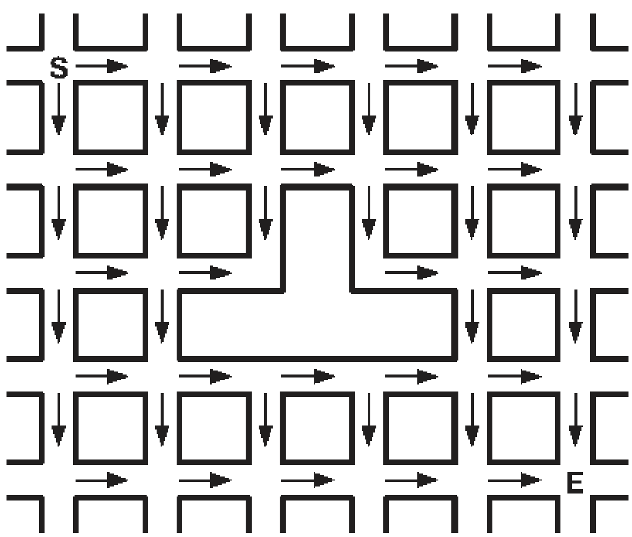

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

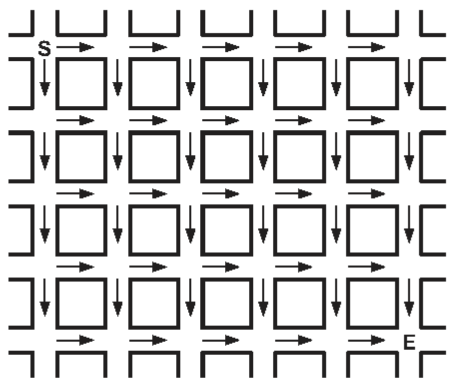

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

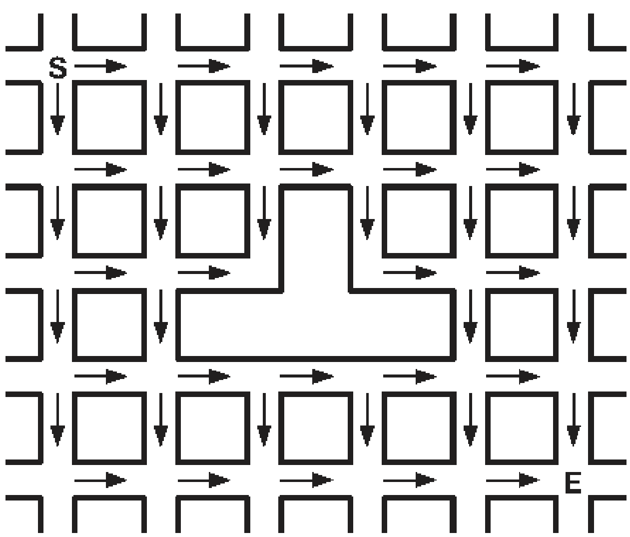

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

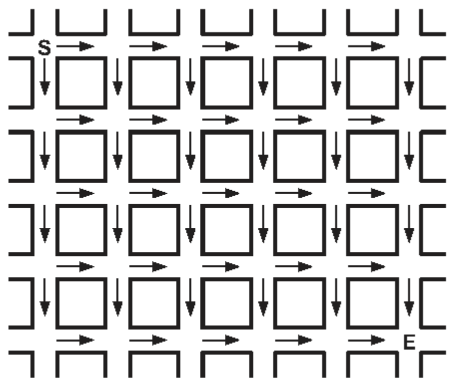

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

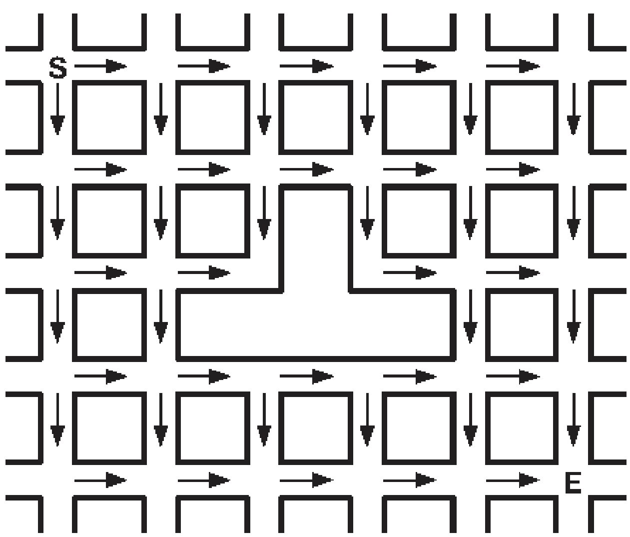

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array $A[1...n]$ of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

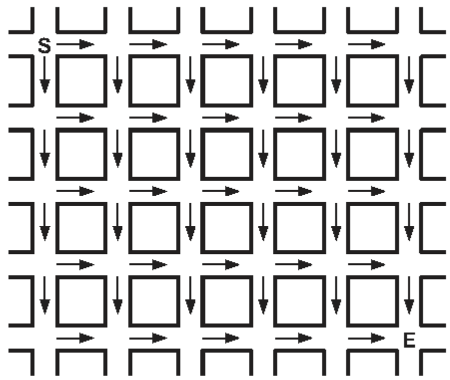

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

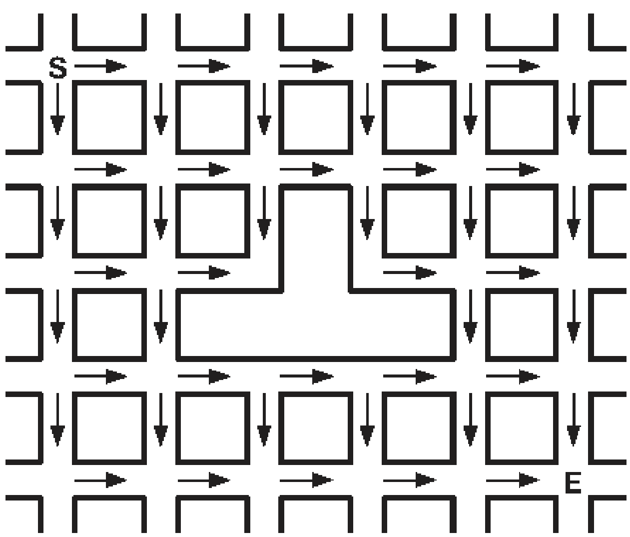

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

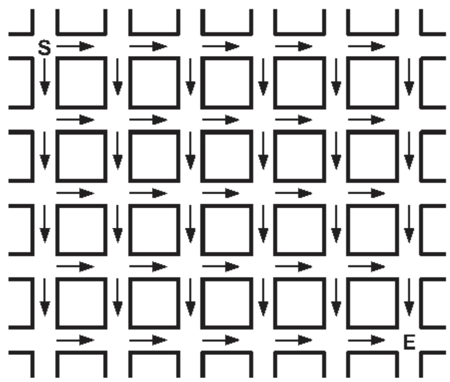

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

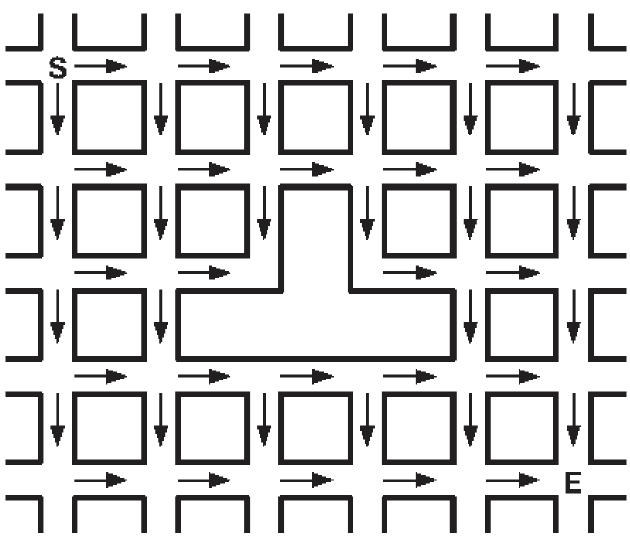

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

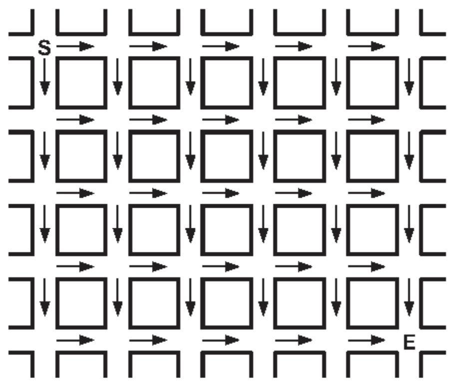

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

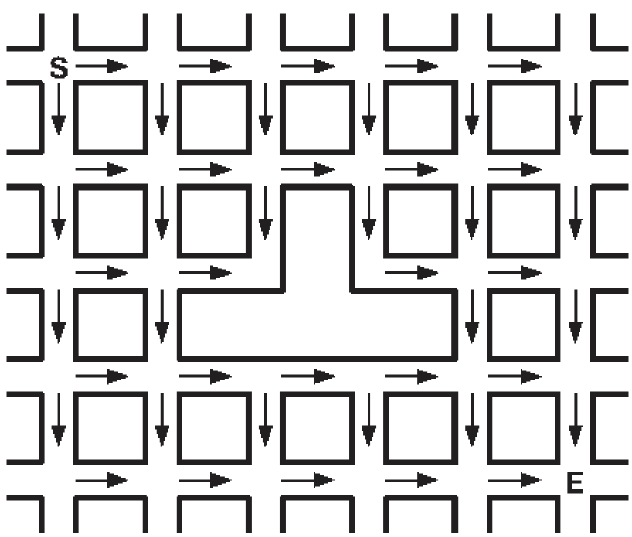

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

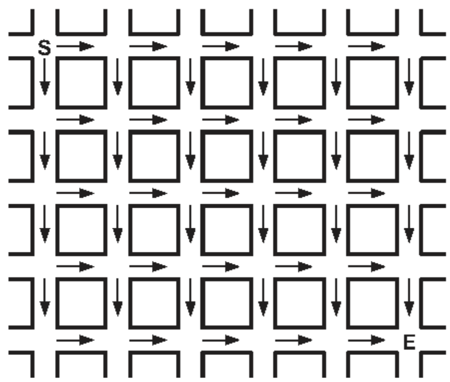

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

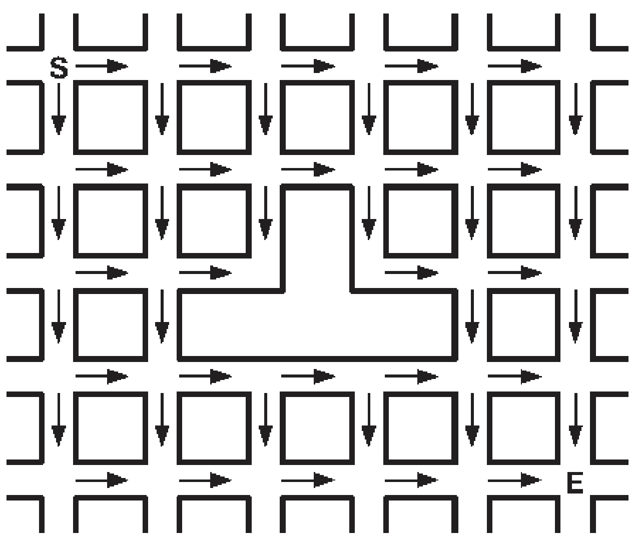

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

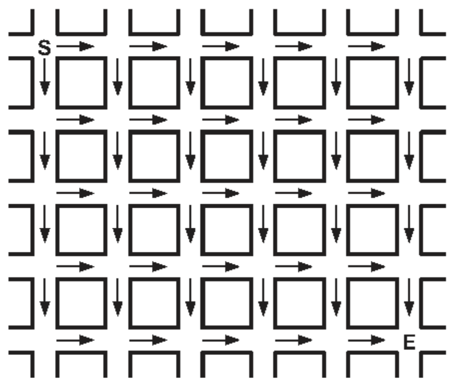

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

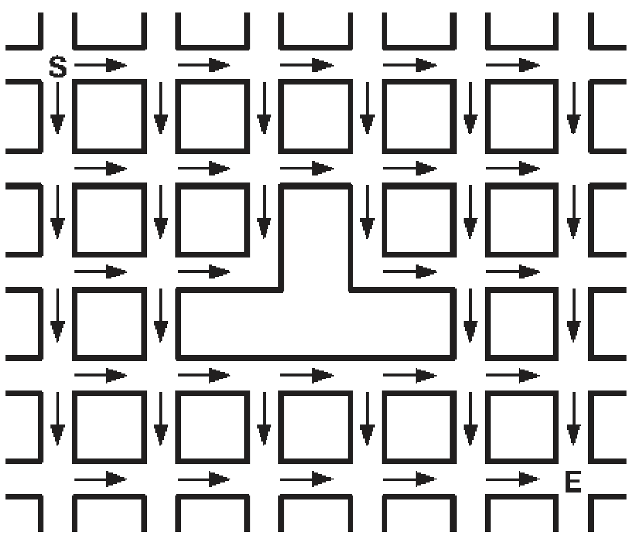

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

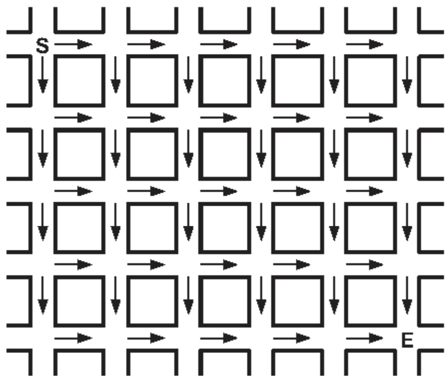

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

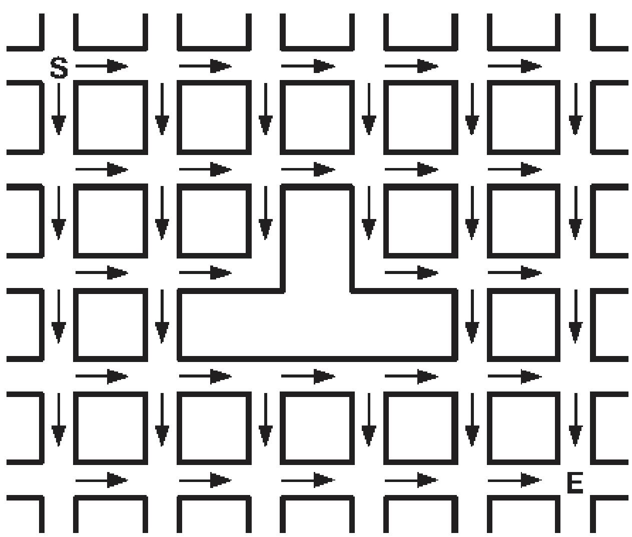

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

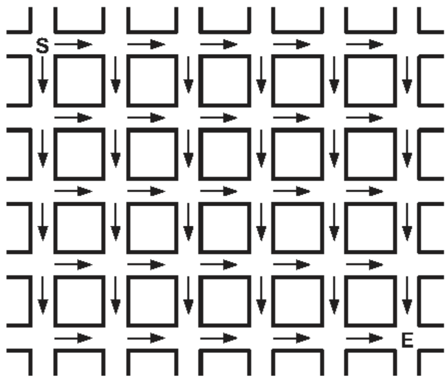

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

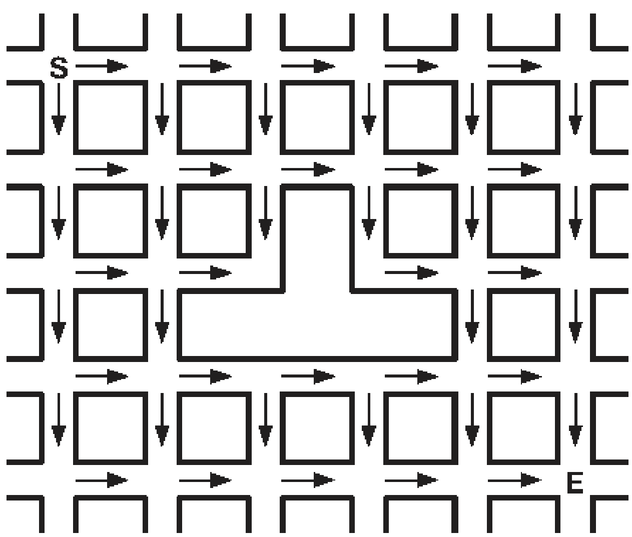

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

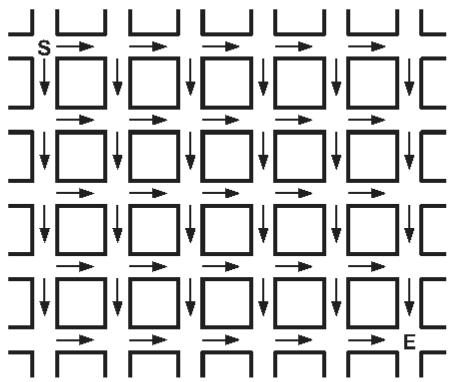

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

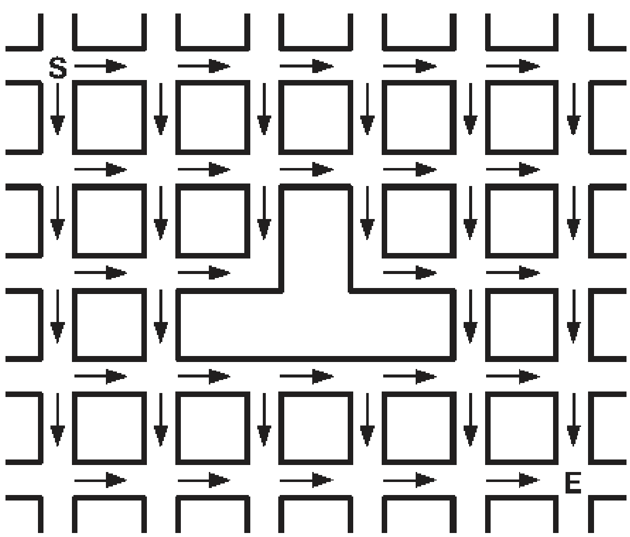

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

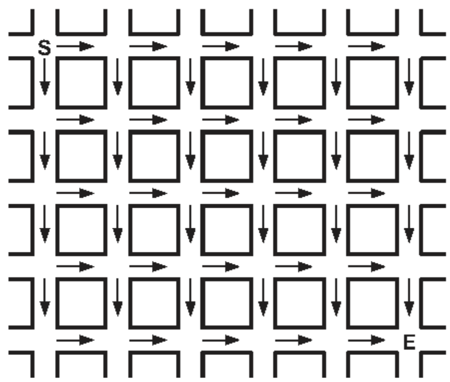

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

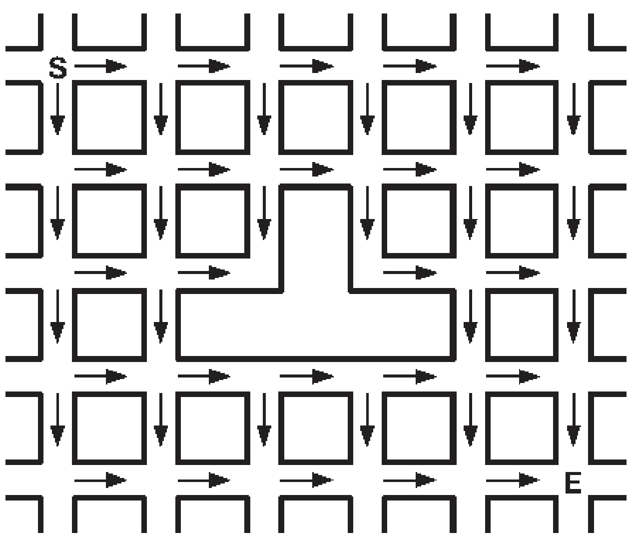

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

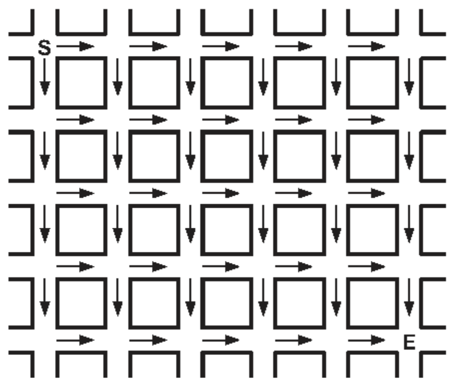

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

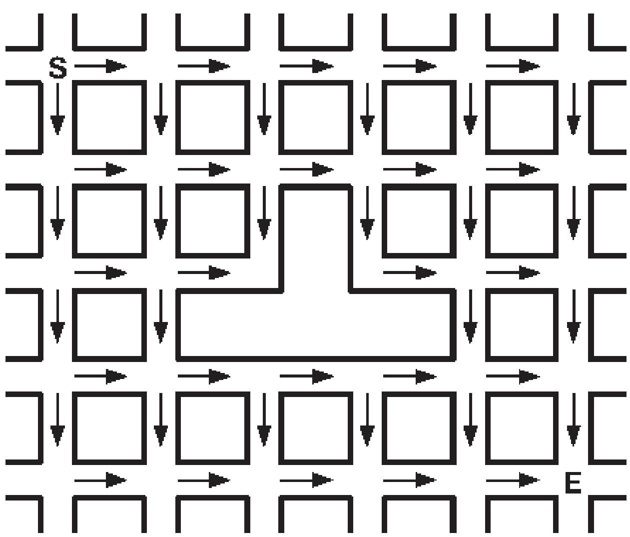

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

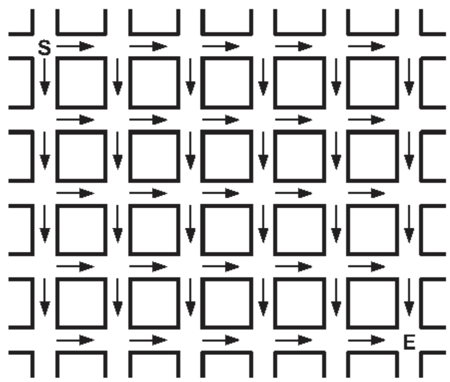

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

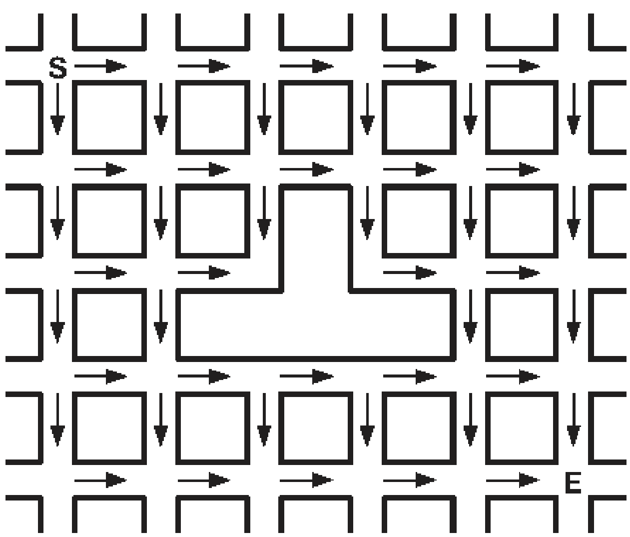

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

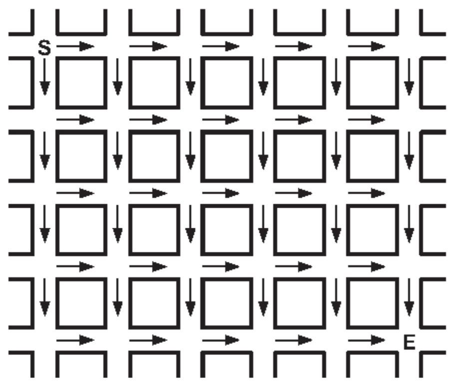

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

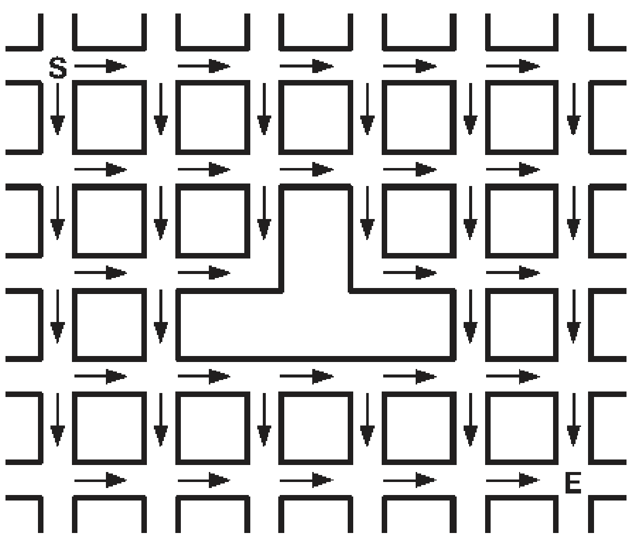

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

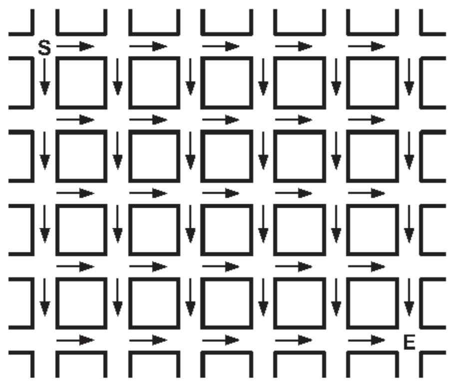

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

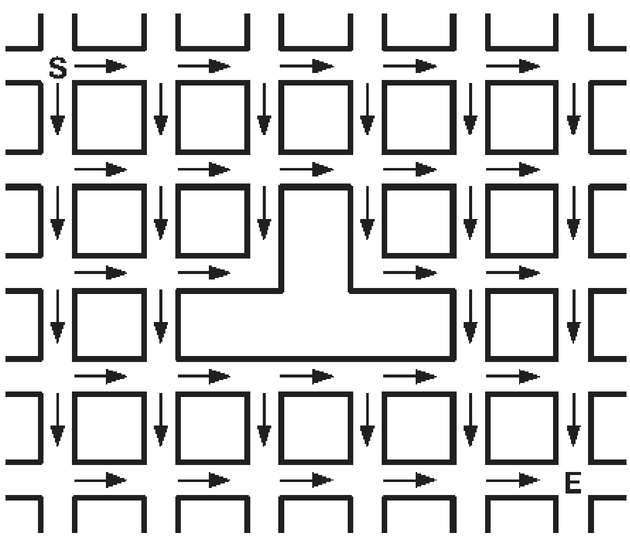

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

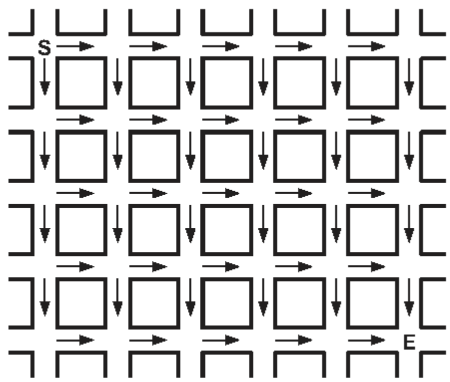

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

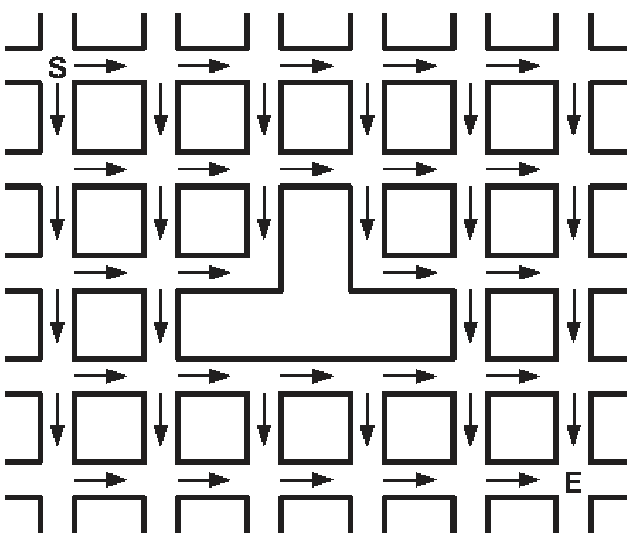

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

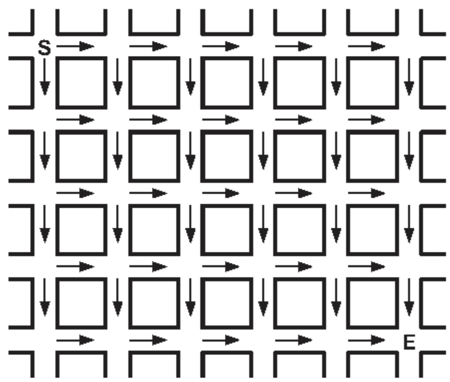

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

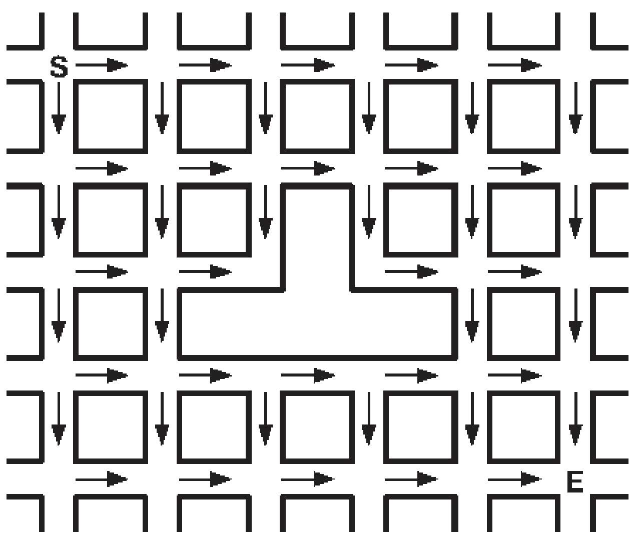

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

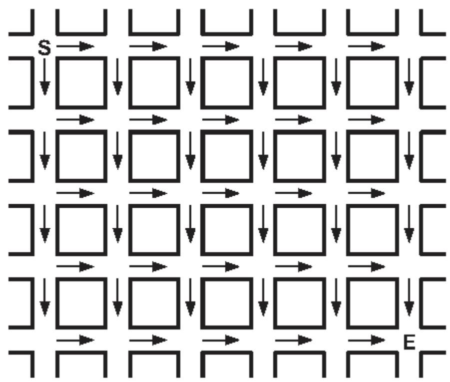

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

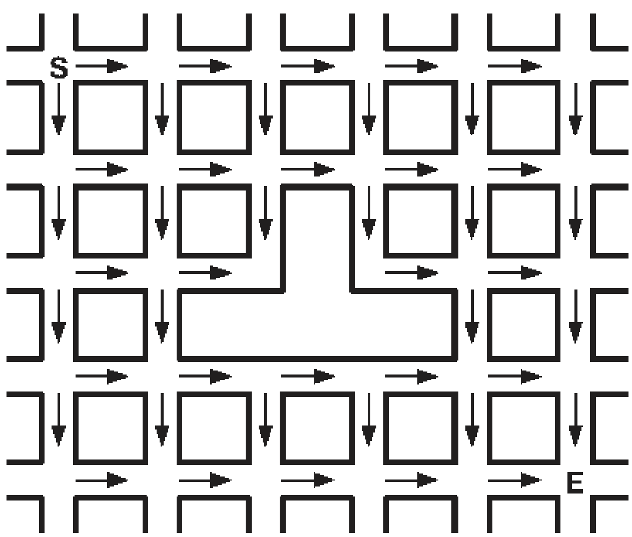

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that $1<i<n$, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

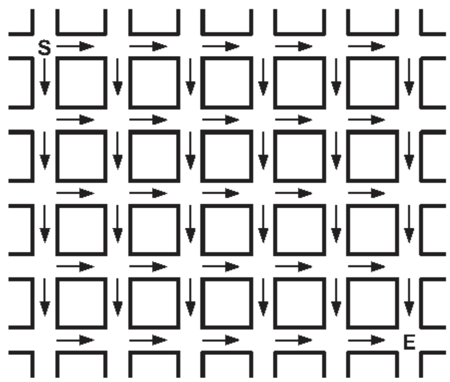

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

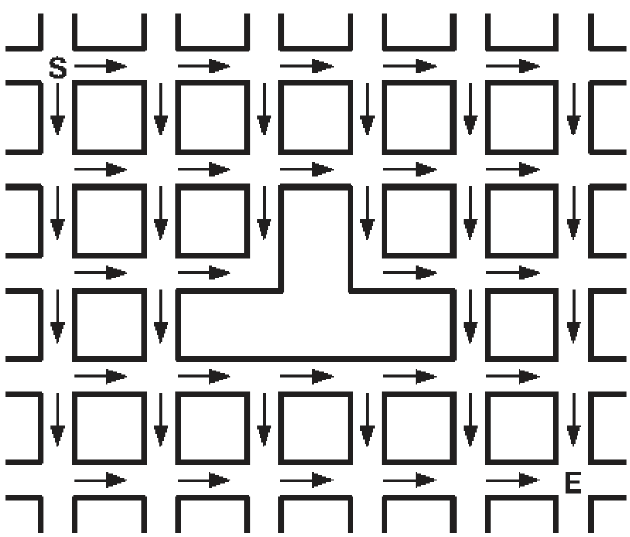

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

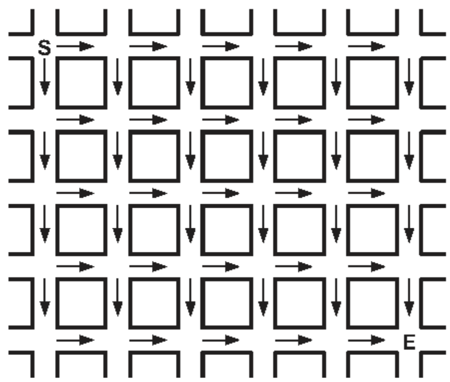

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

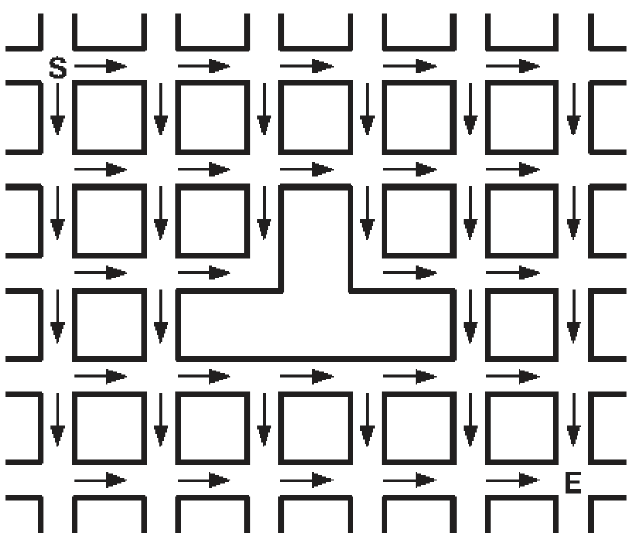

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

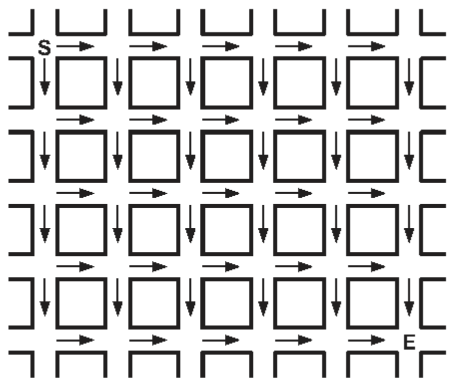

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

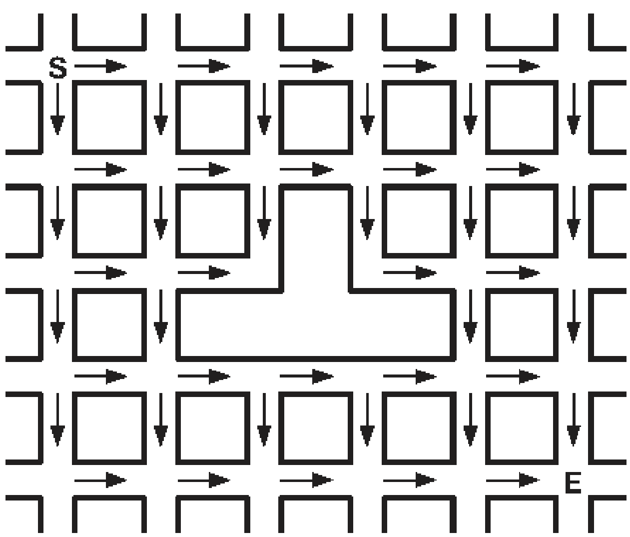

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

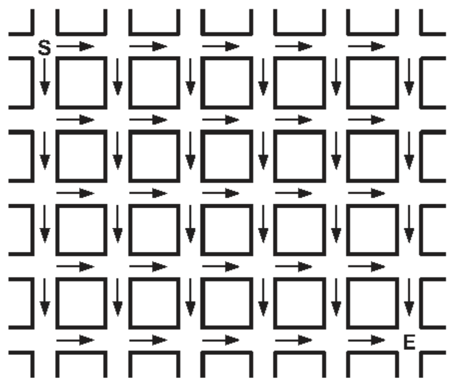

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

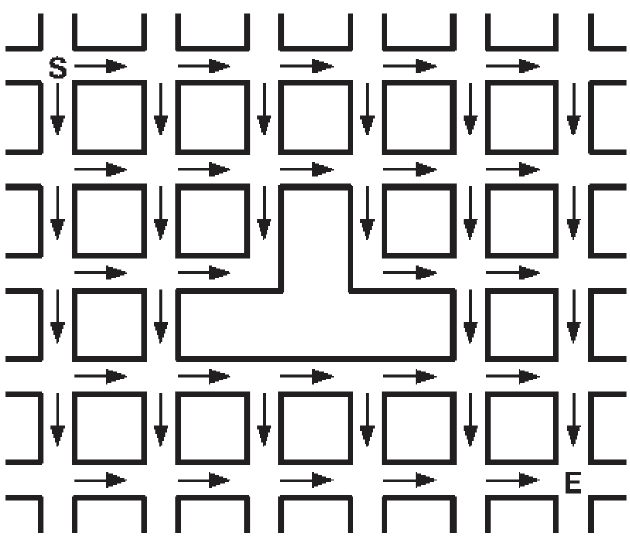

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

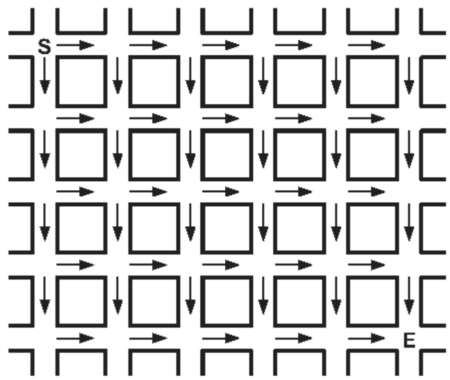

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

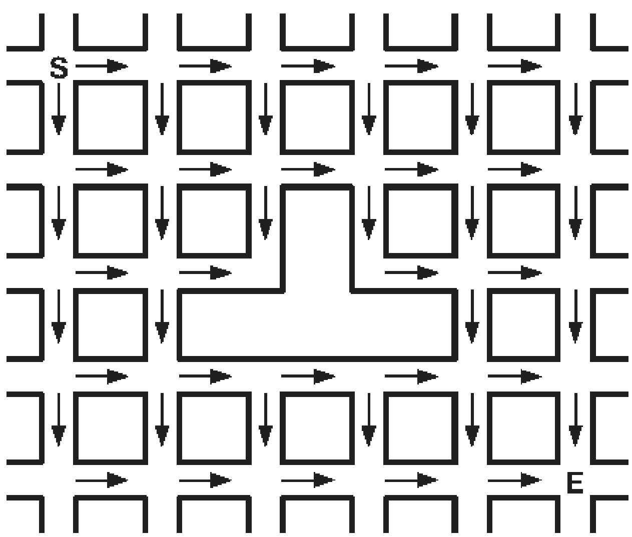

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

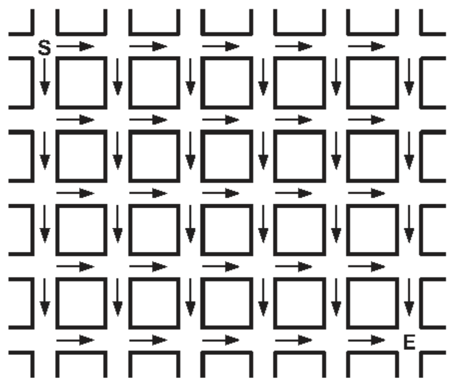

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

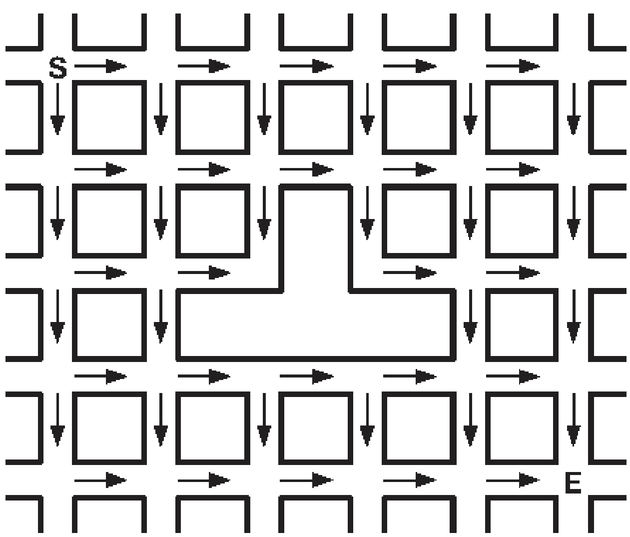

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array $A[1...n]$ of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most $O(\log n)$ pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

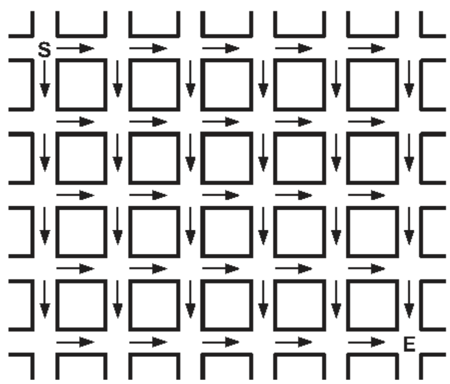

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

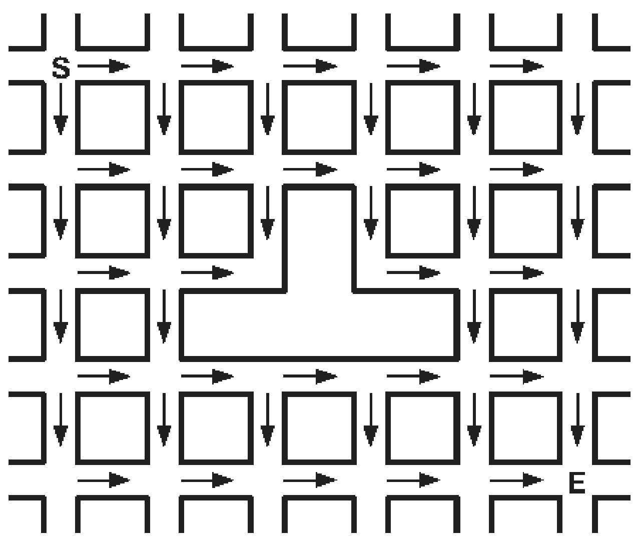

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

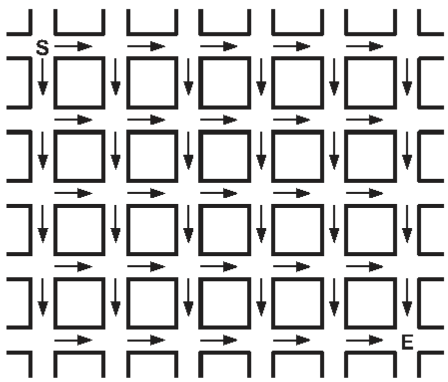

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

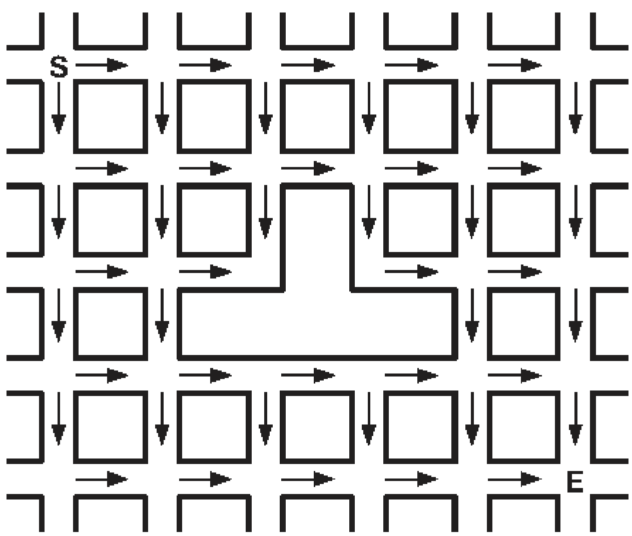

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

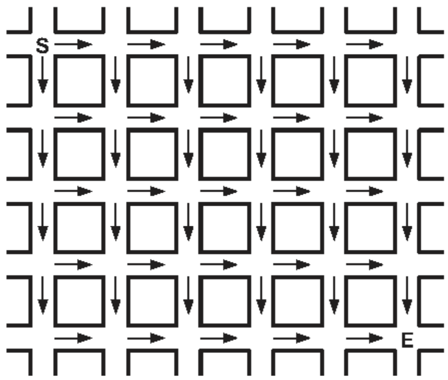

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

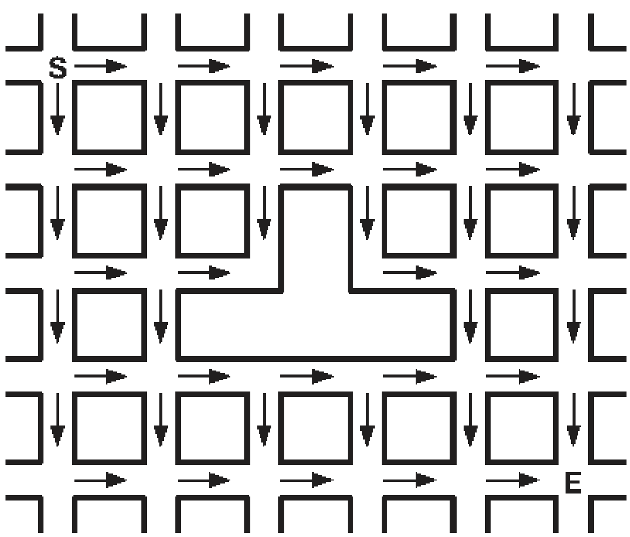

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

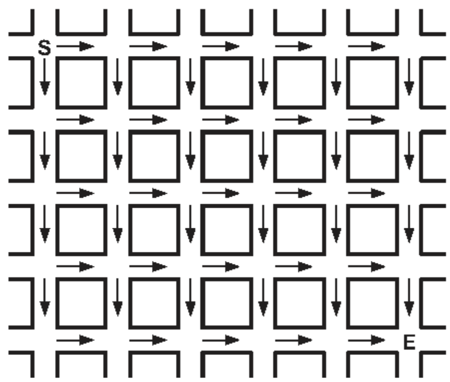

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

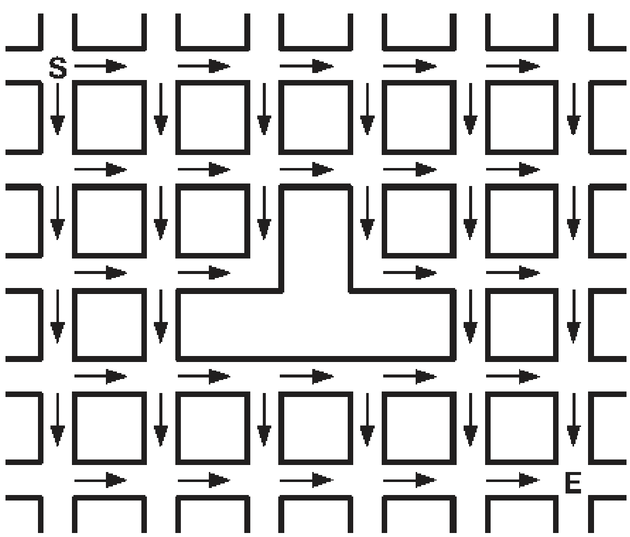

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

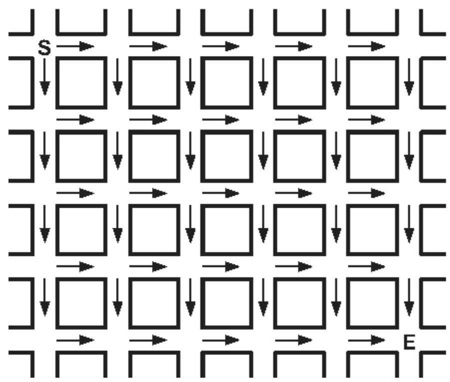

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

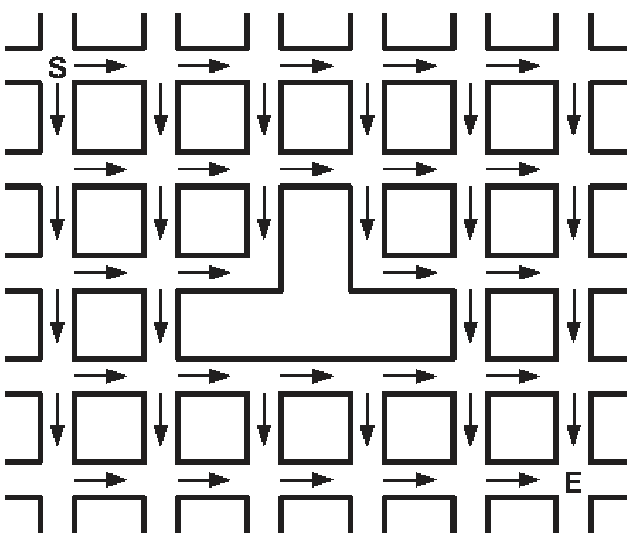

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

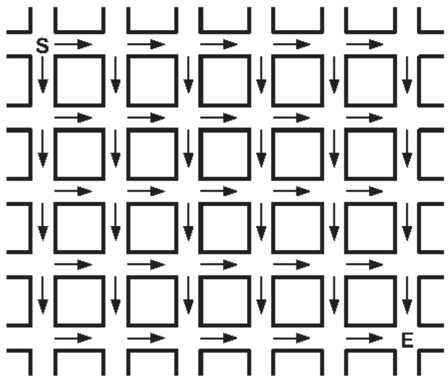

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

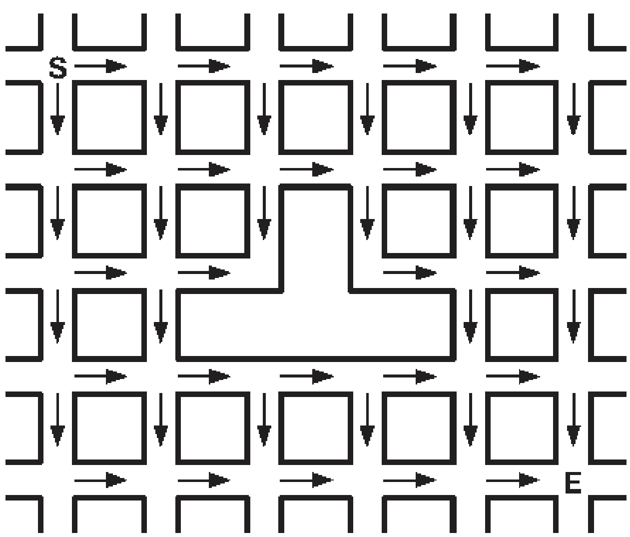

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

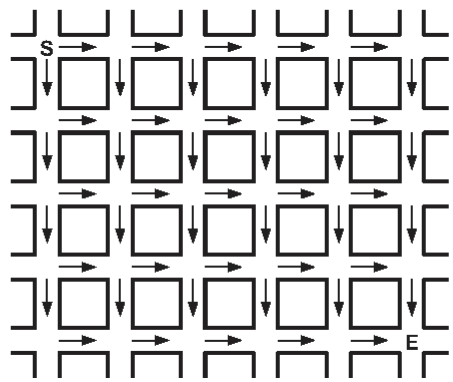

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

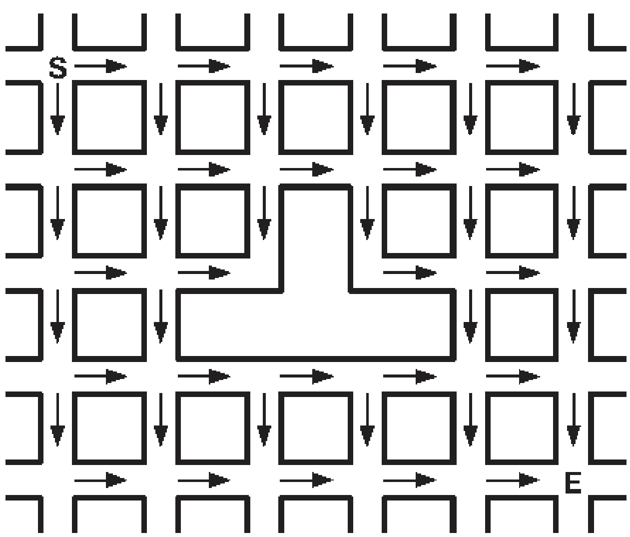

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

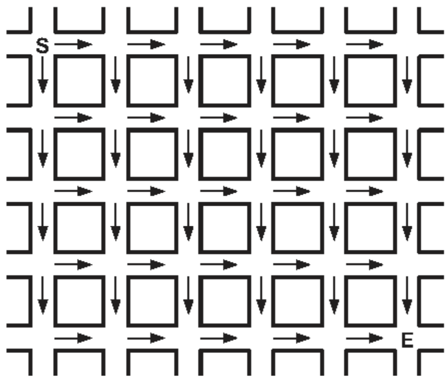

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

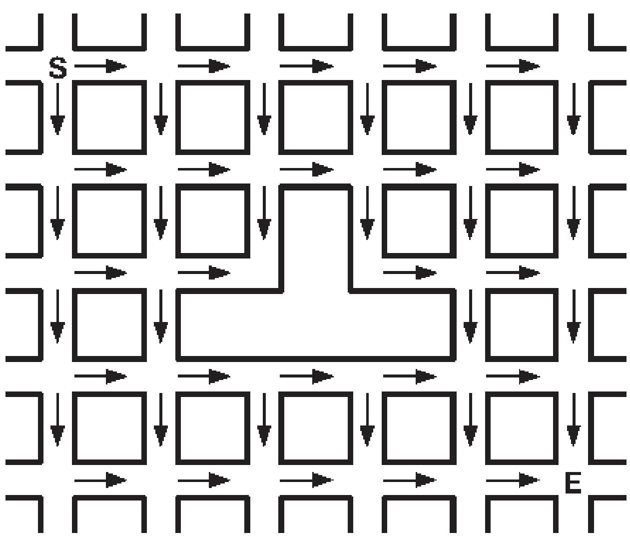

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

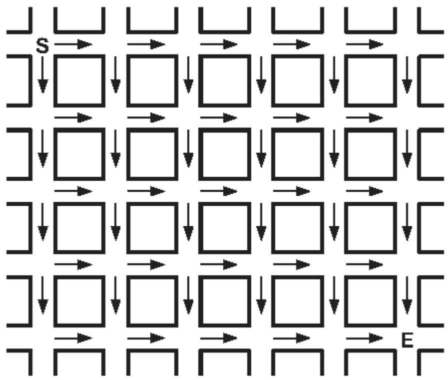

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

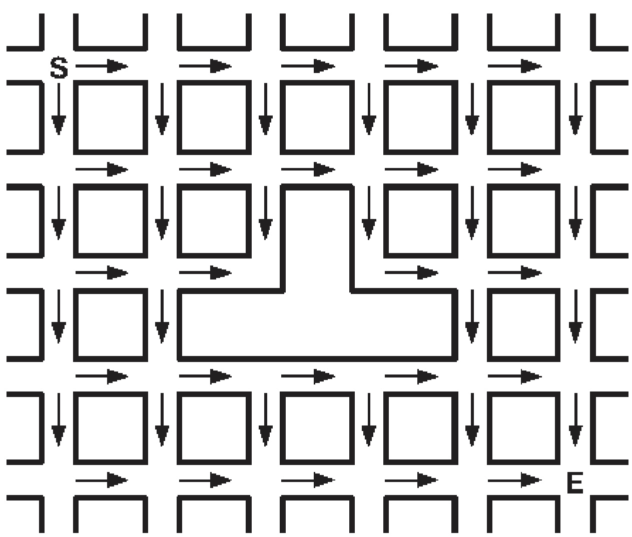

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

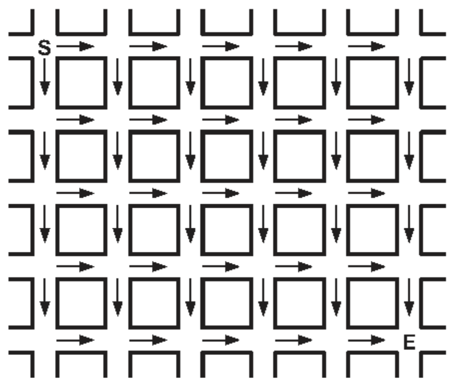

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

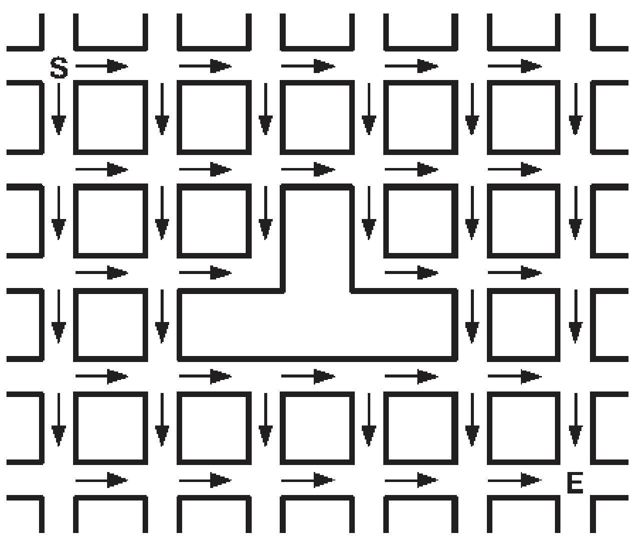

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

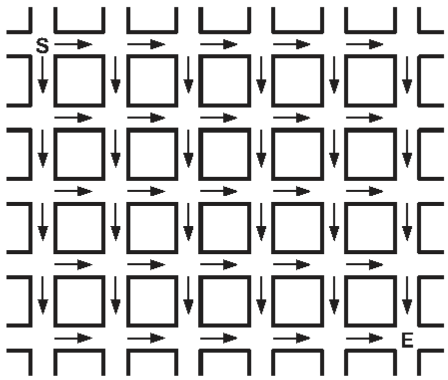

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

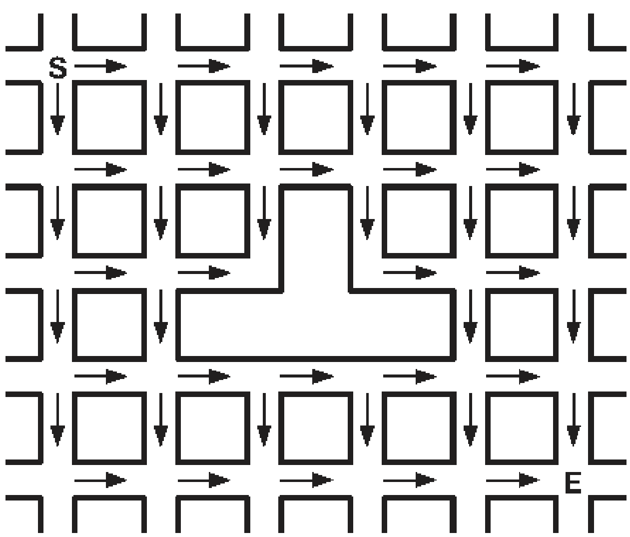

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

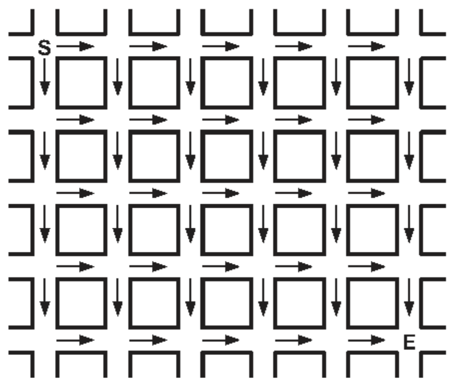

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

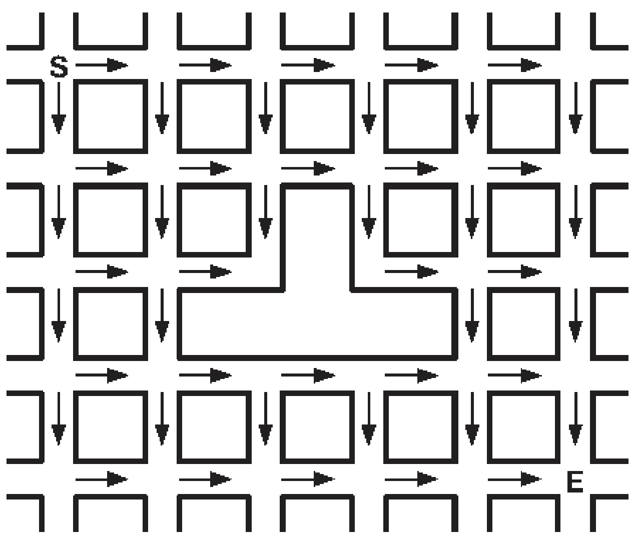

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

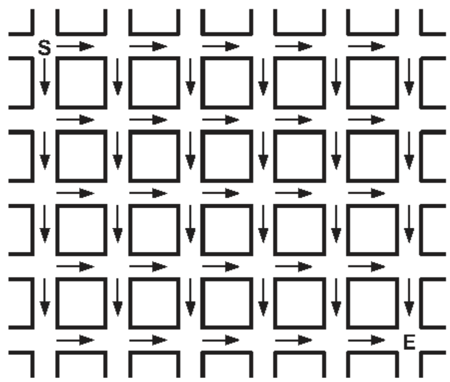

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

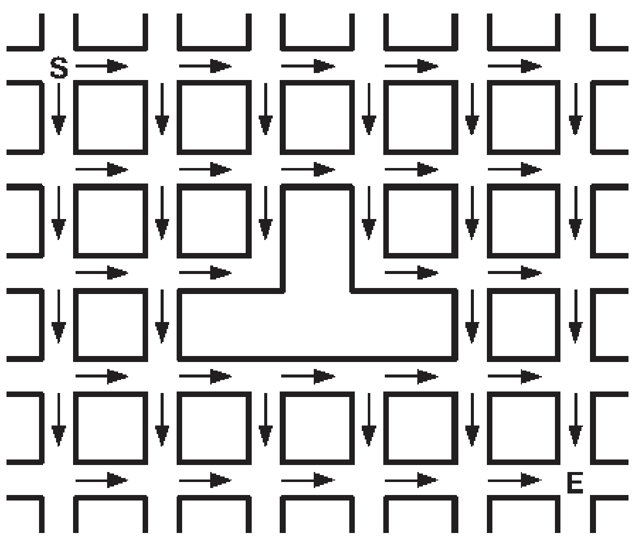

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

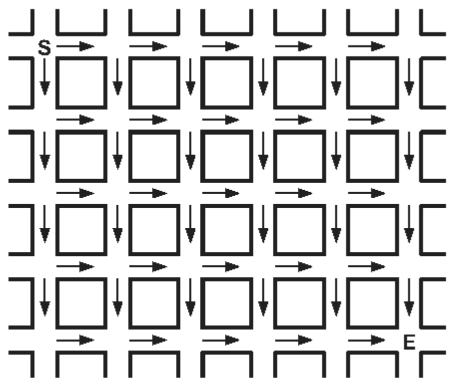

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

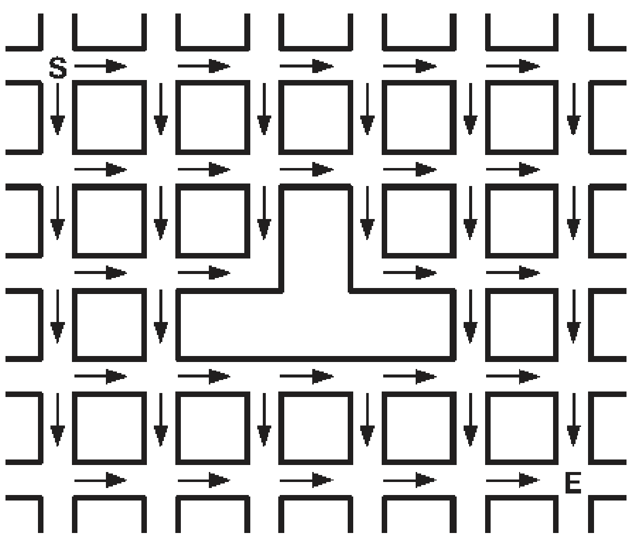

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

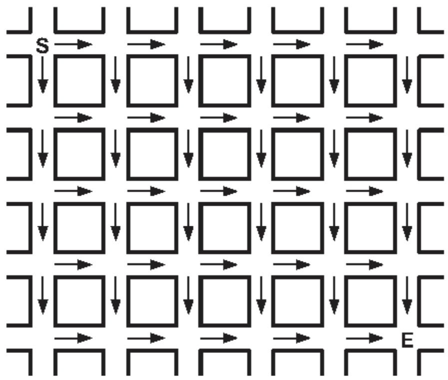

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

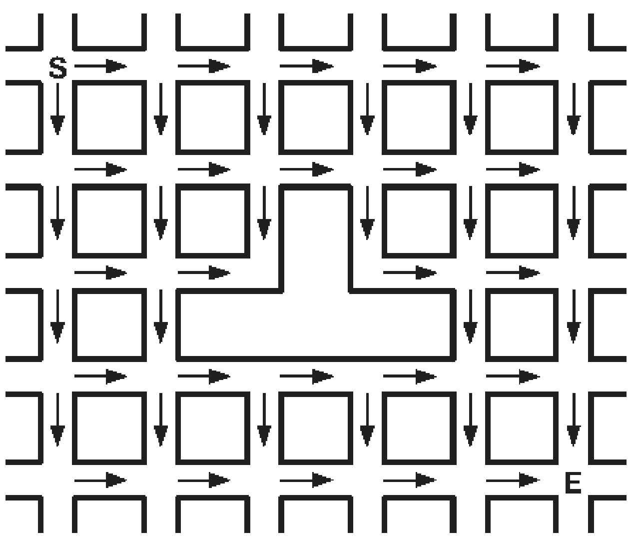

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

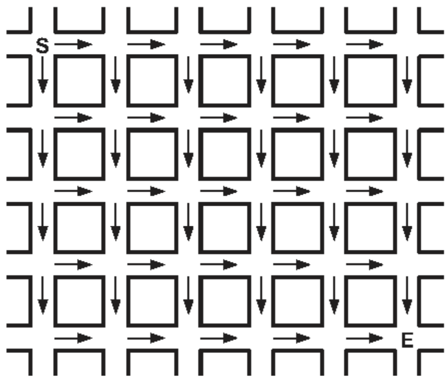

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

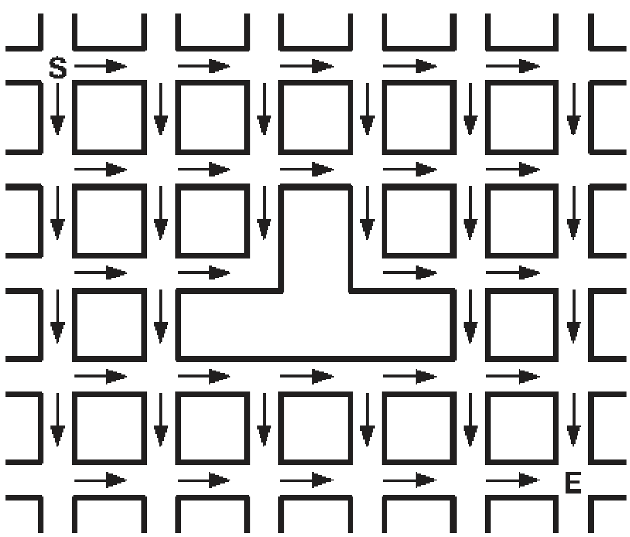

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

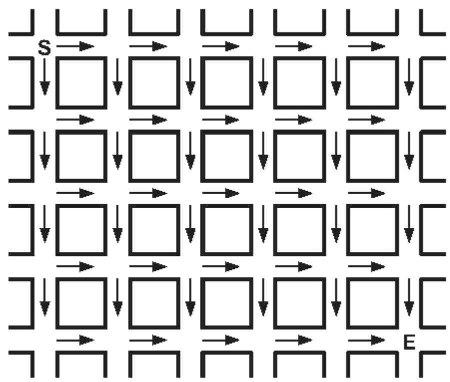

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

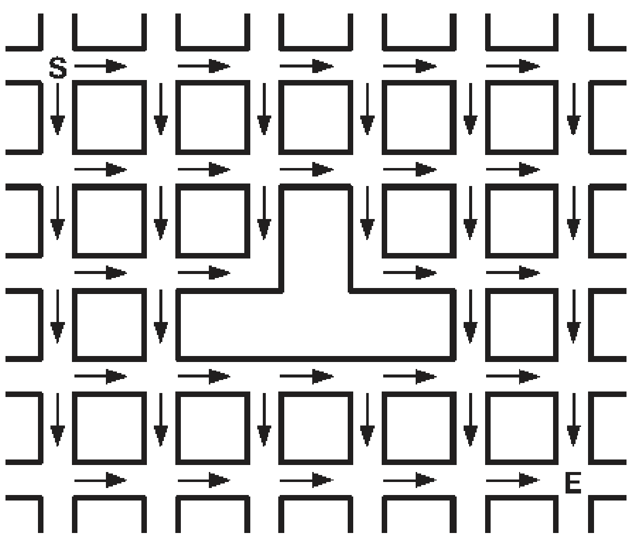

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

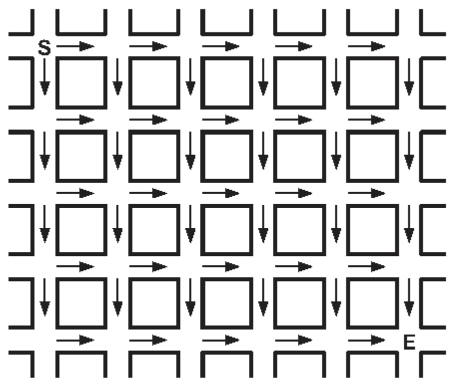

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

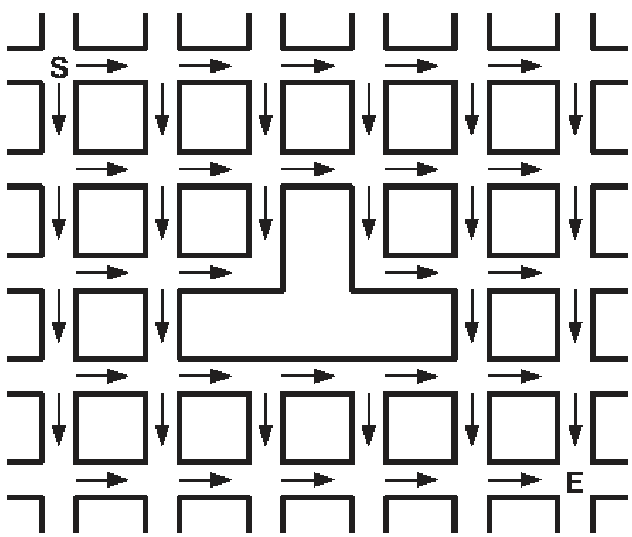

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

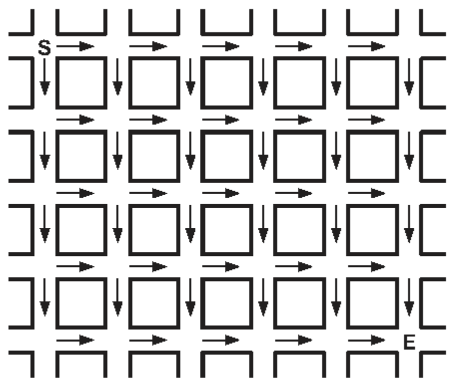

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

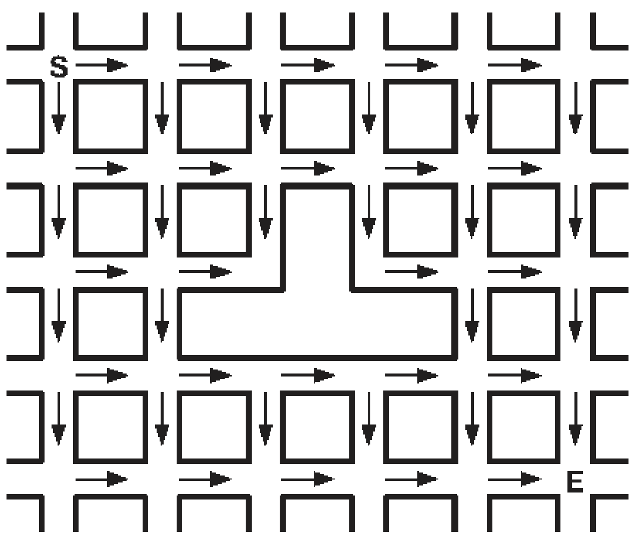

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

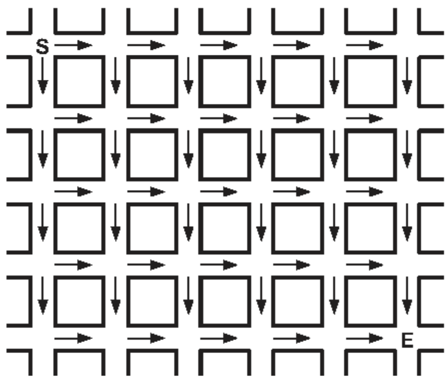

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

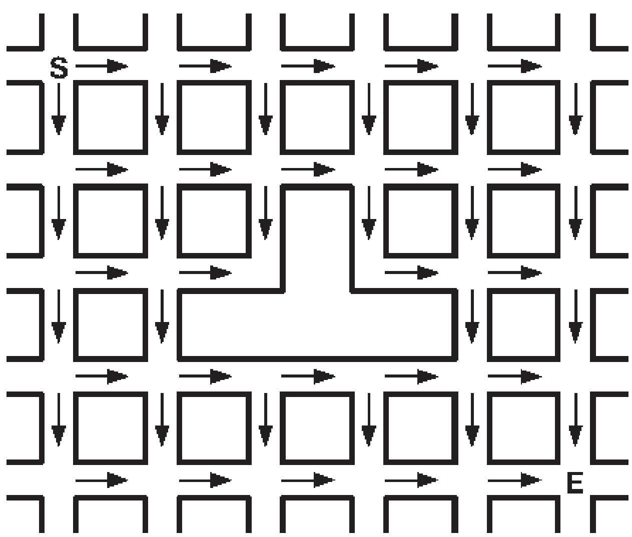

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

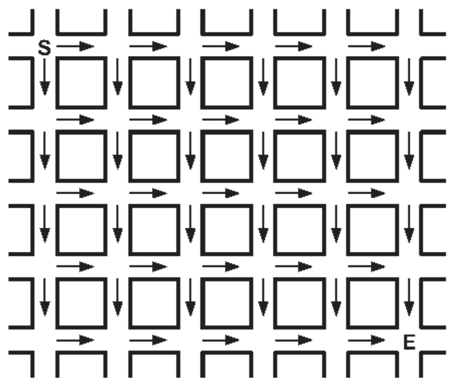

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

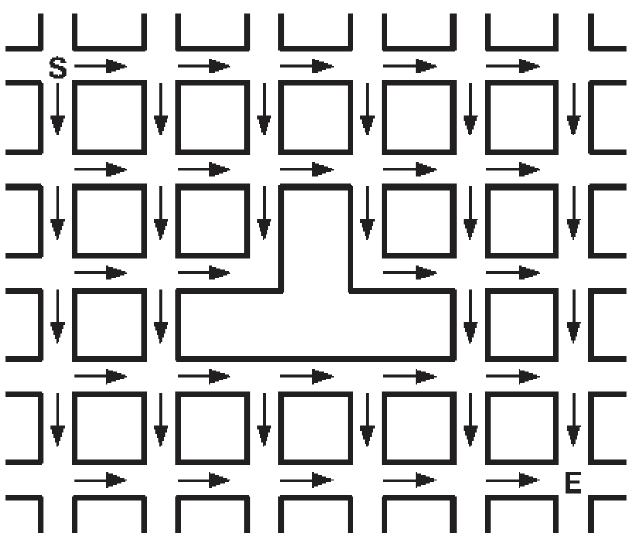

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

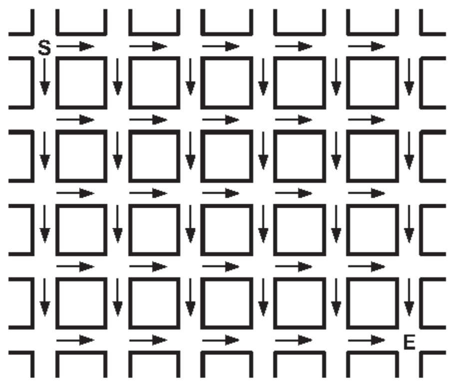

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

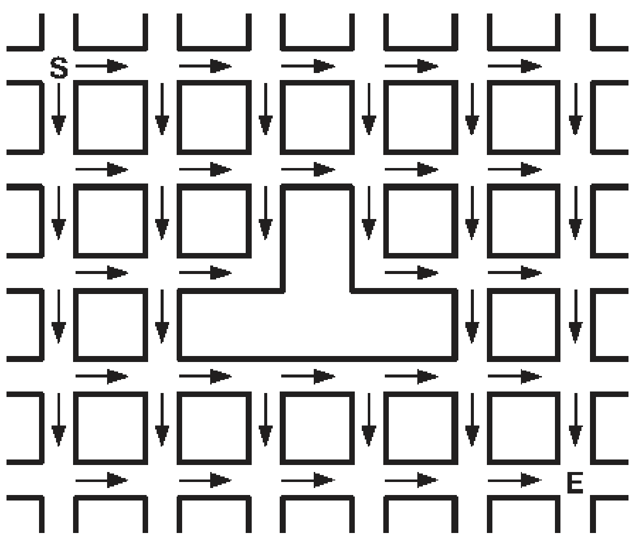

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

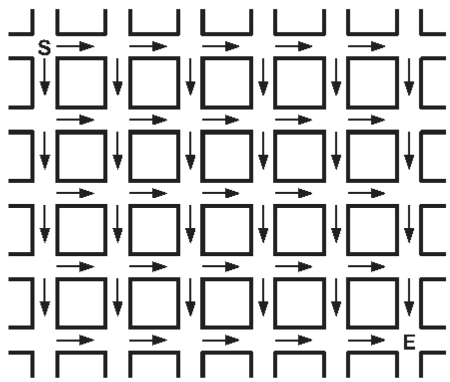

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

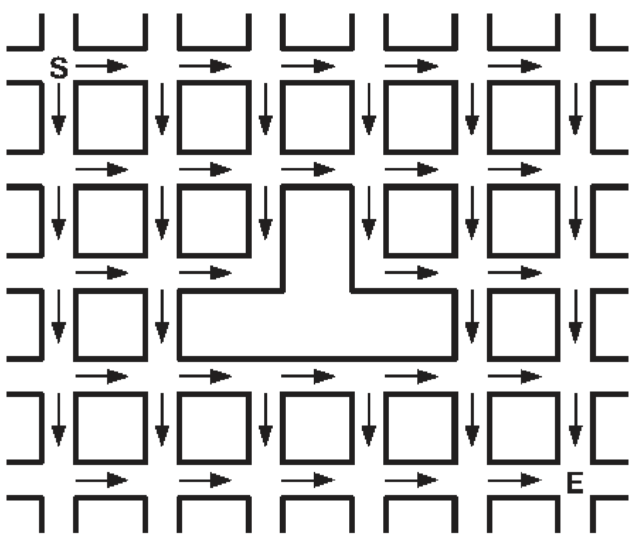

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

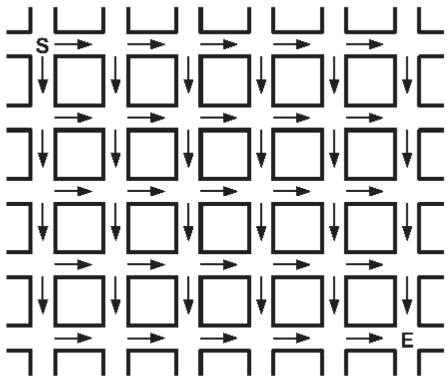

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

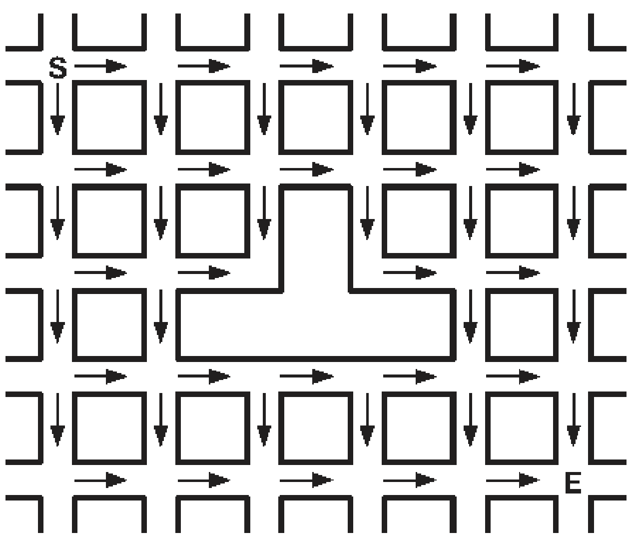

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

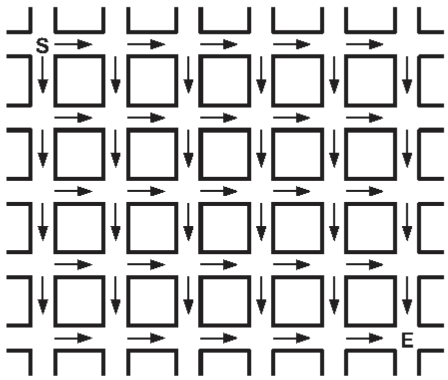

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

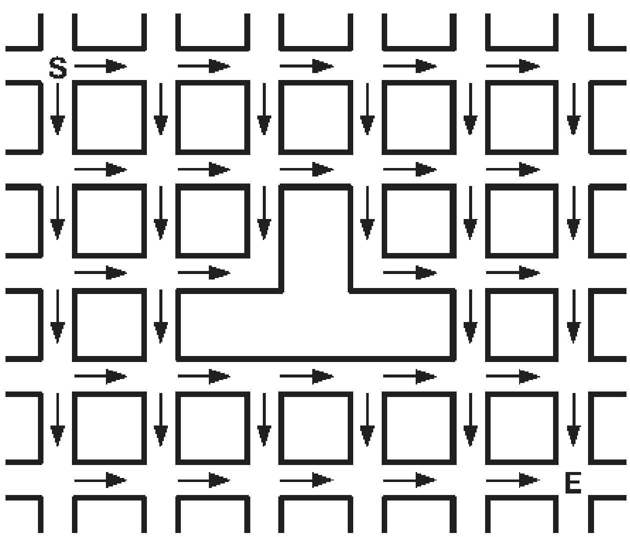

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

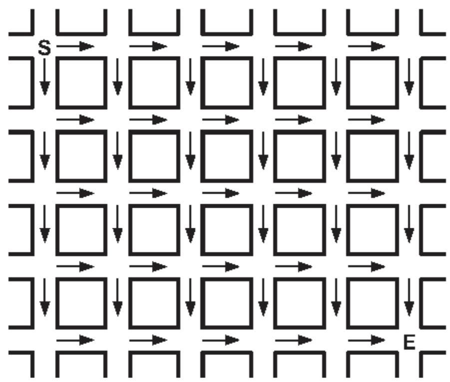

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

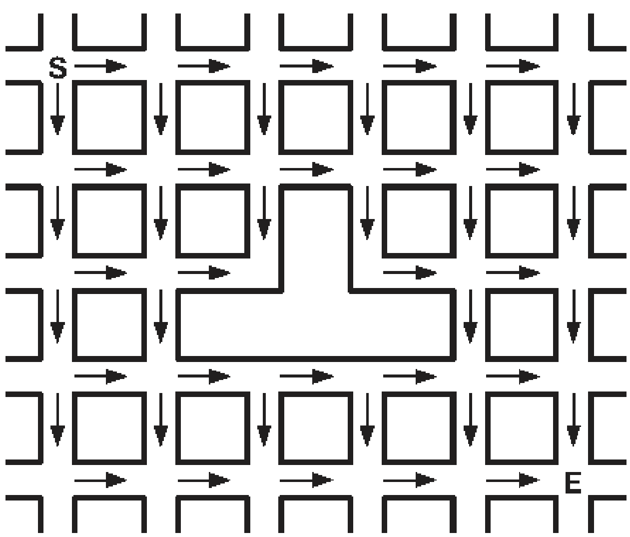

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

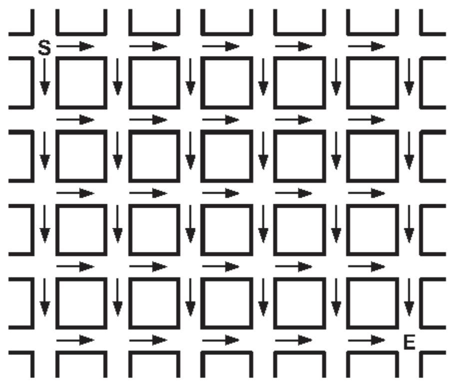

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

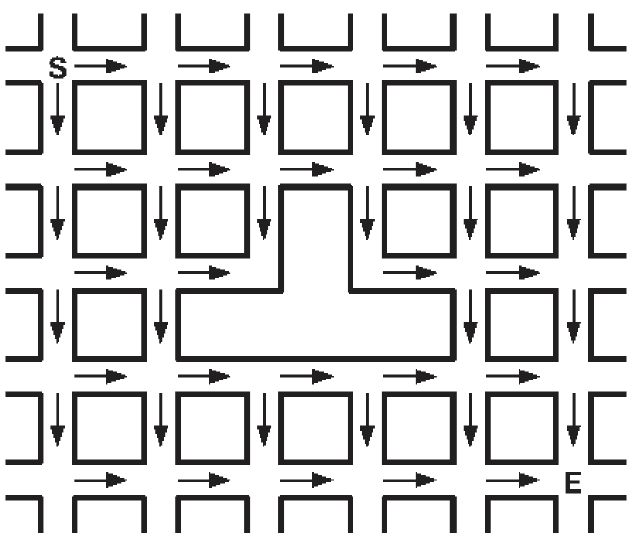

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

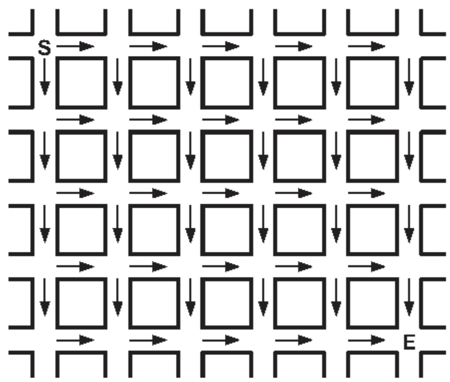

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

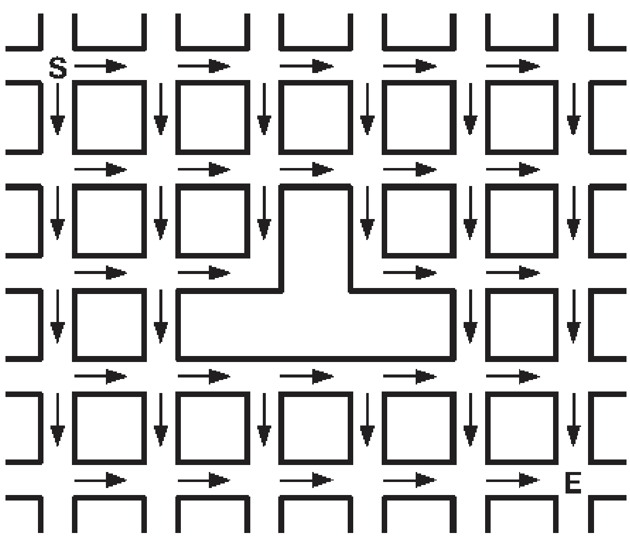

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

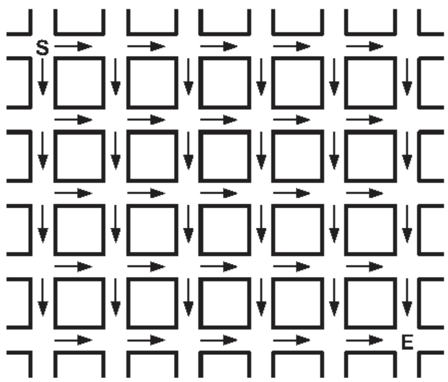

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

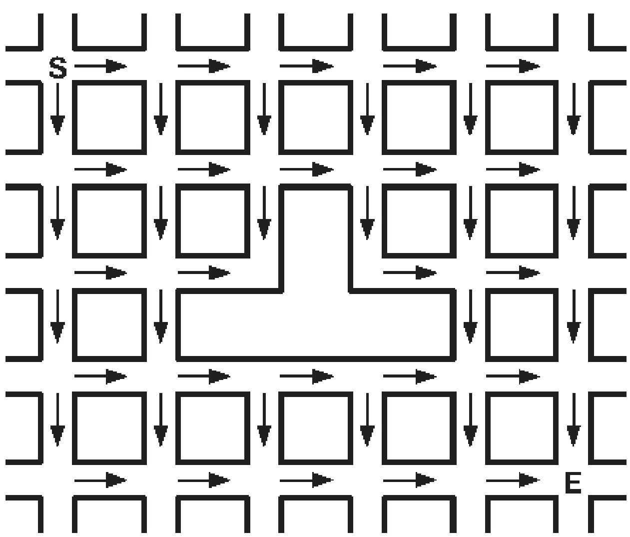

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

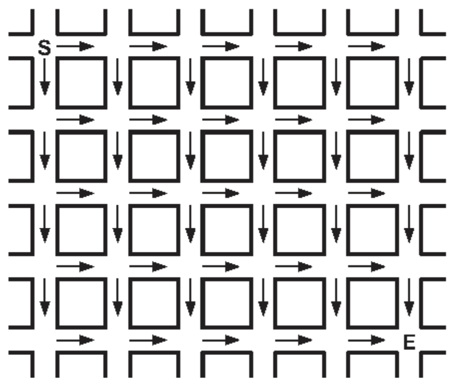

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

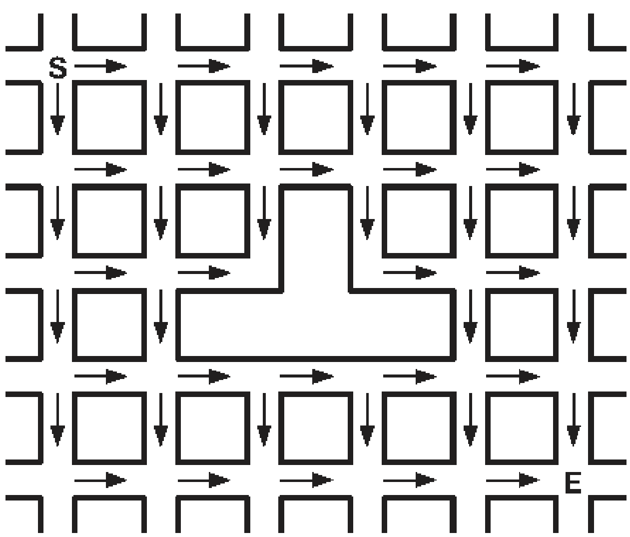

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ______________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

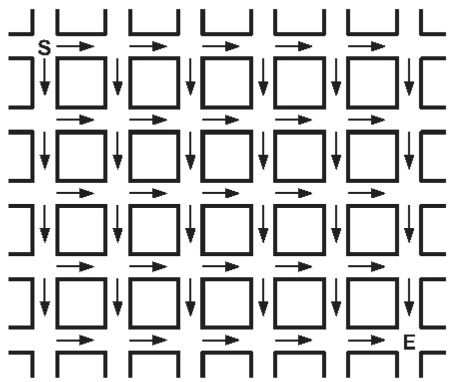

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

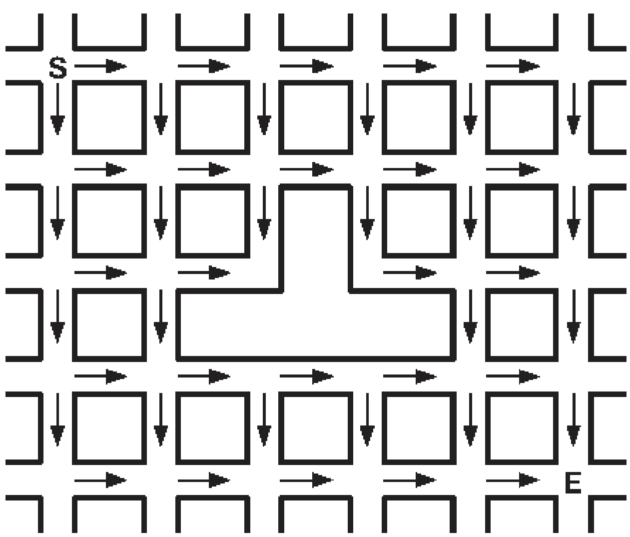

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

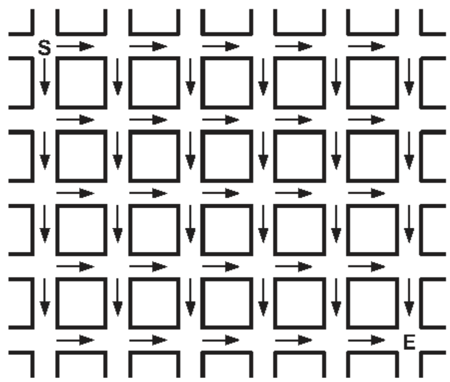

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

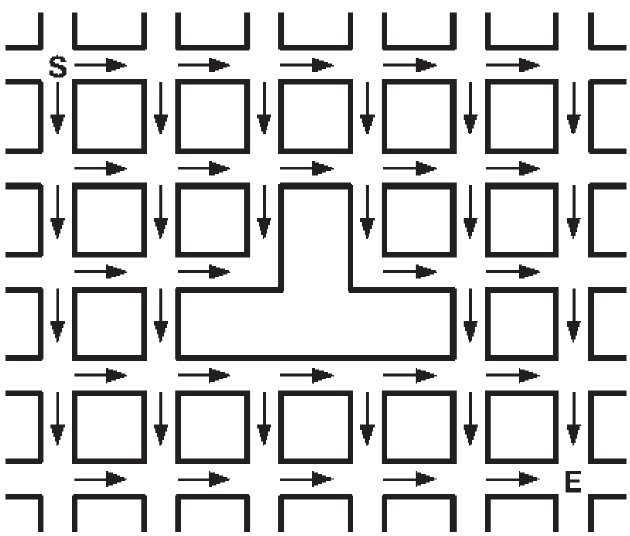

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

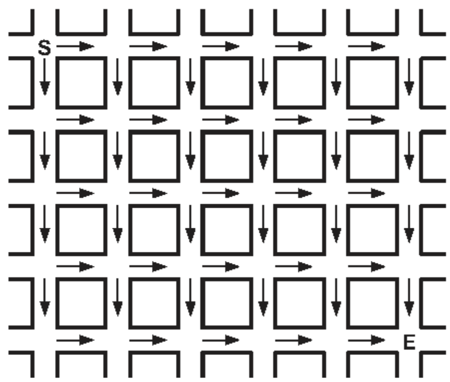

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

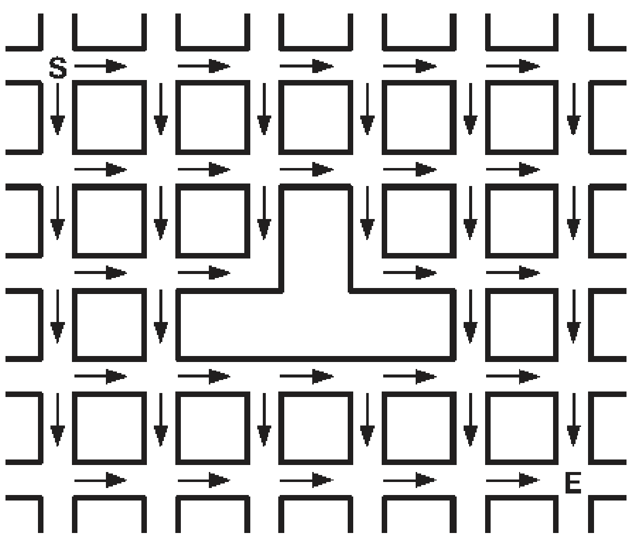

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \dots, p_n\}$ and the nights $\{d_1, d_2, \dots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \dots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

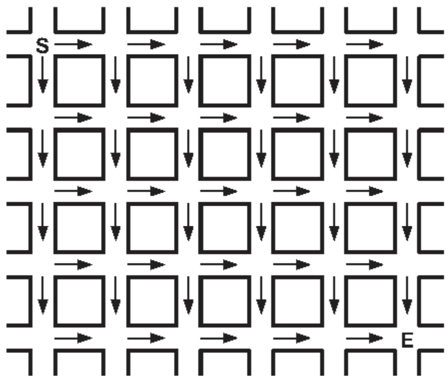

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

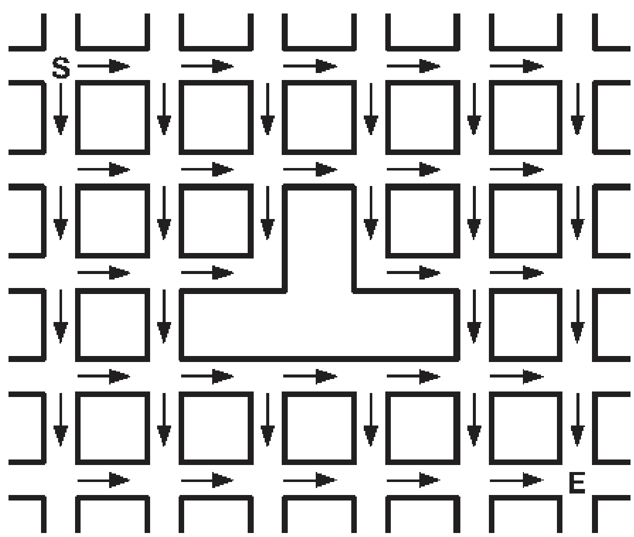

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

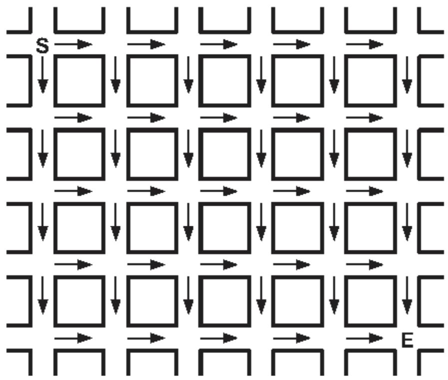

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

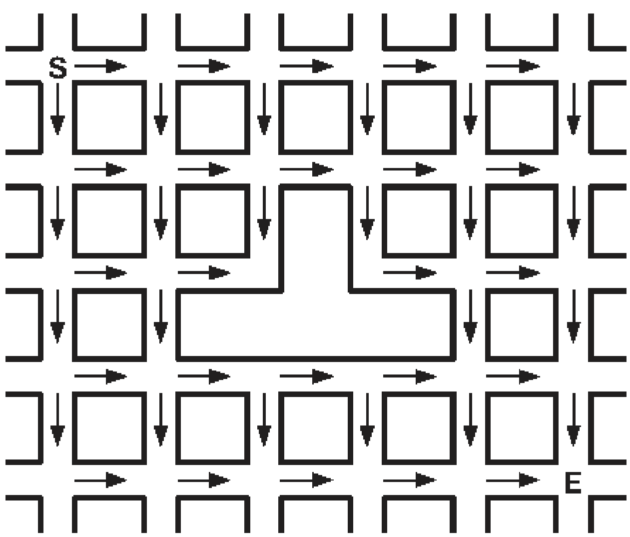

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

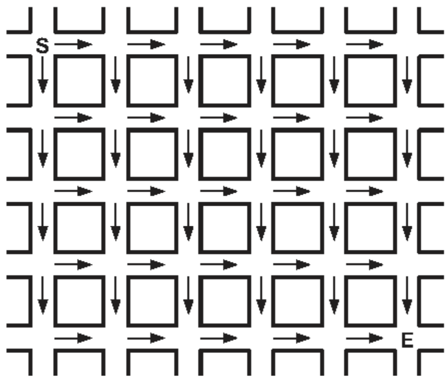

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

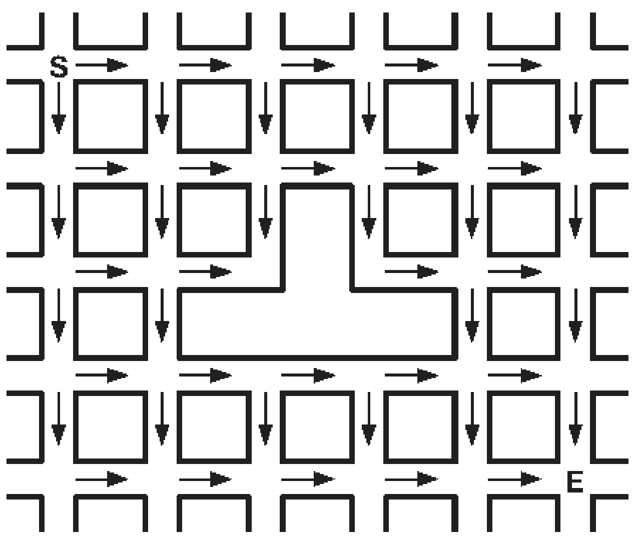

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

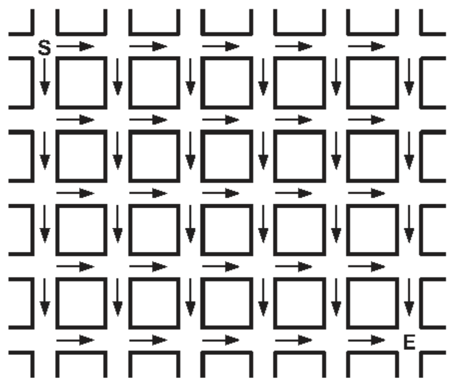

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

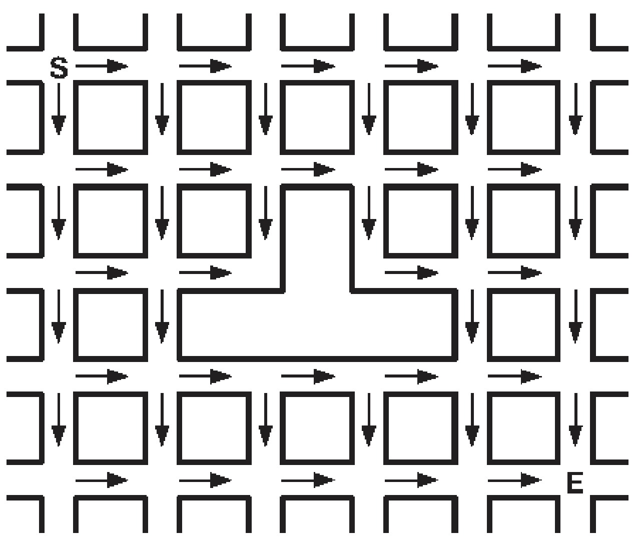

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

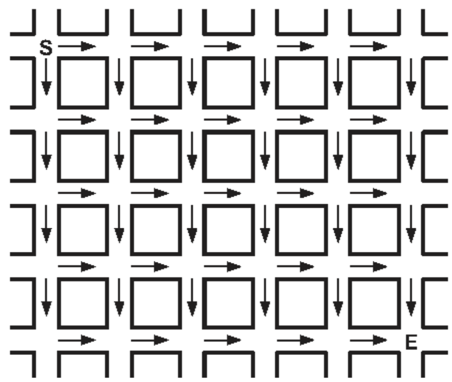

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

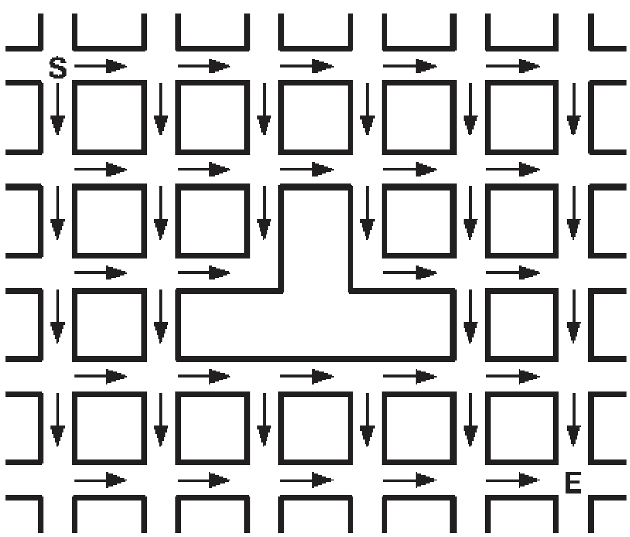

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

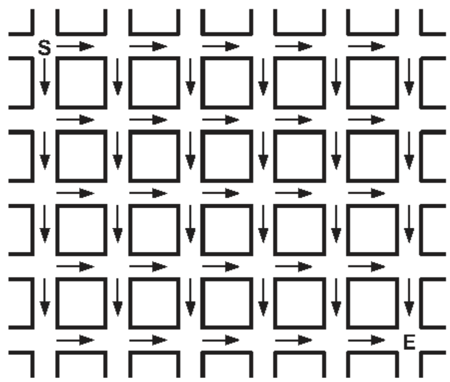

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

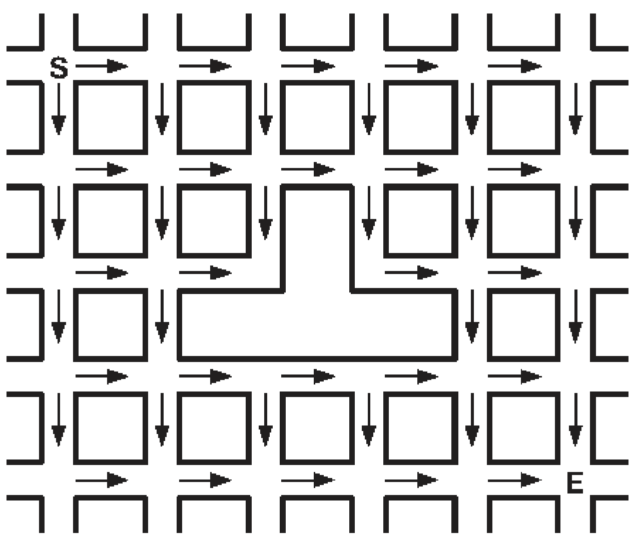

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

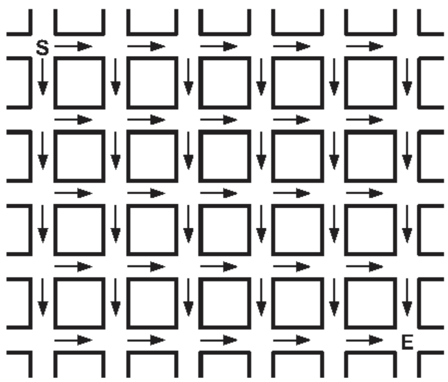

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

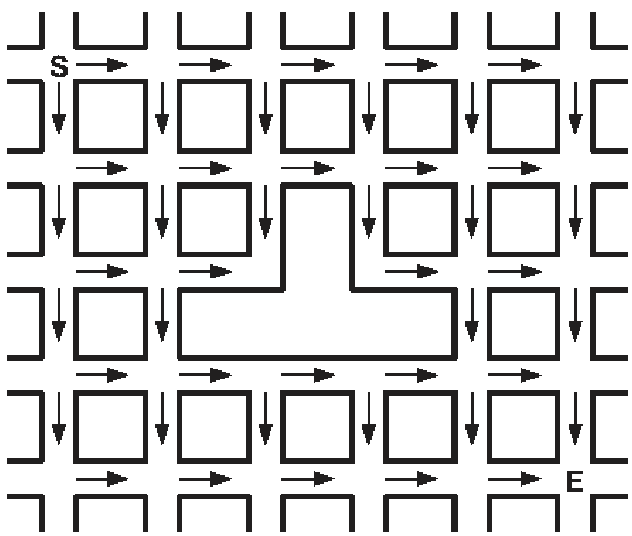

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

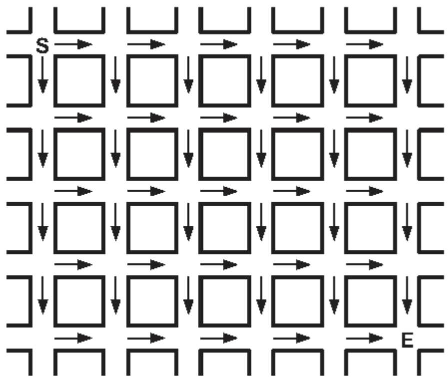

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

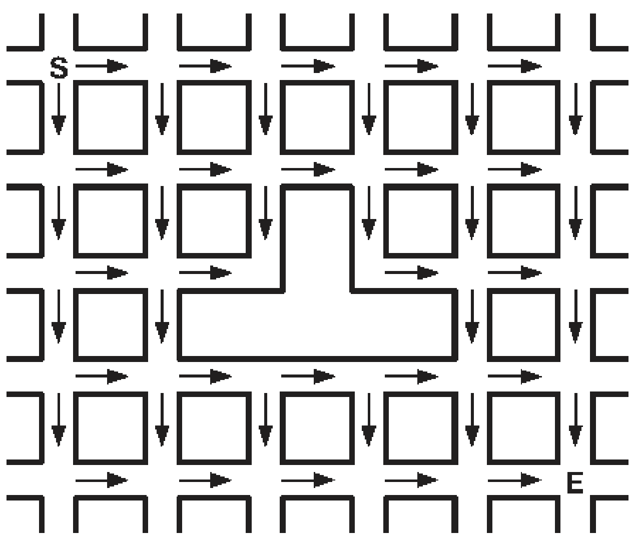

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

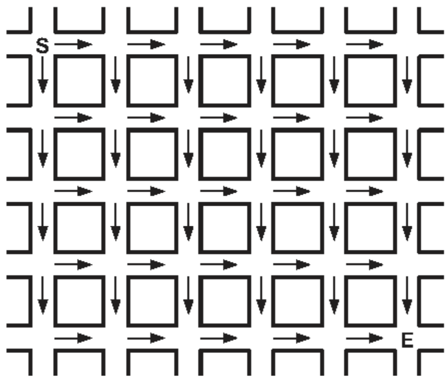

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

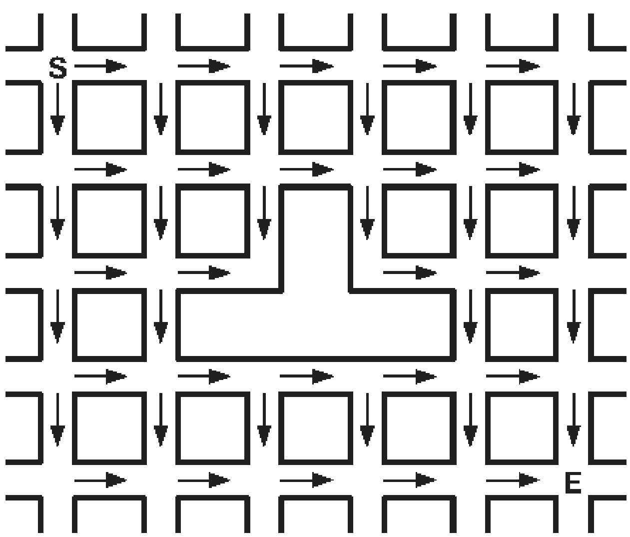

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

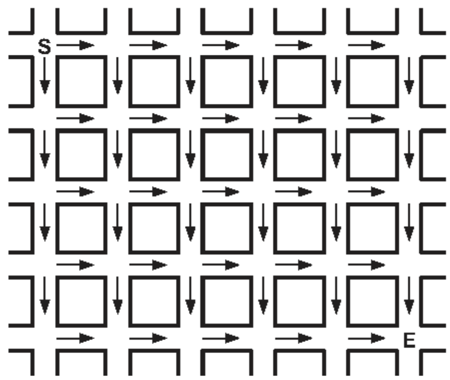

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

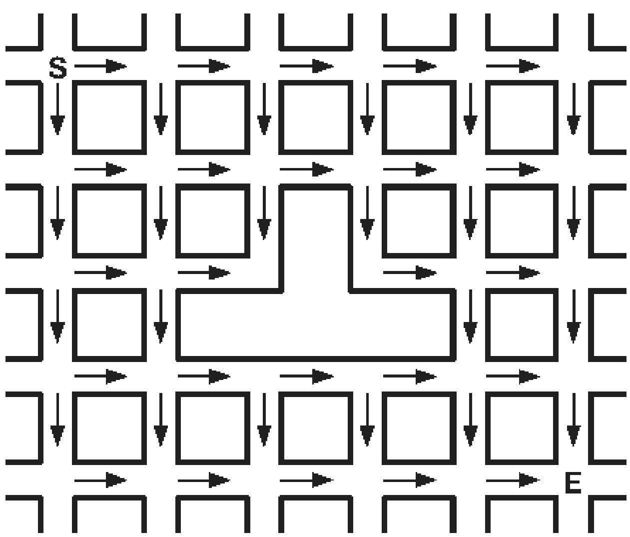

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

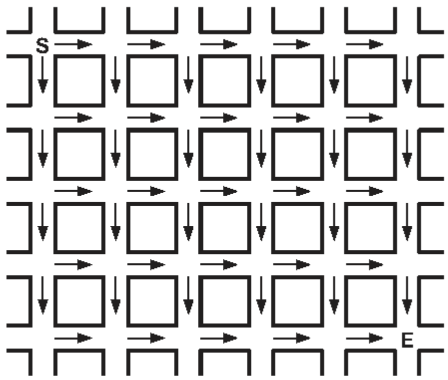

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

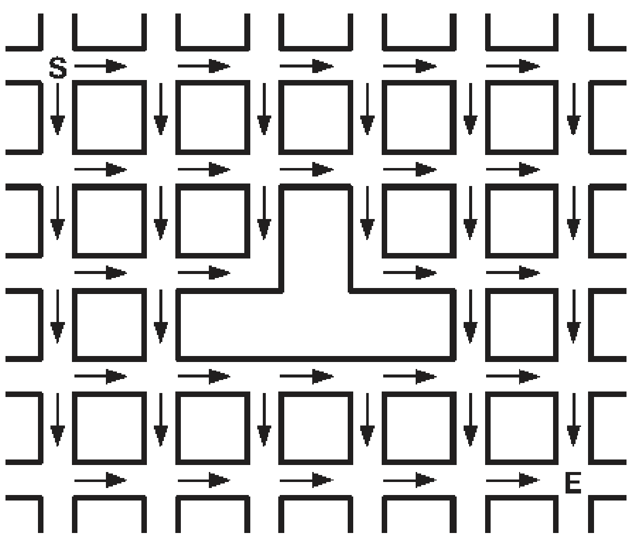

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

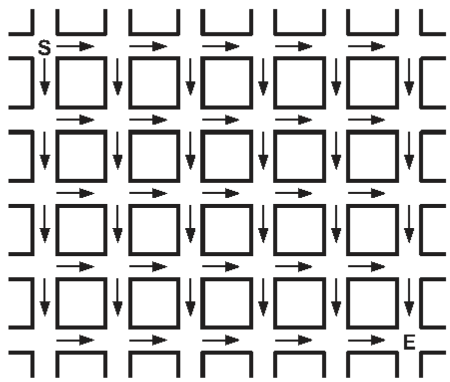

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

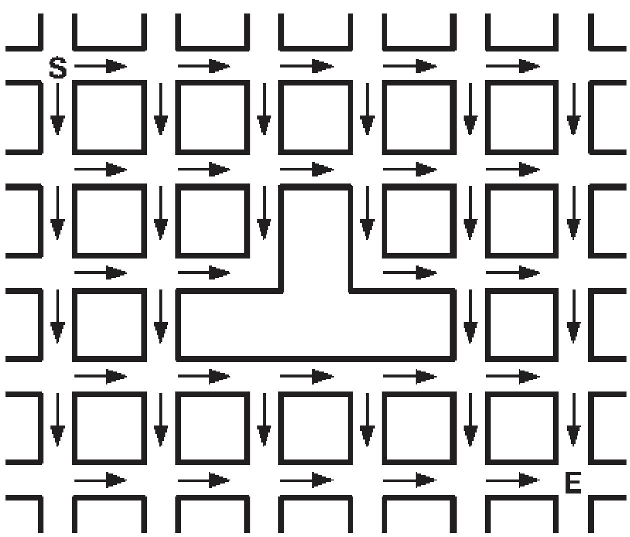

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

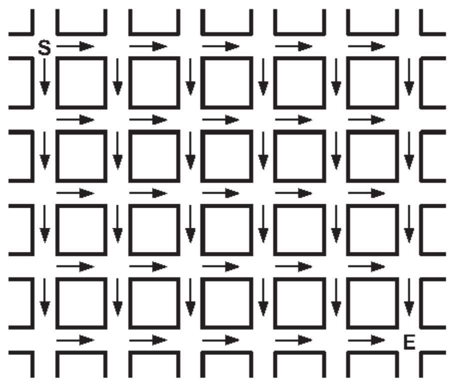

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

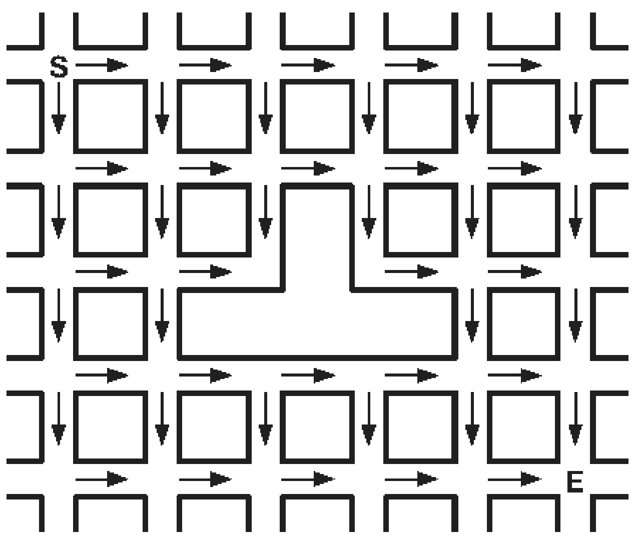

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts
Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.
Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

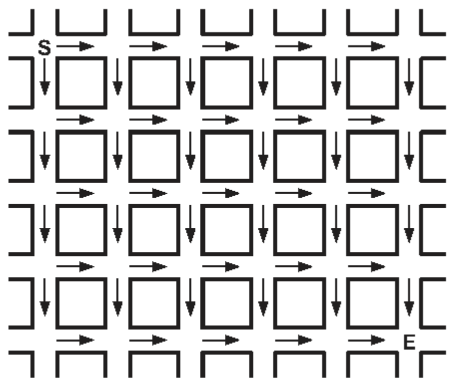

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

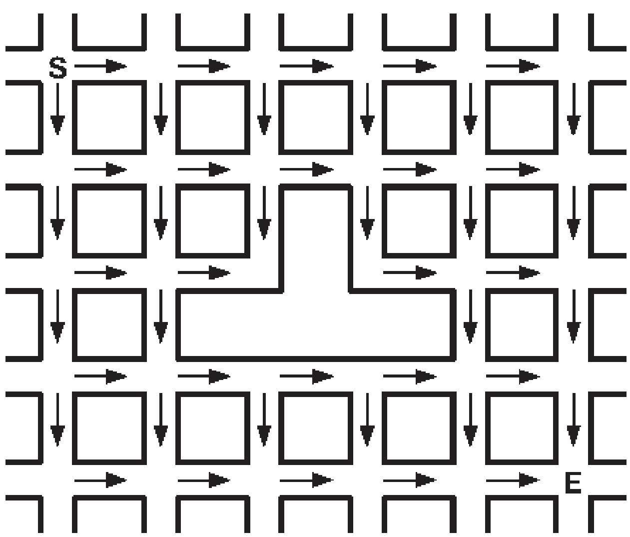

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

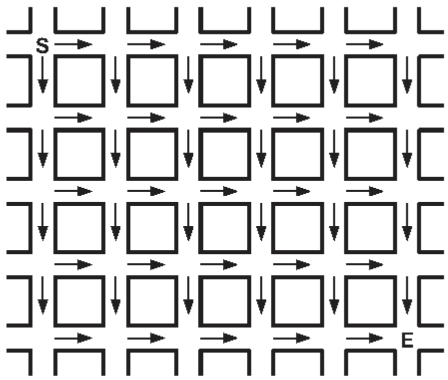

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

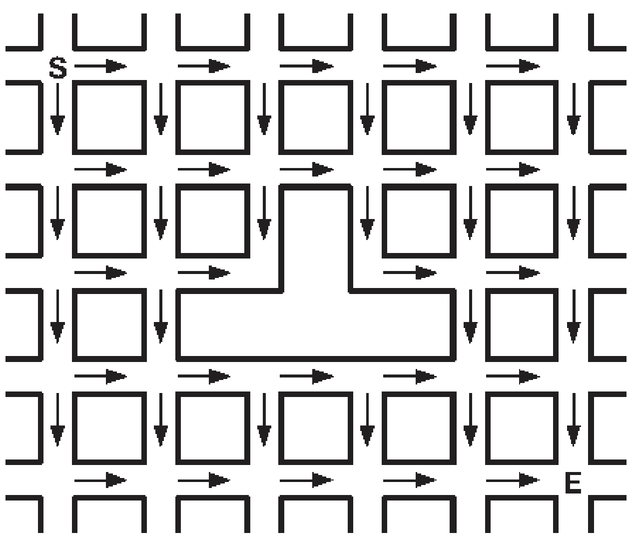

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

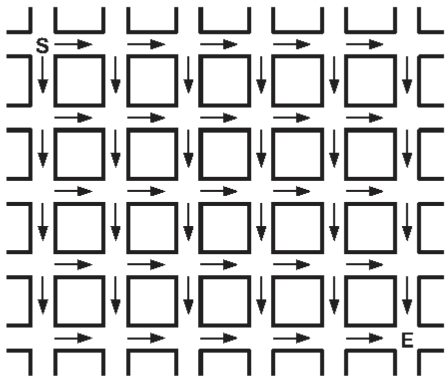

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

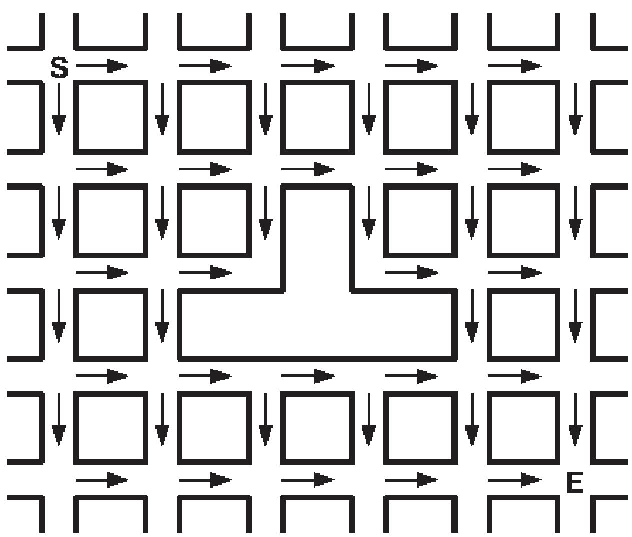

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

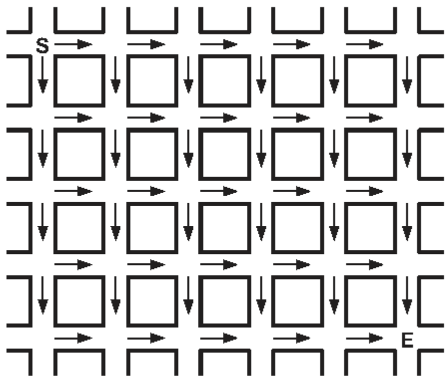

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

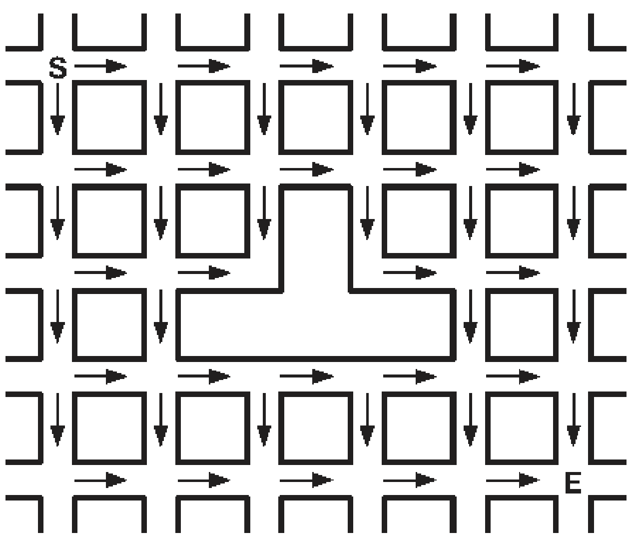

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

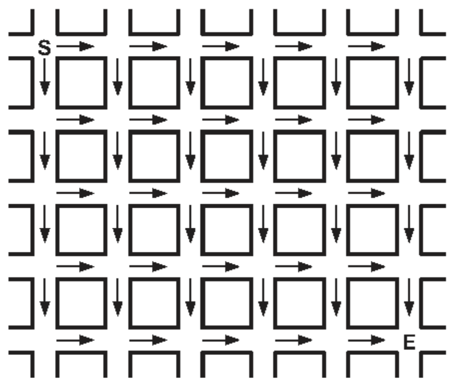

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

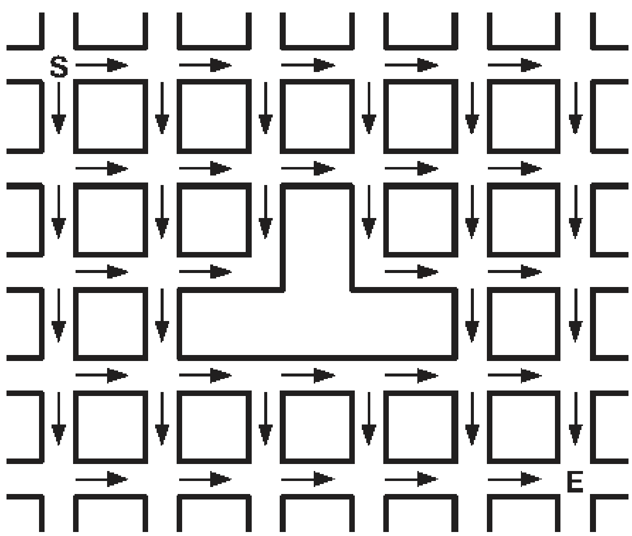

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

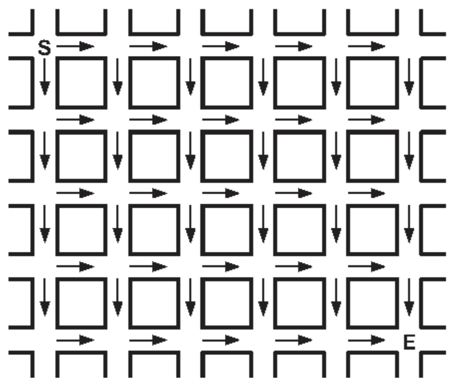

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

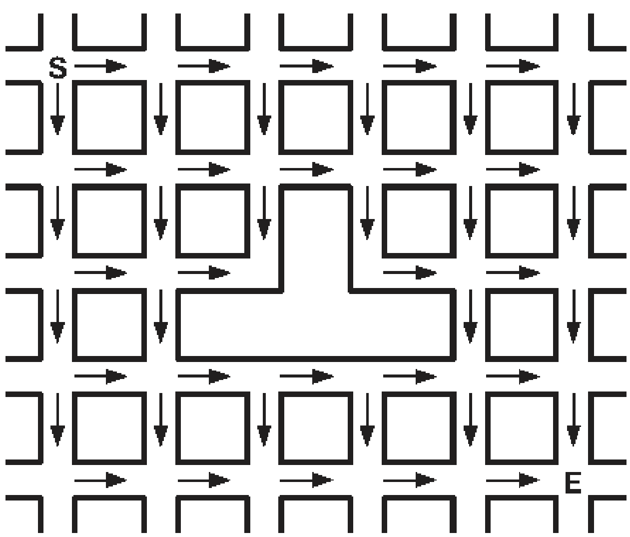

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

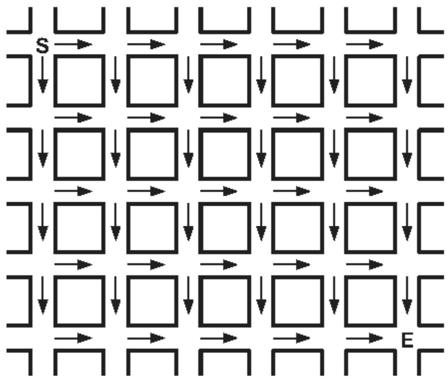

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

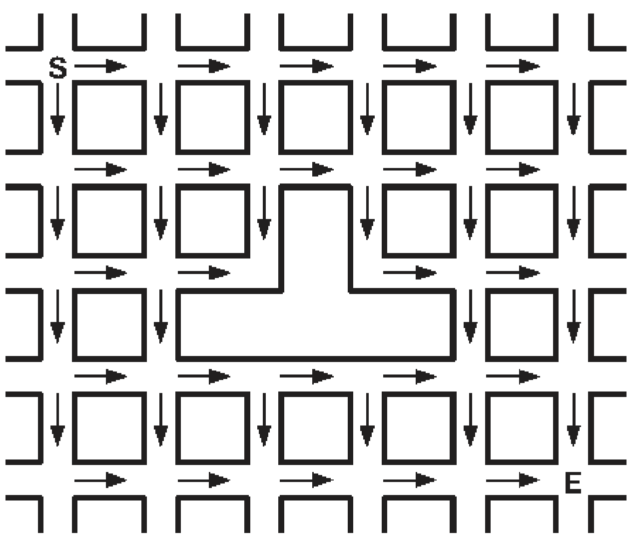

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

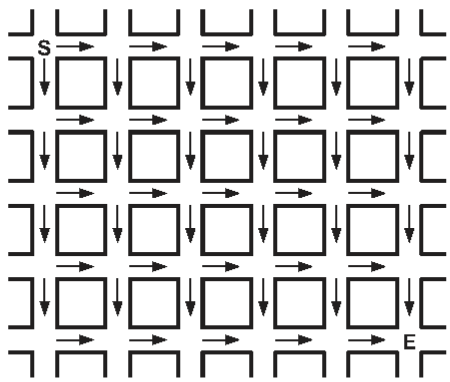

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

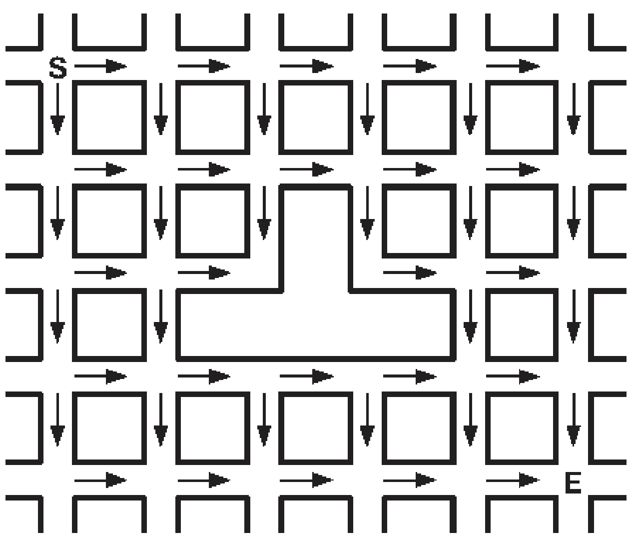

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

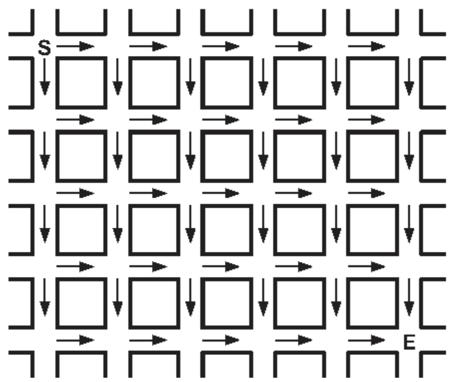

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

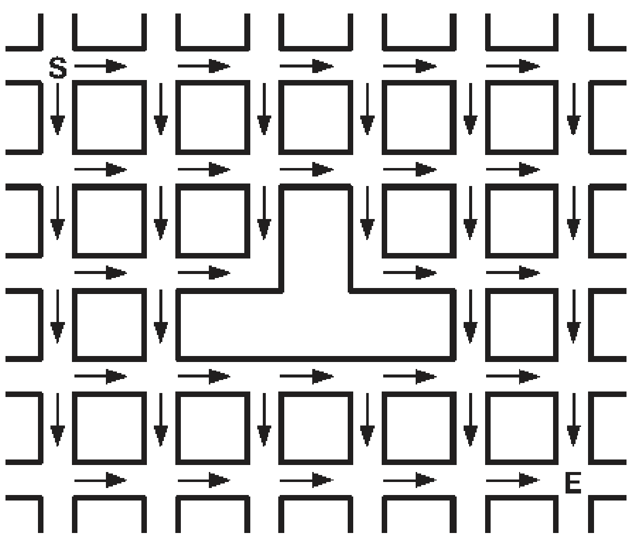

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

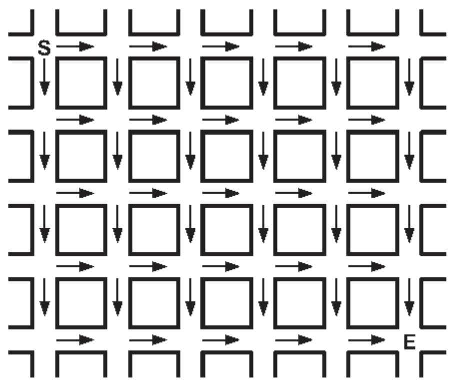

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

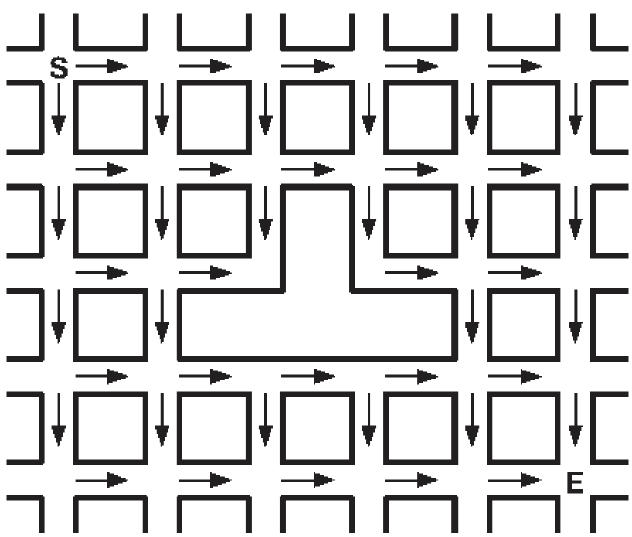

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

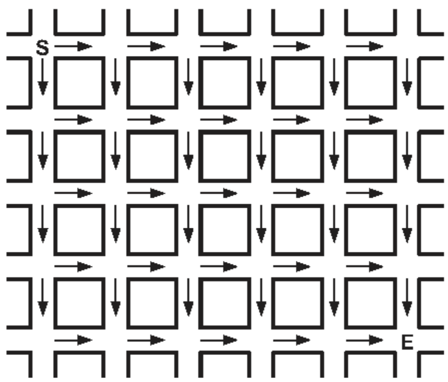

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

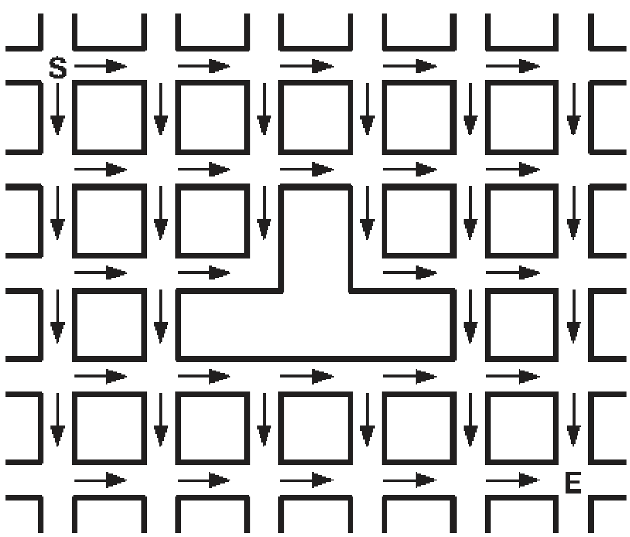

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

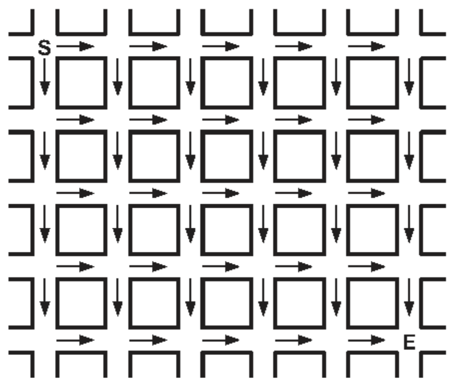

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

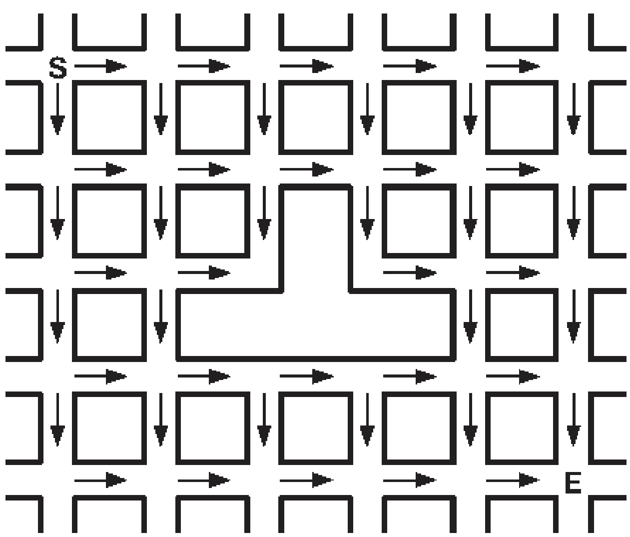

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

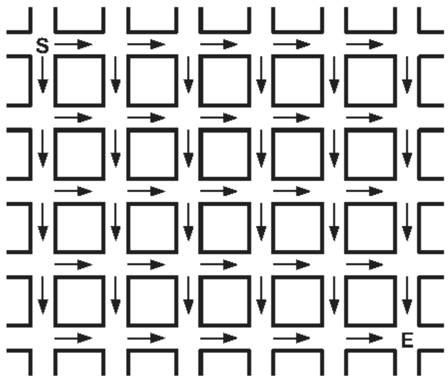

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

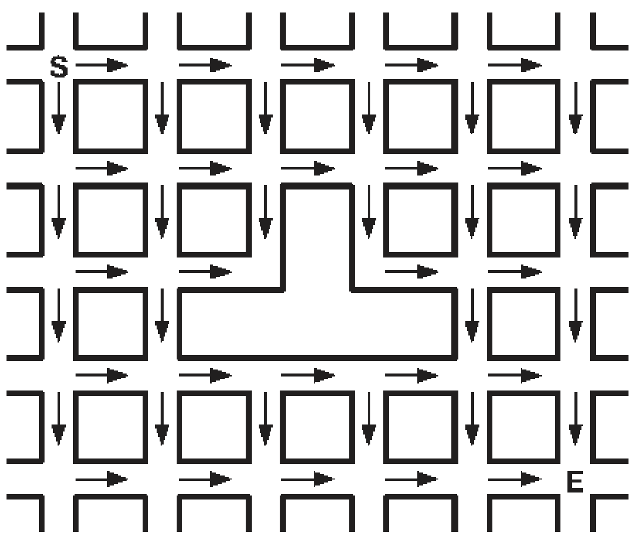

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

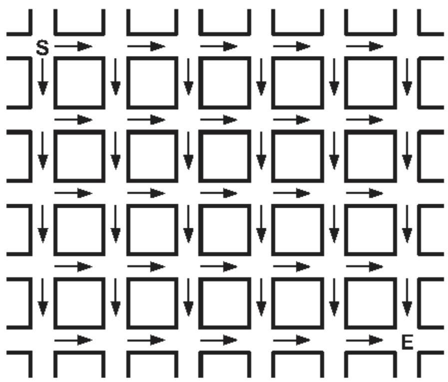

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

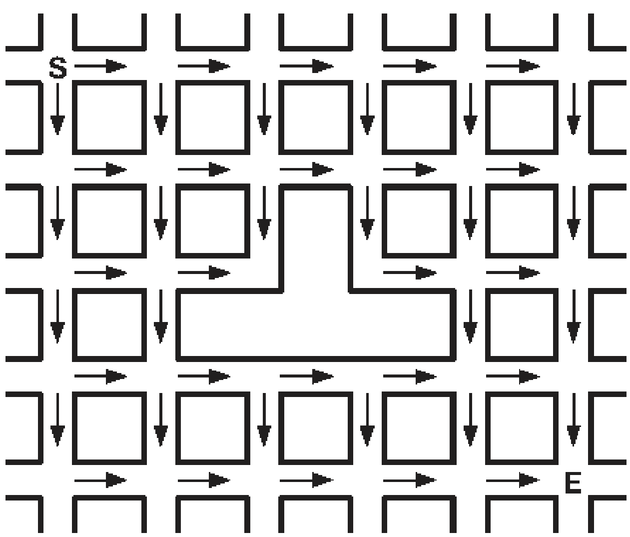

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

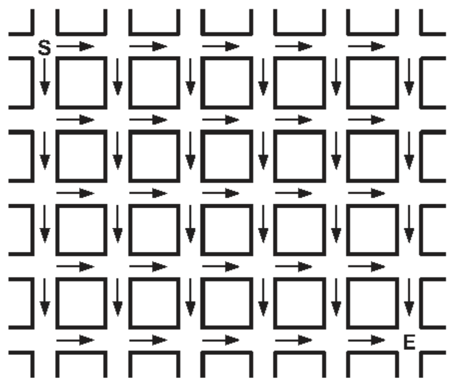

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

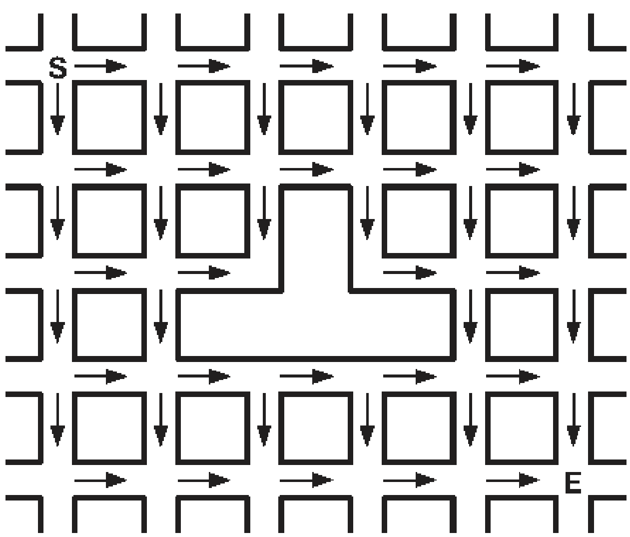

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

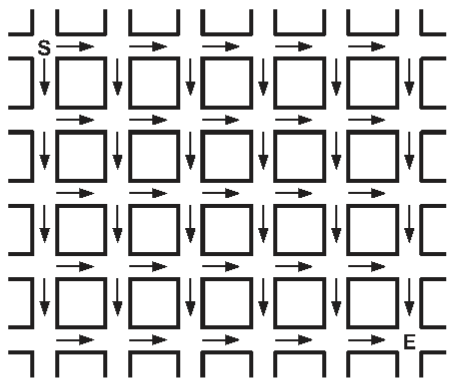

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

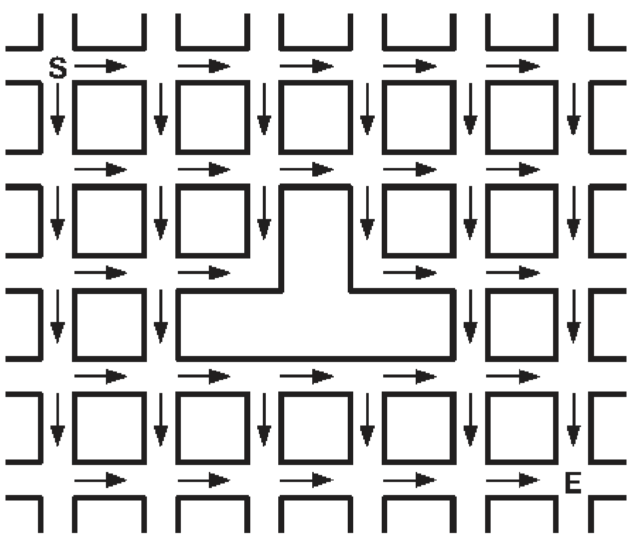

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

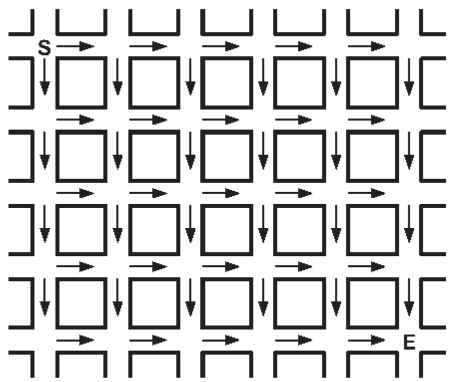

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

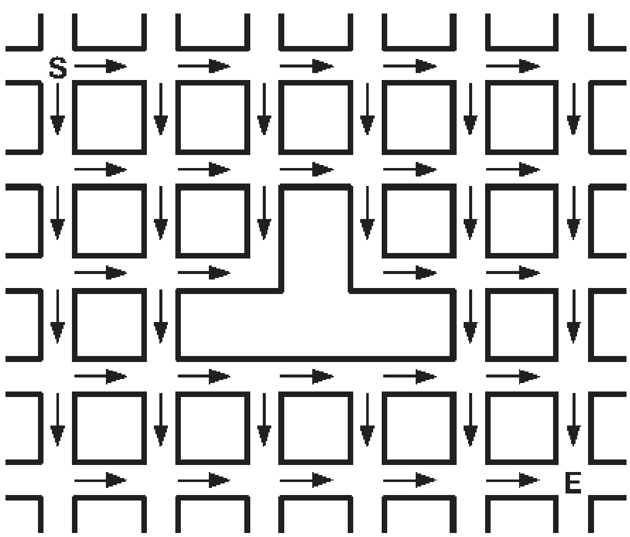

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

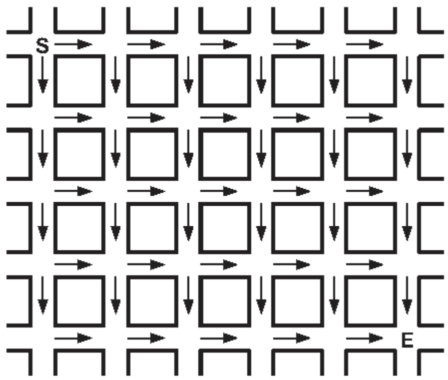

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

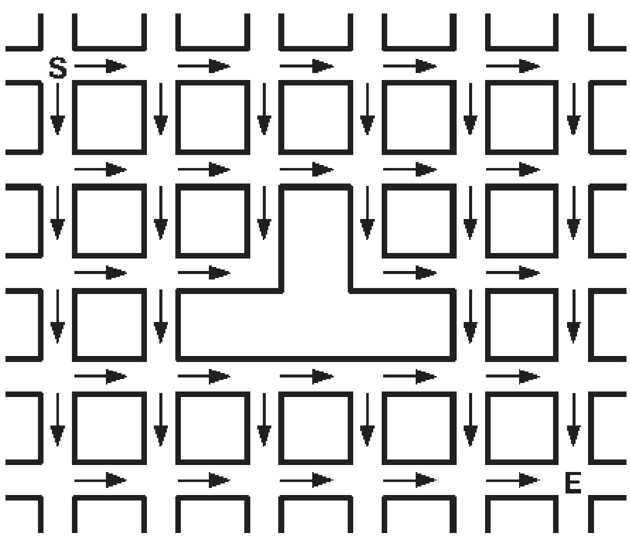

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

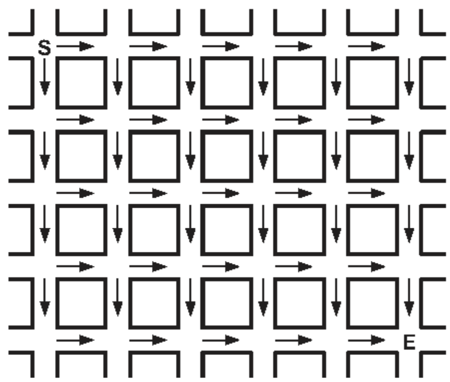

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

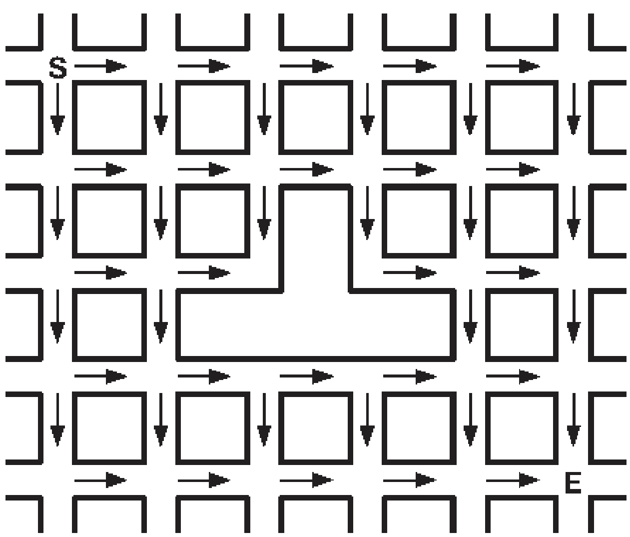

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

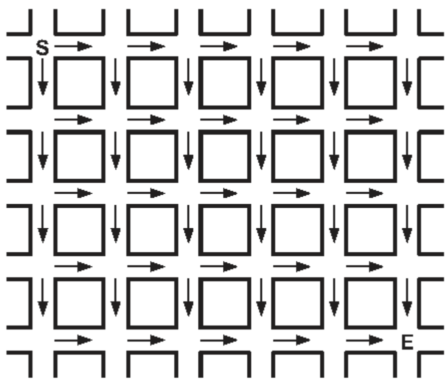

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

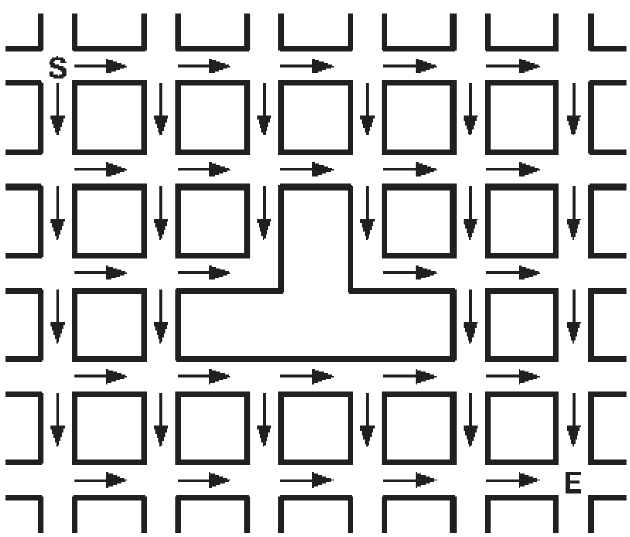

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

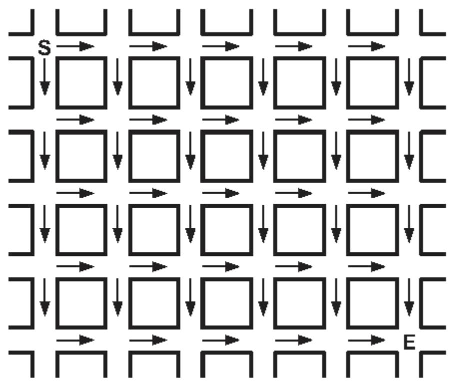

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

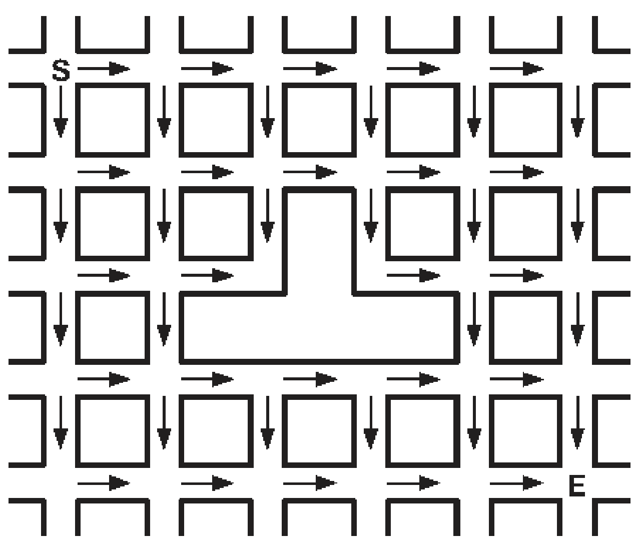

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

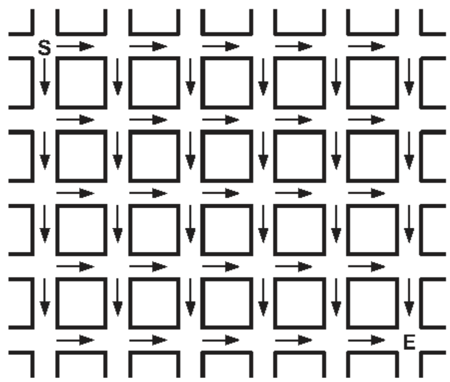

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

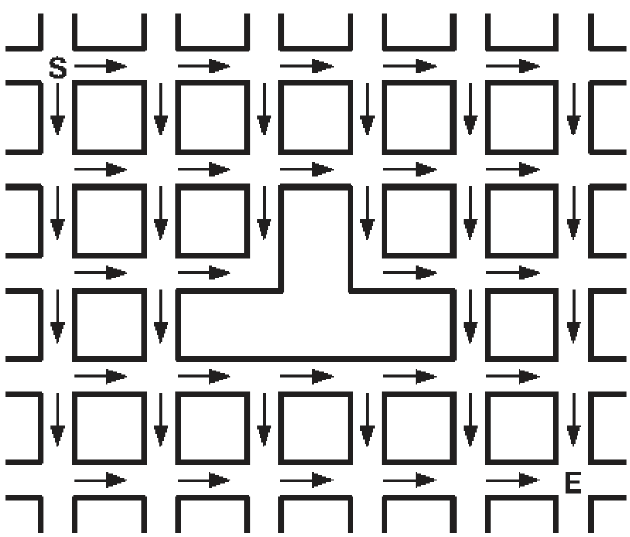

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

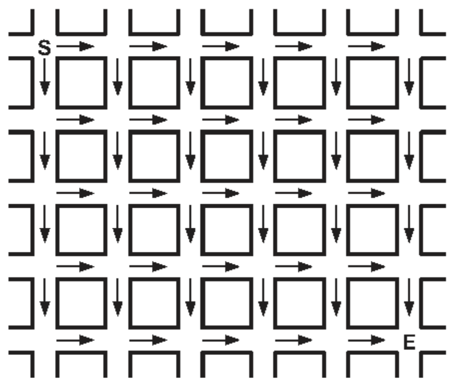

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

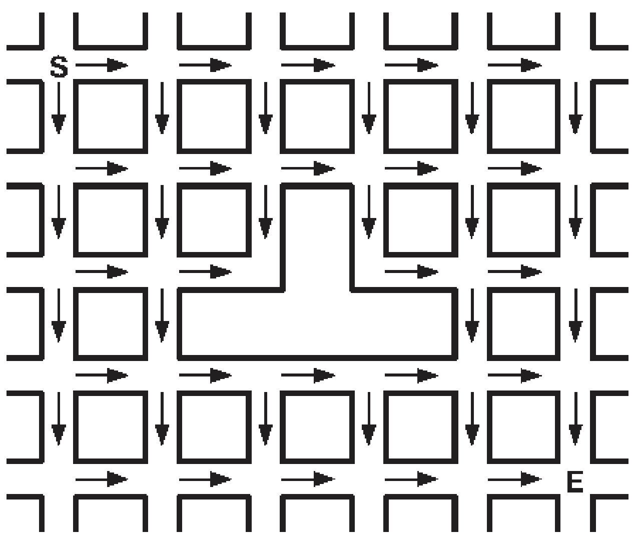

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

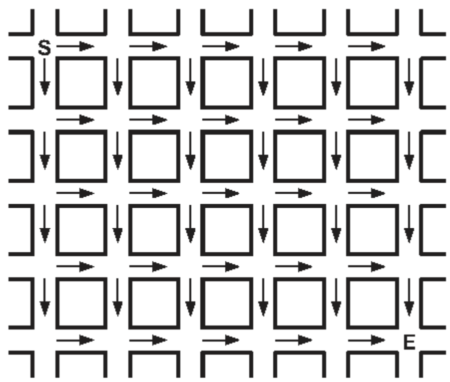

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

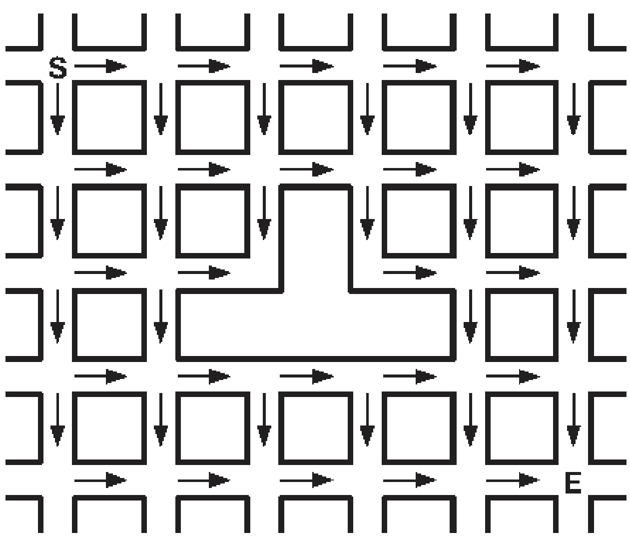

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

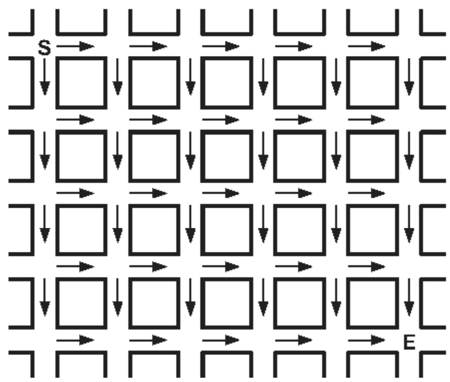

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

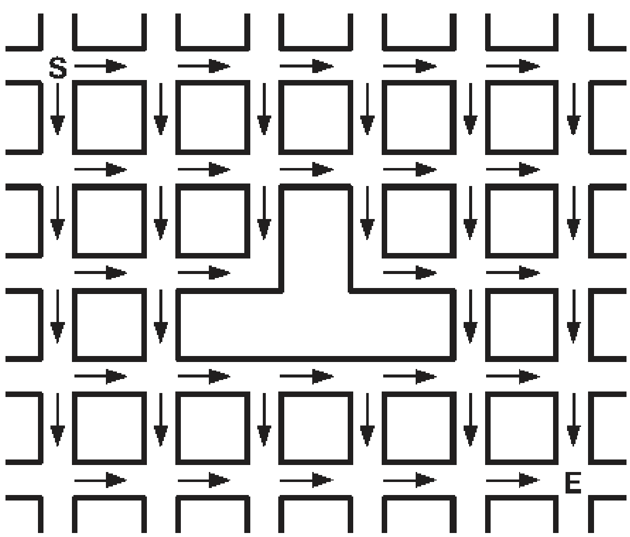

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

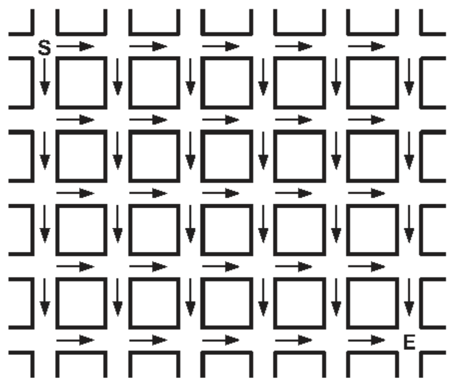

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

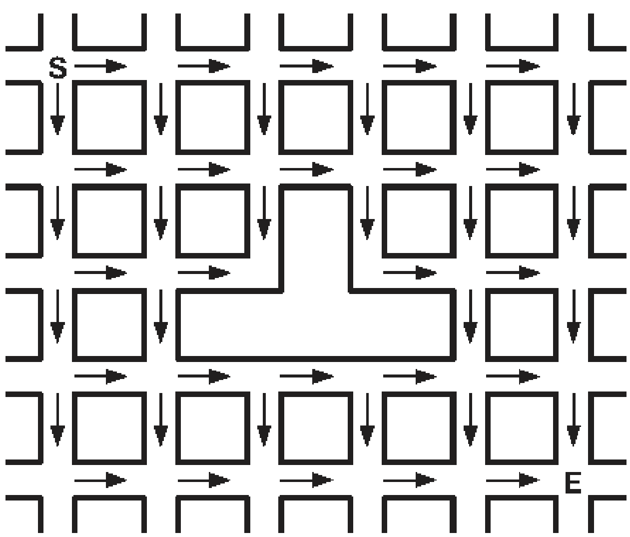

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

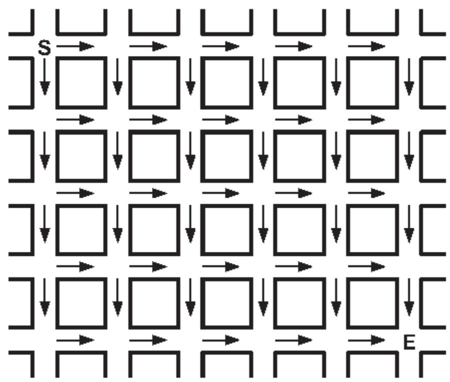

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

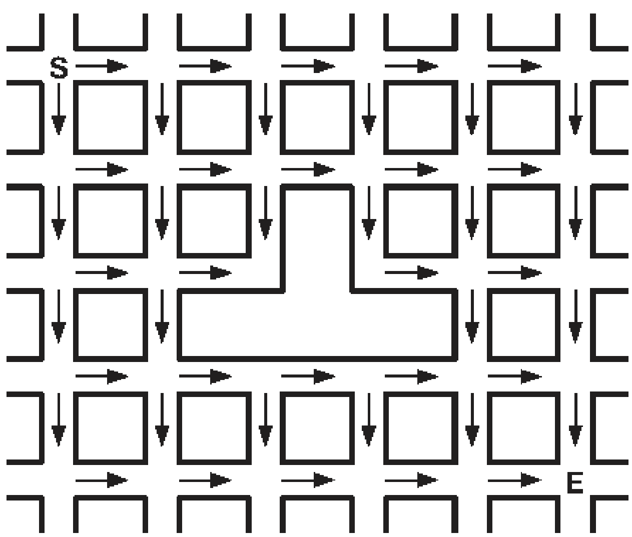

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

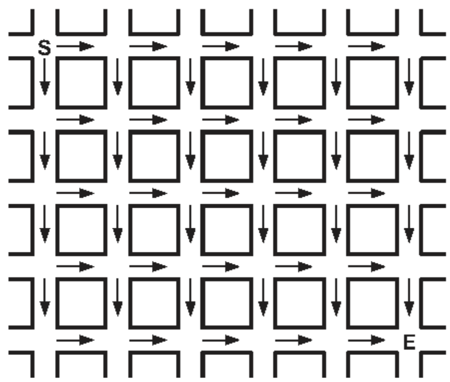

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

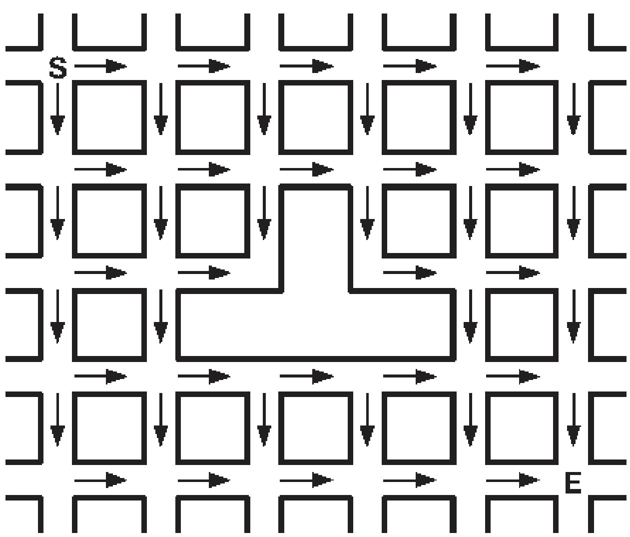

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

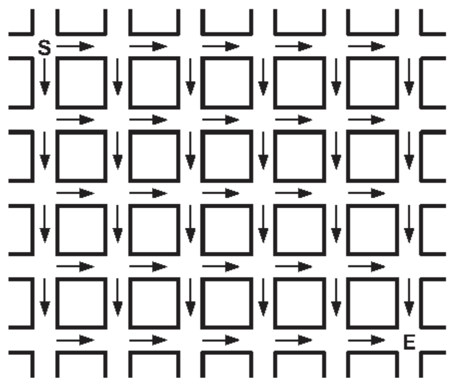

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

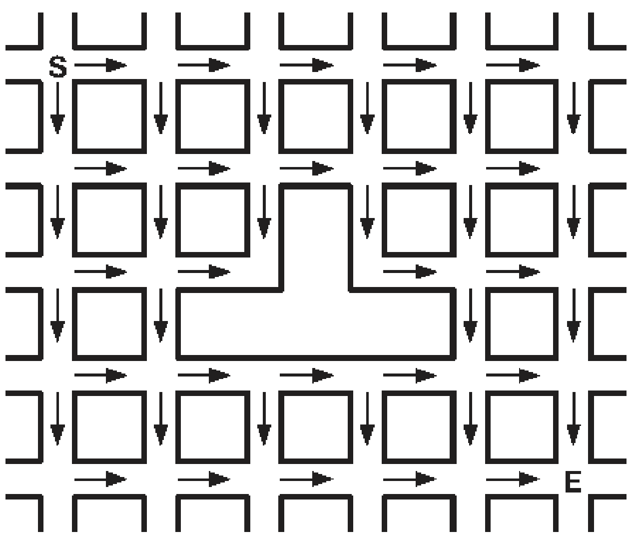

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

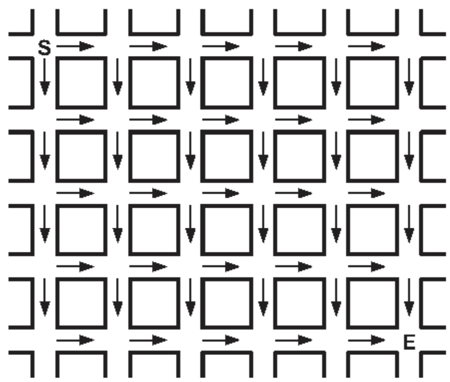

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

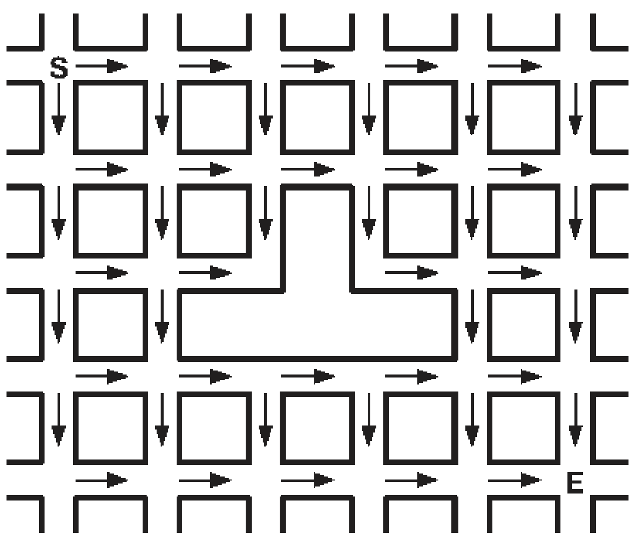

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

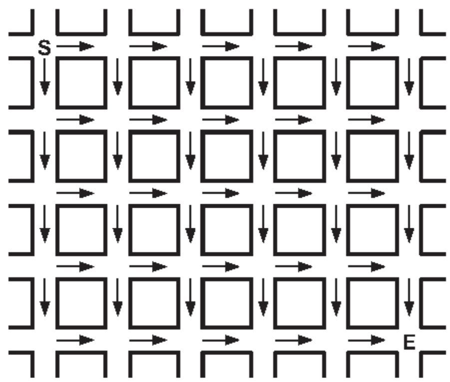

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

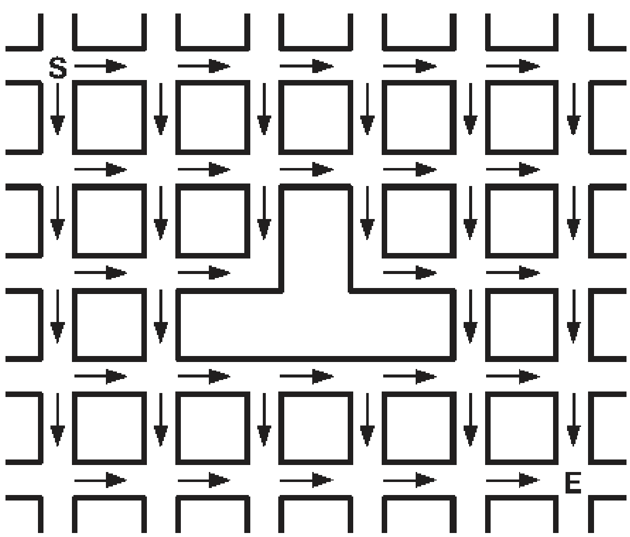

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

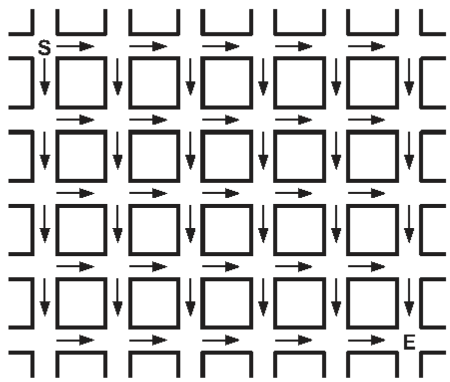

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

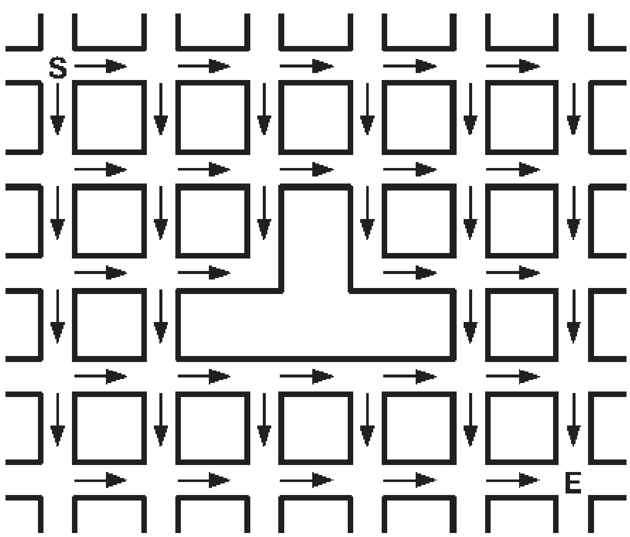

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

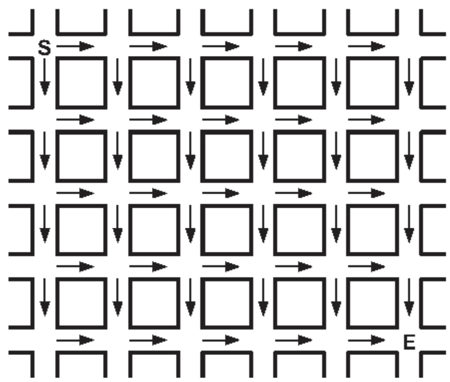

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

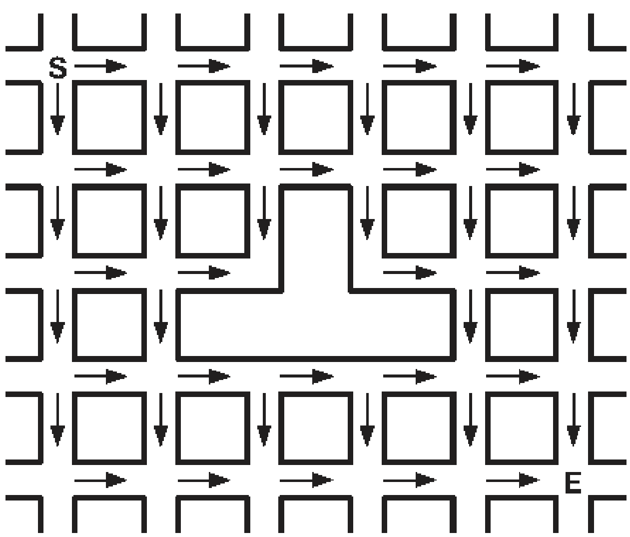

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

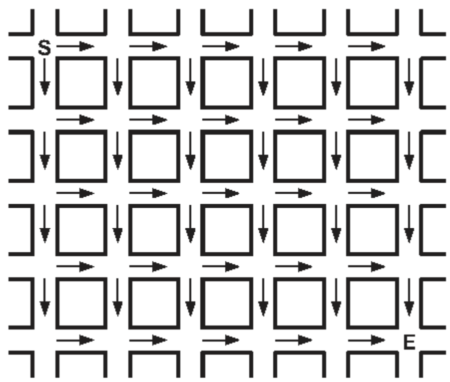

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

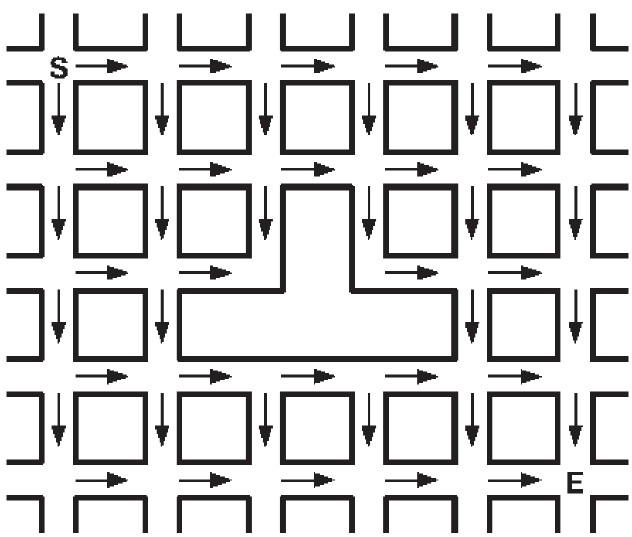

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

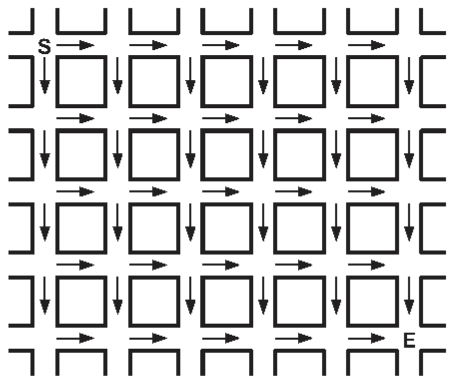

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

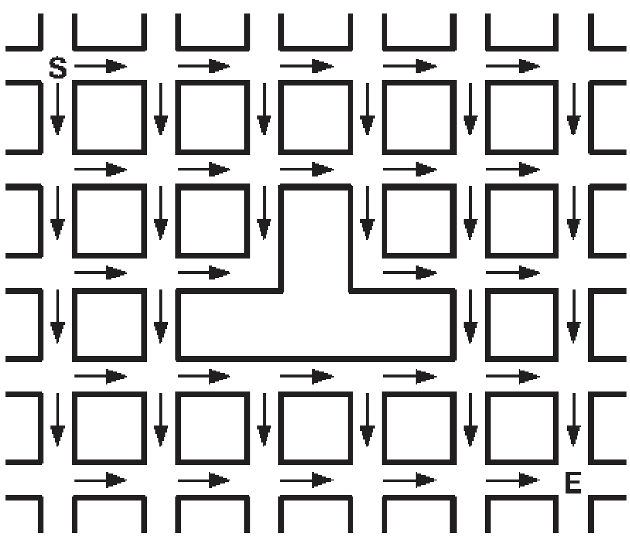

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

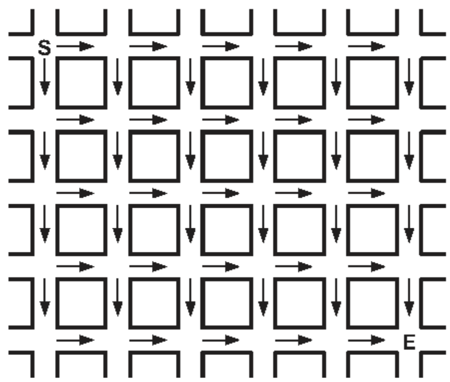

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

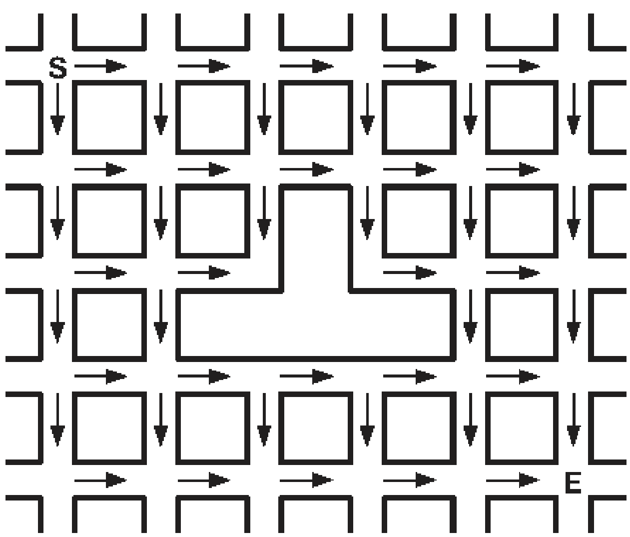

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

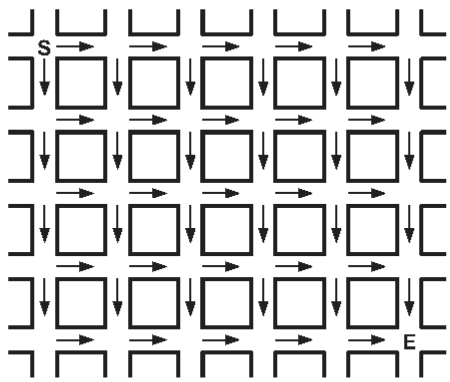

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

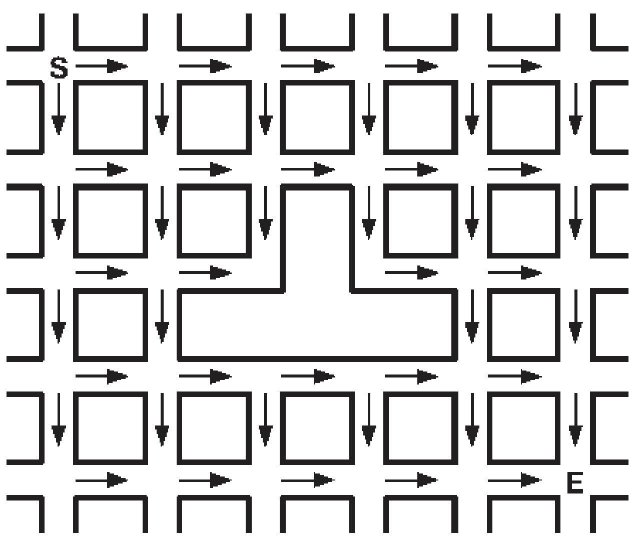

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

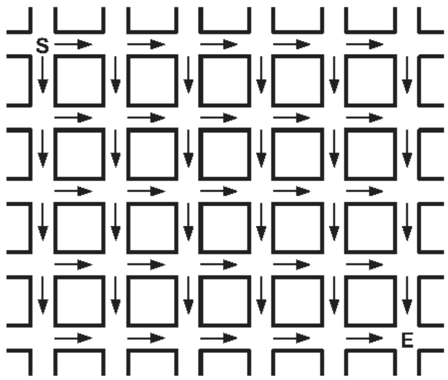

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

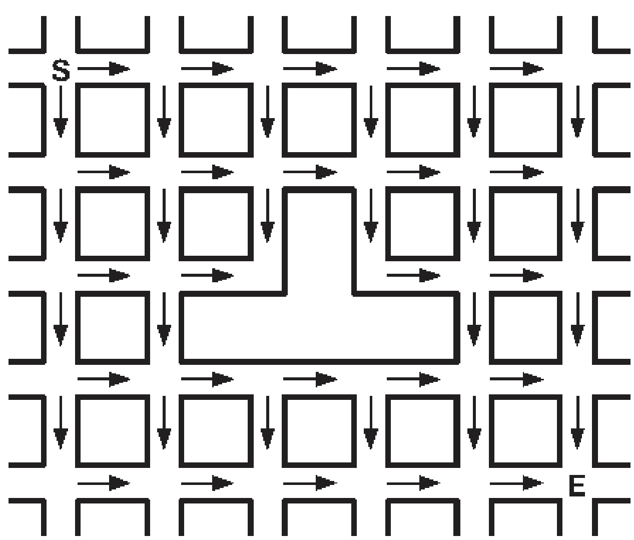

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

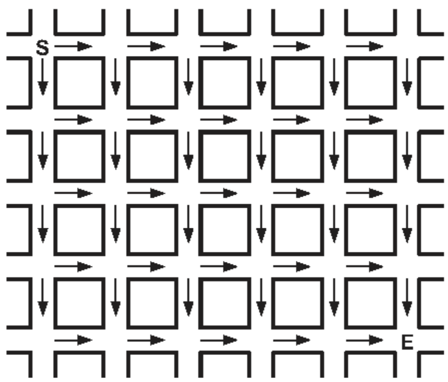

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

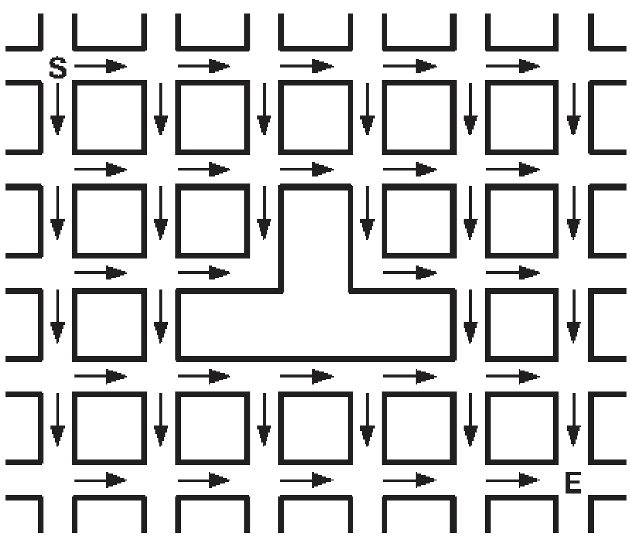

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

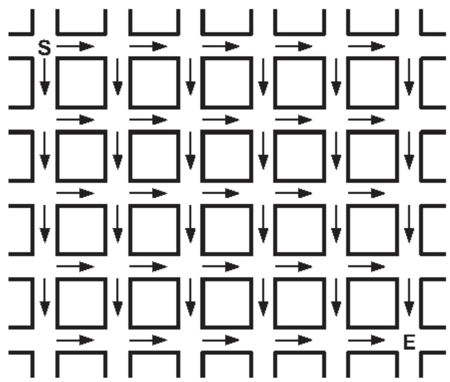

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

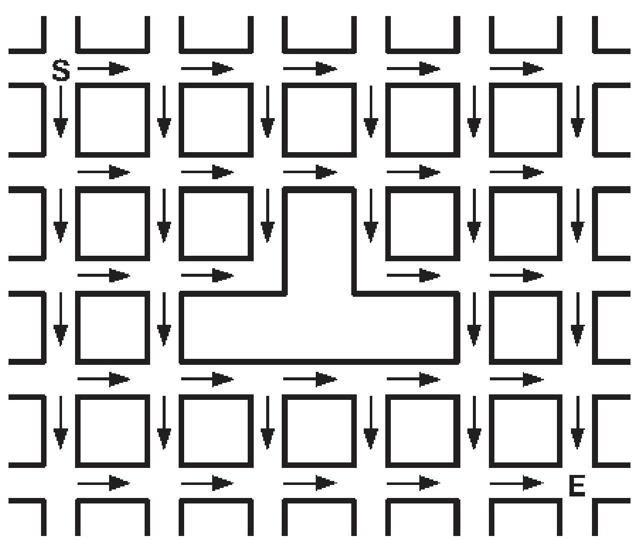

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

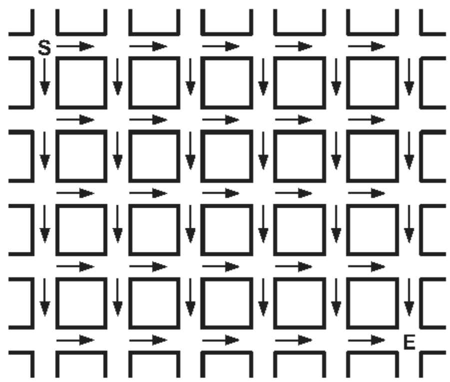

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

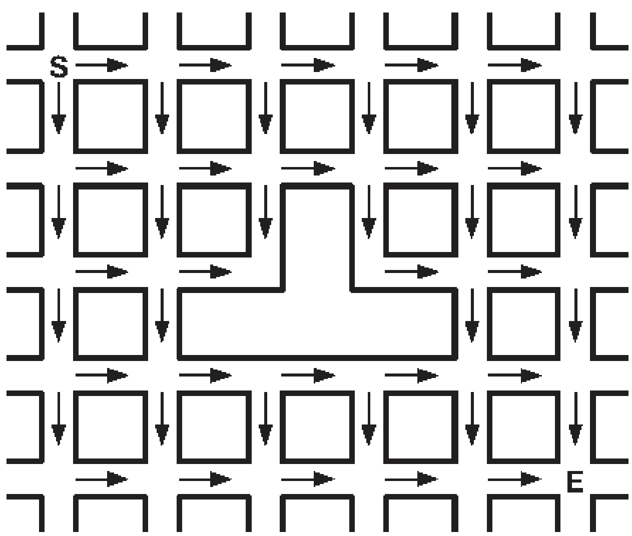

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

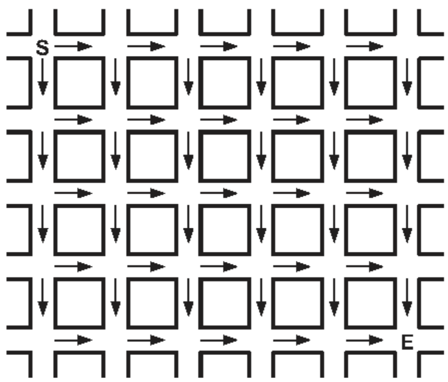

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

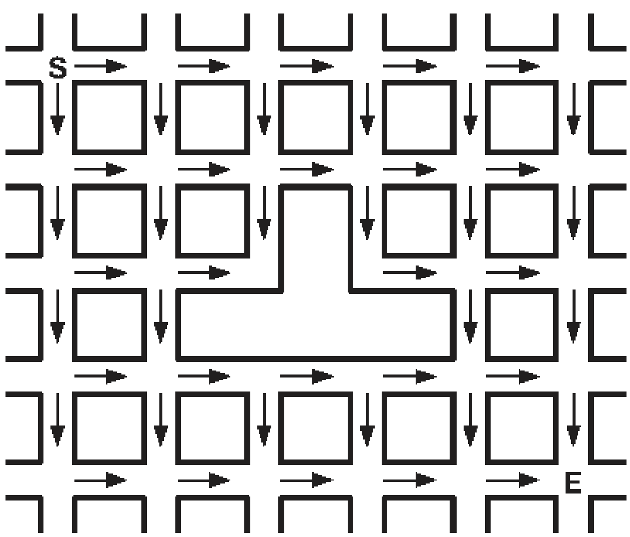

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

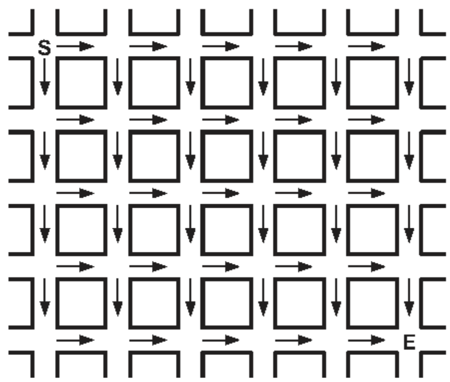

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

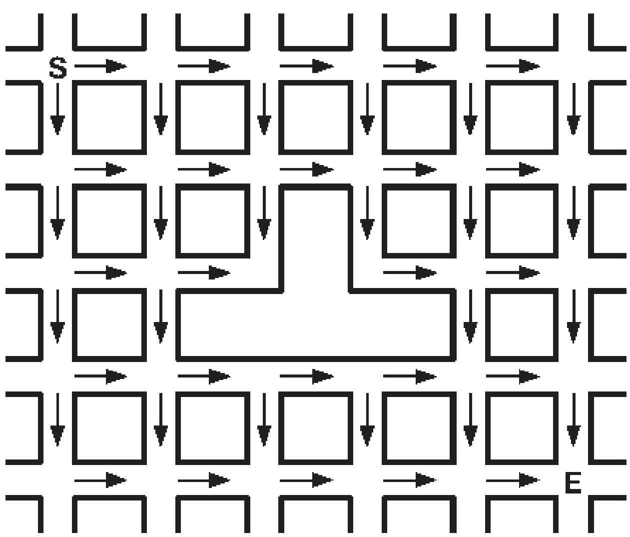

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

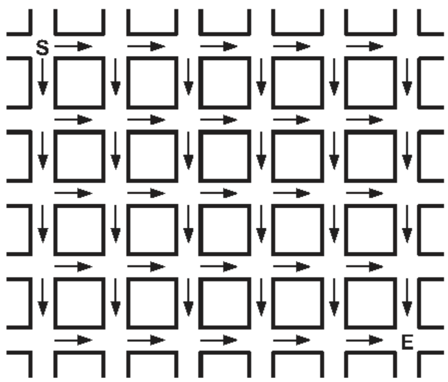

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

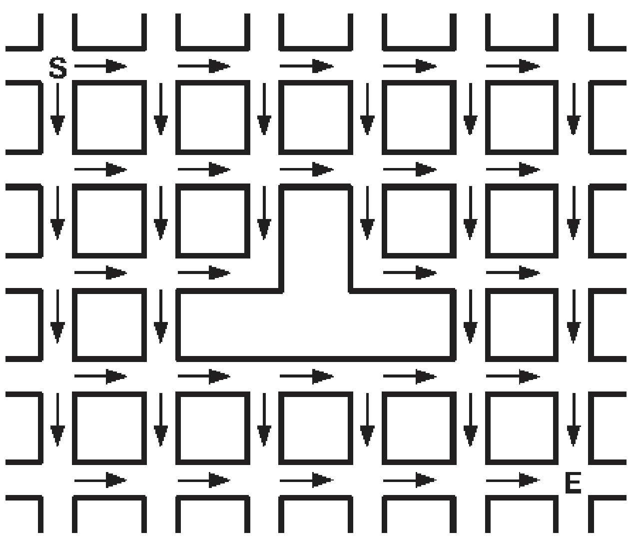

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

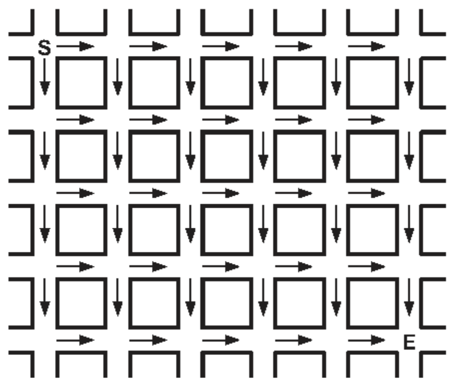

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

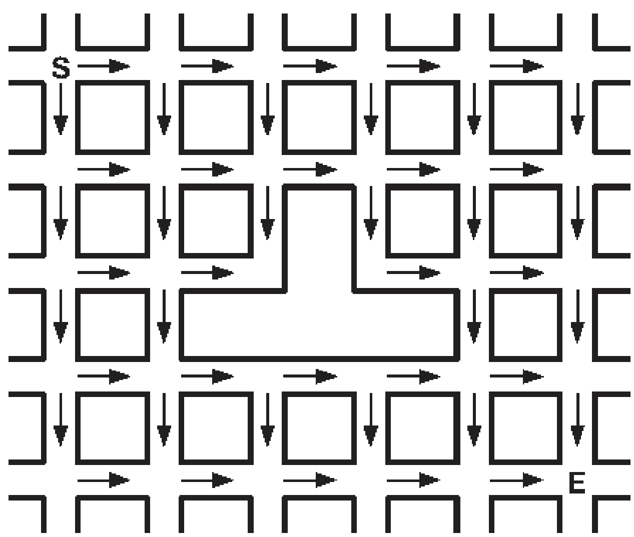

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

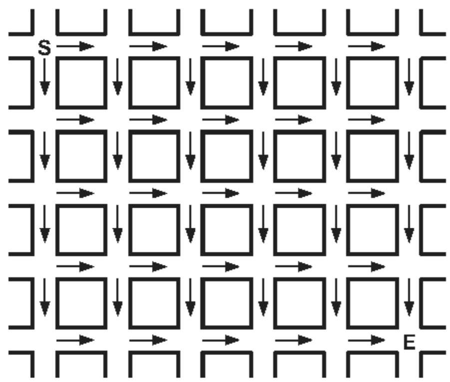

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

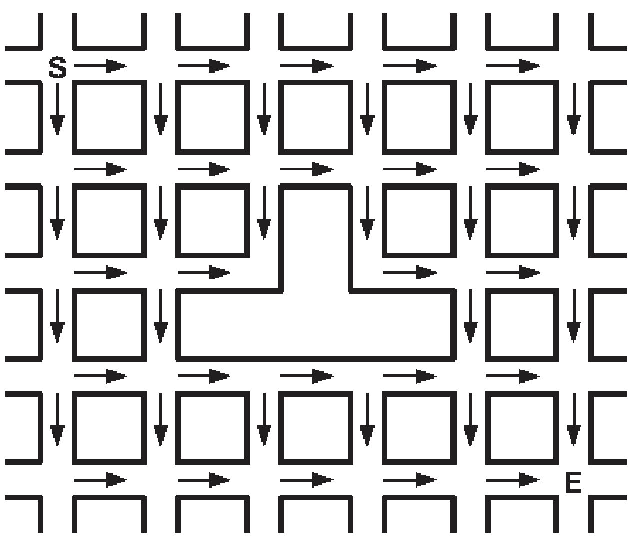

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

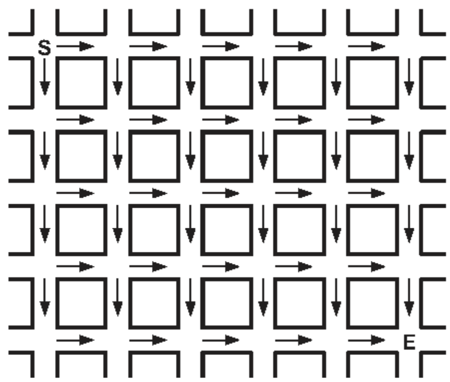

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

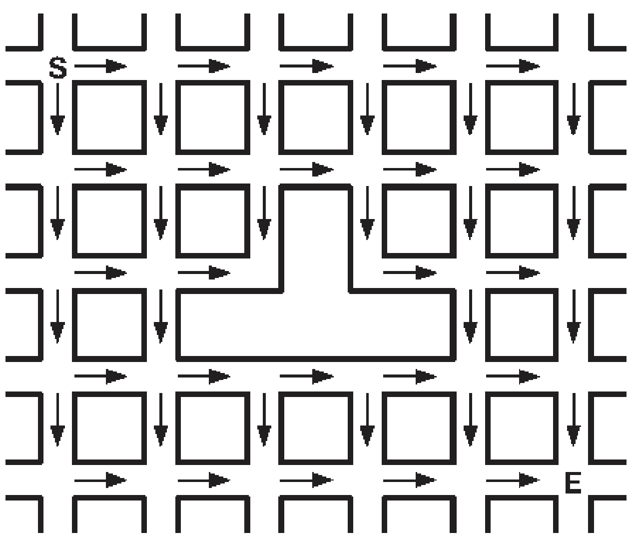

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

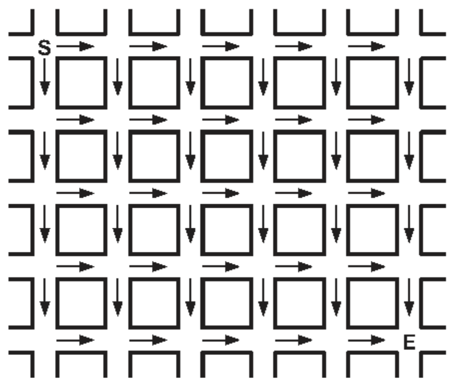

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

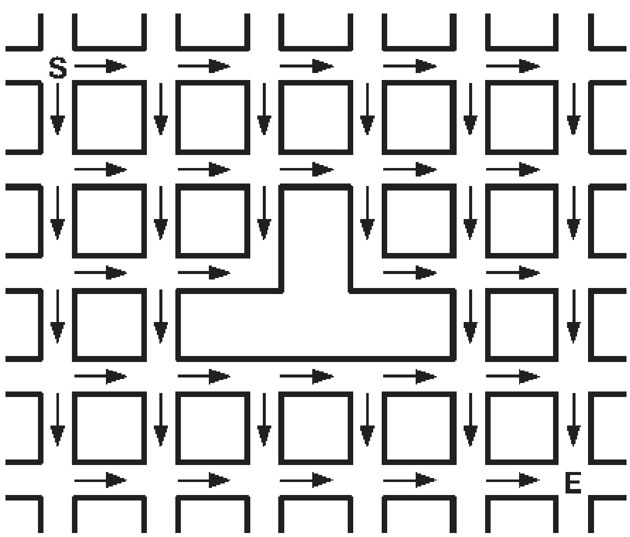

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

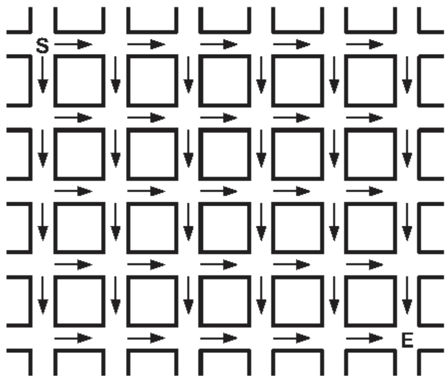

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

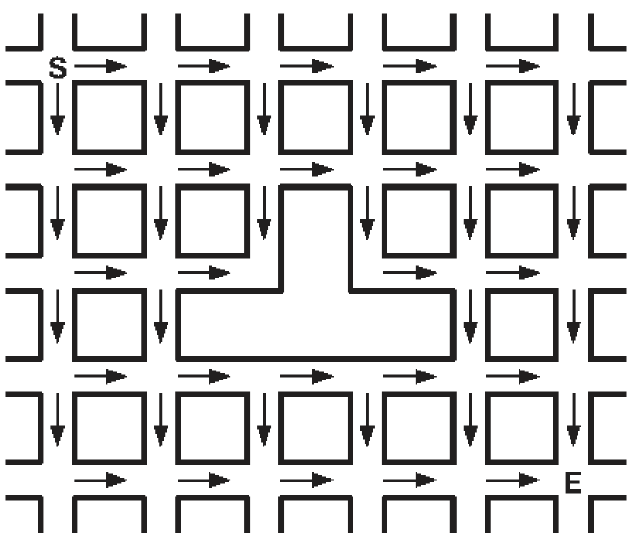

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

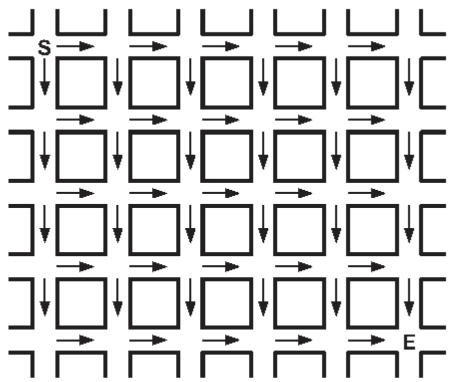

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

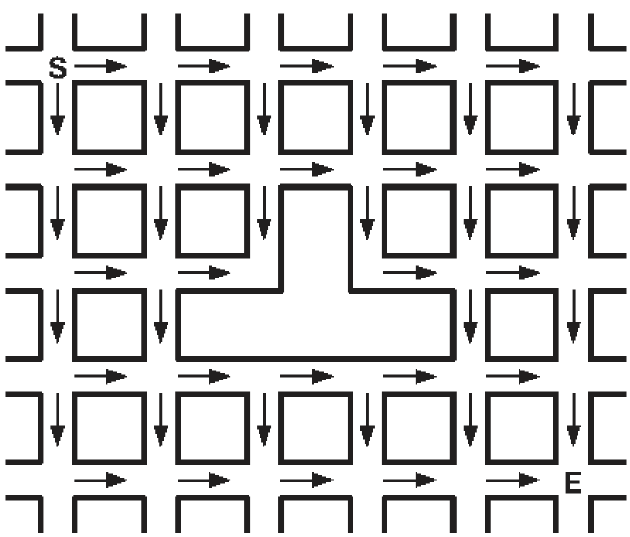

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that $1<i<n$, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

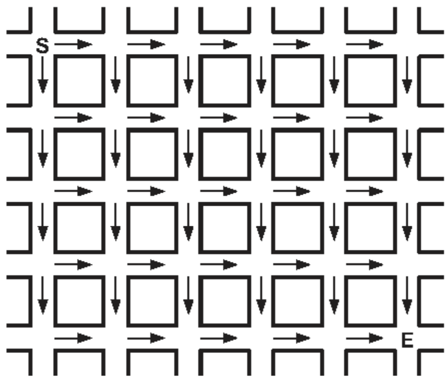

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

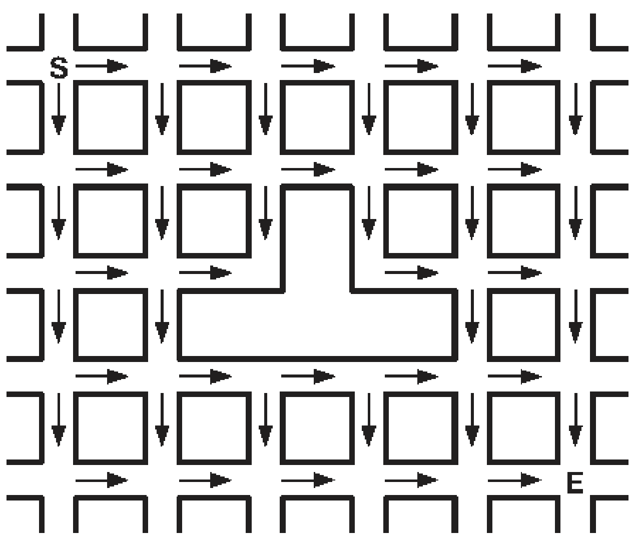

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

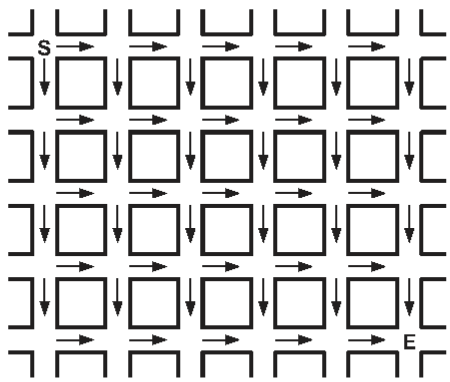

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

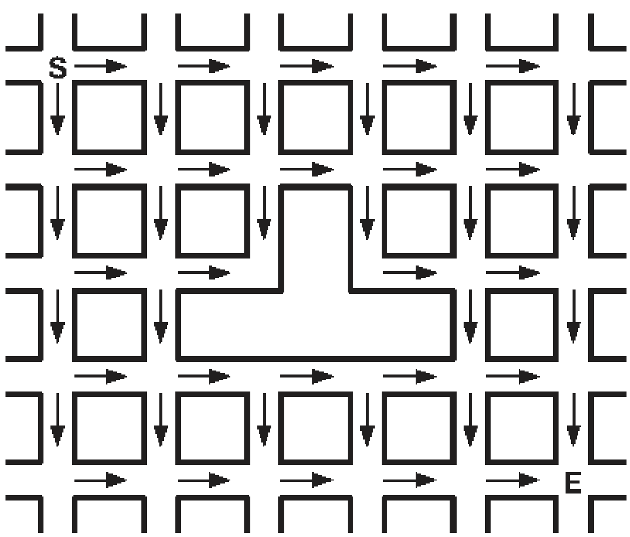

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

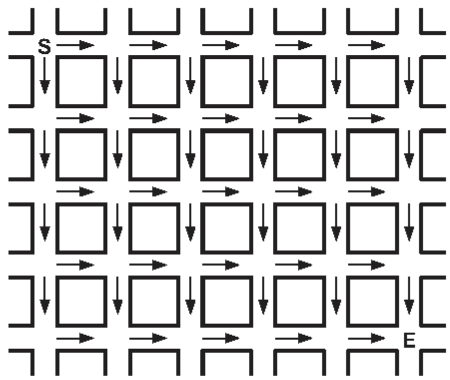

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

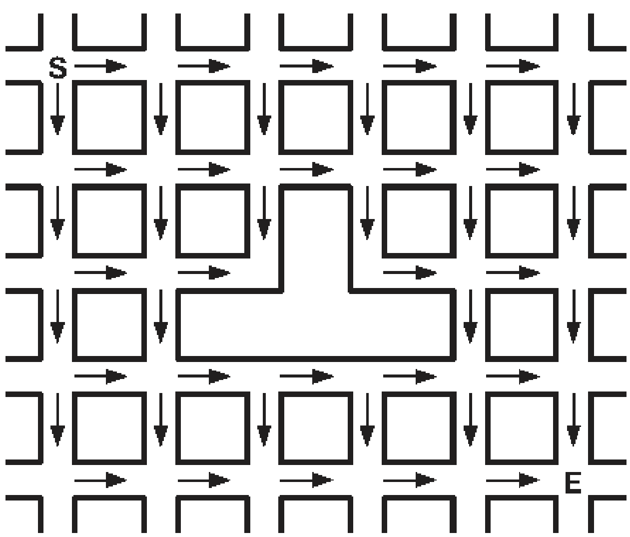

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

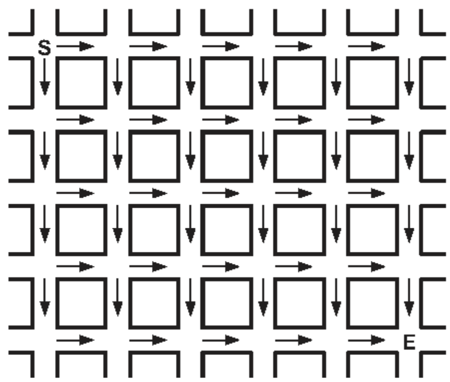

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

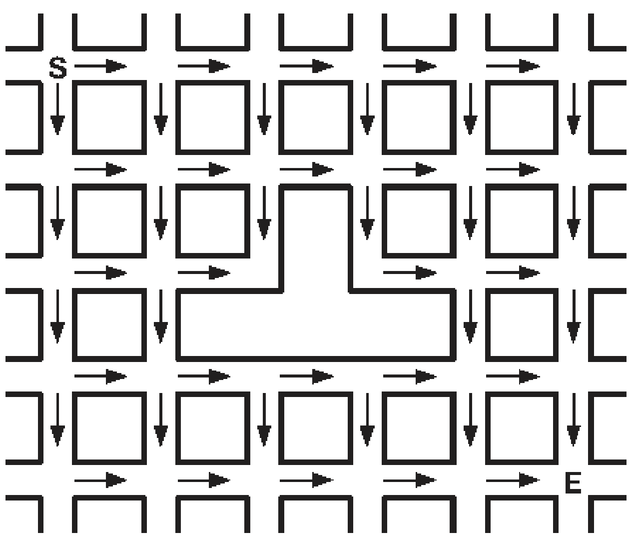

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \dots, p_n\}$ and the nights $\{d_1, d_2, \dots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \dots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

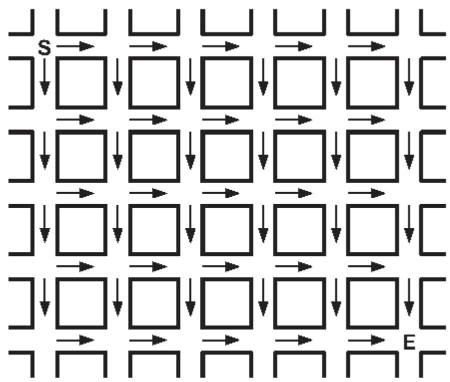

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

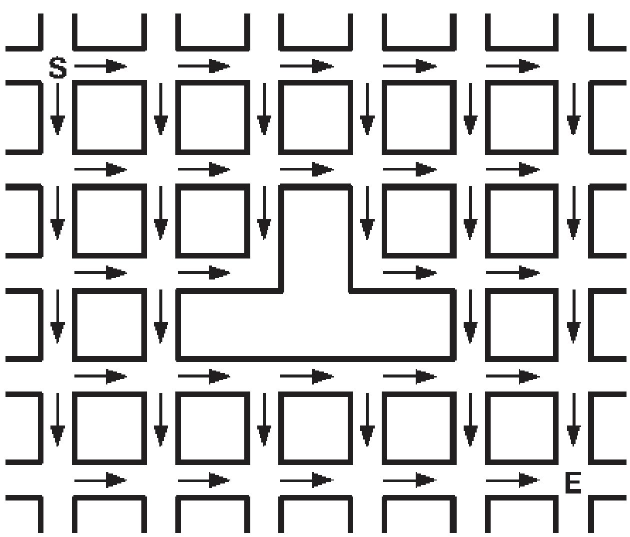

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

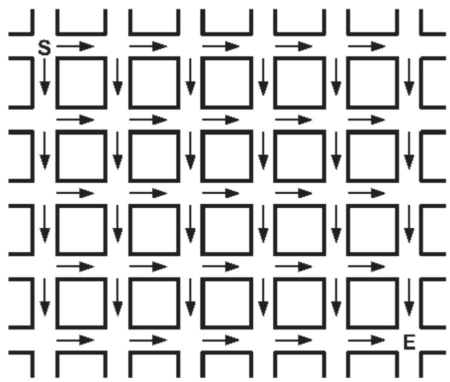

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

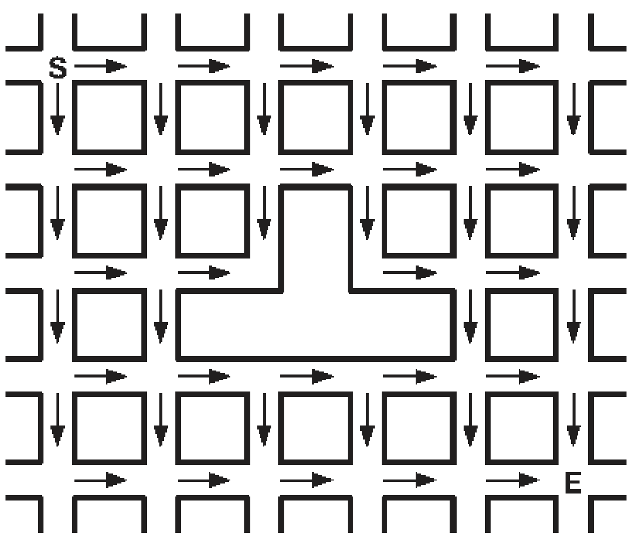

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

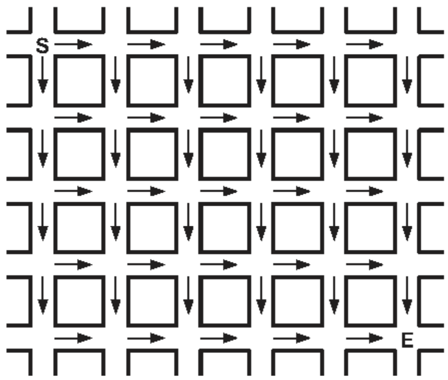

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

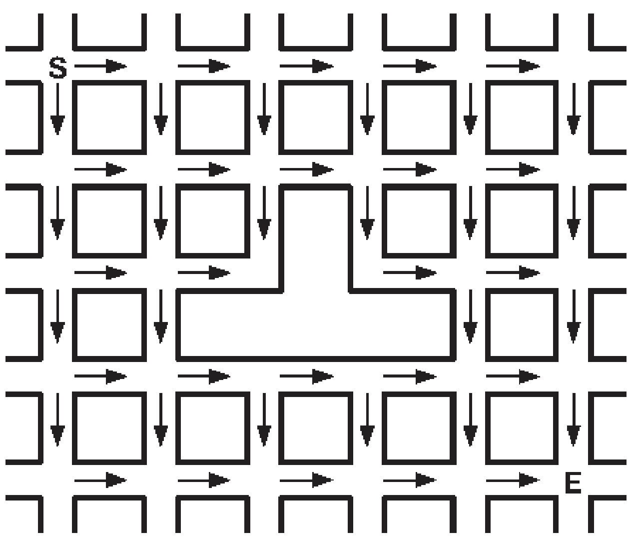

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

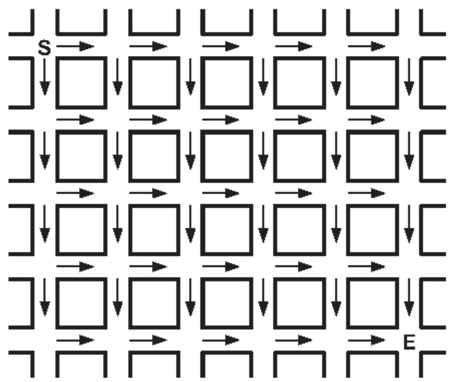

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

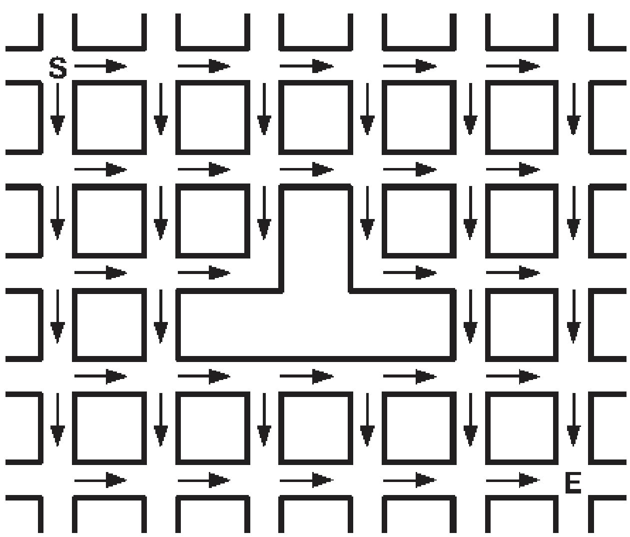

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

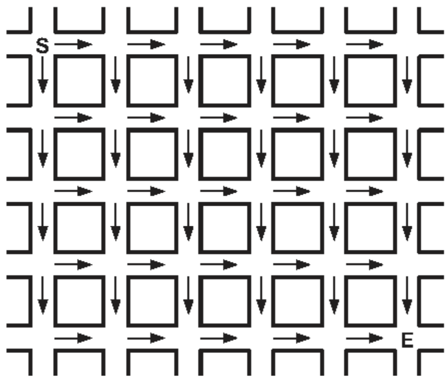

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

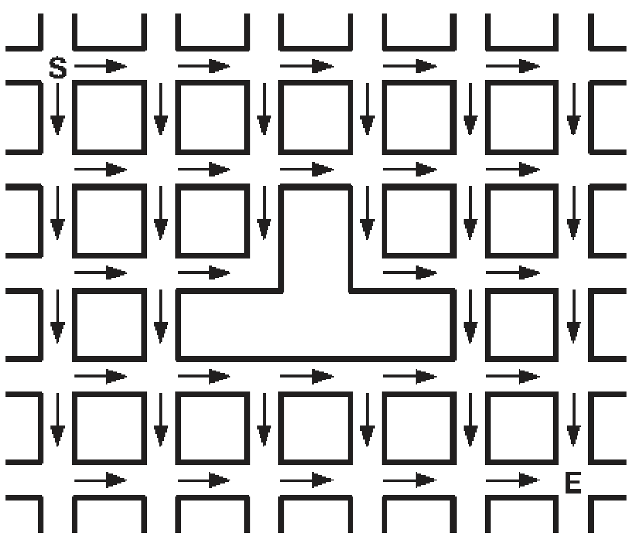

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

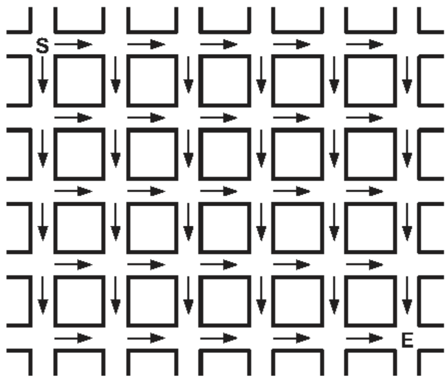

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

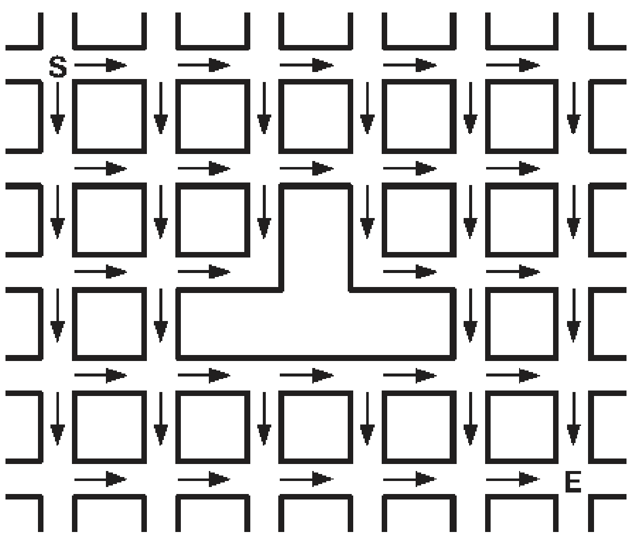

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

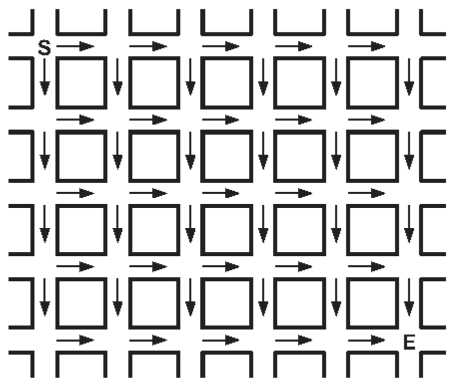

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

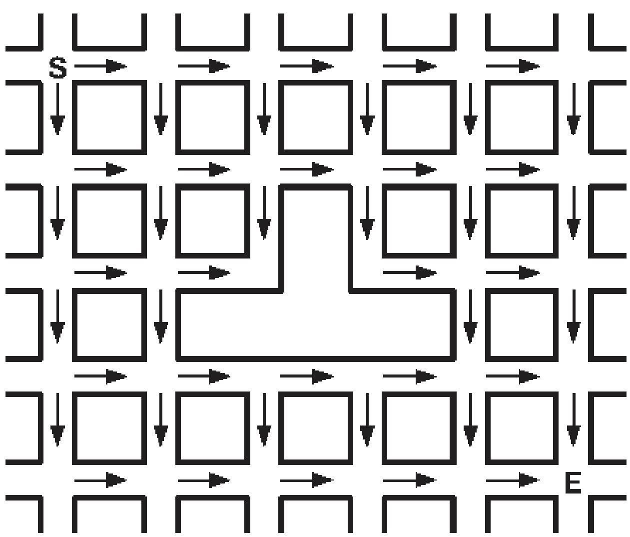

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

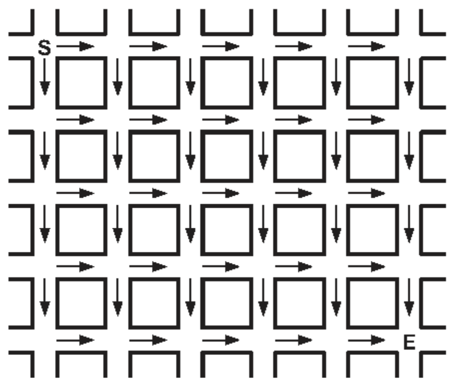

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

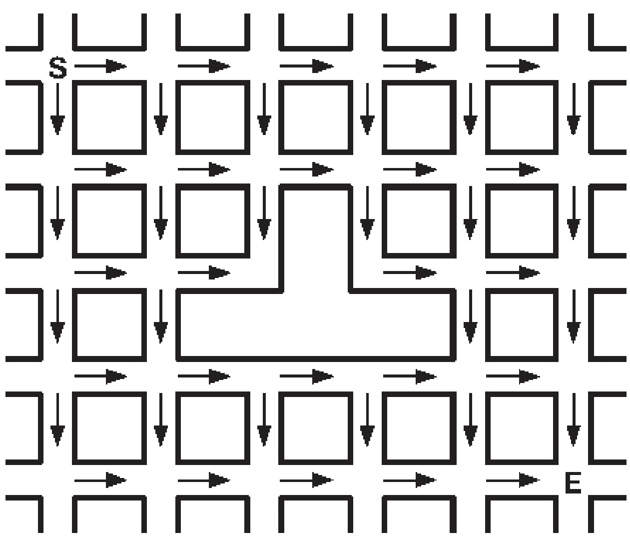

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

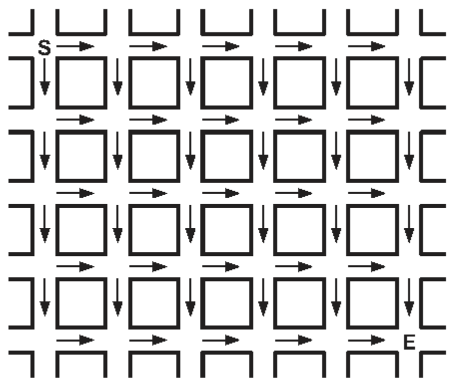

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

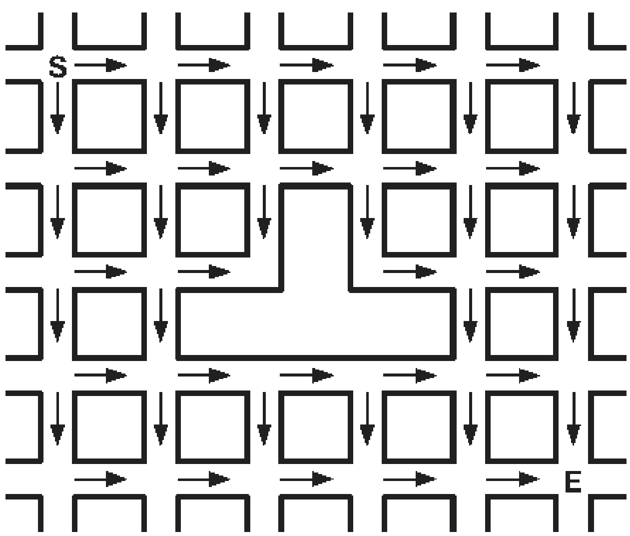

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

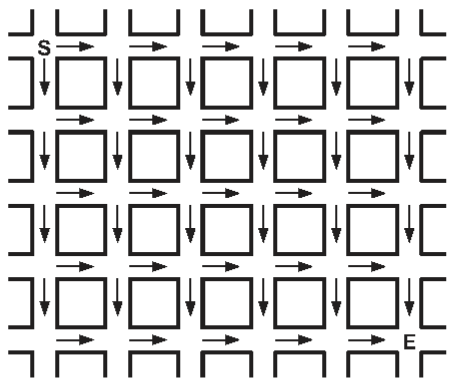

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

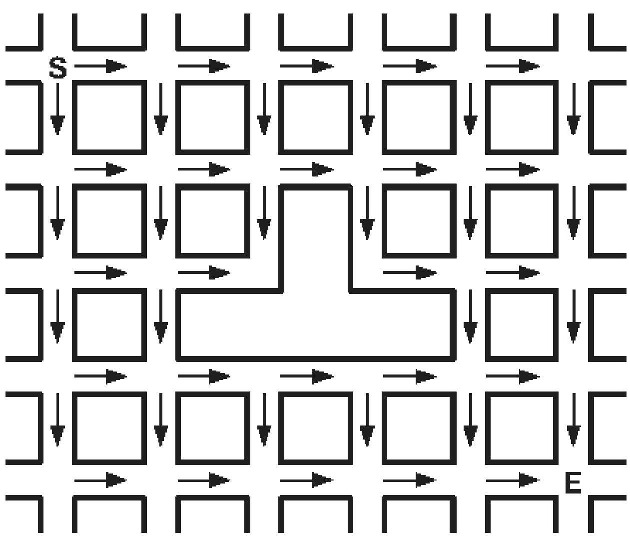

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

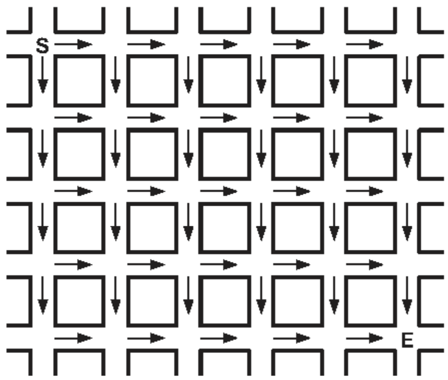

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

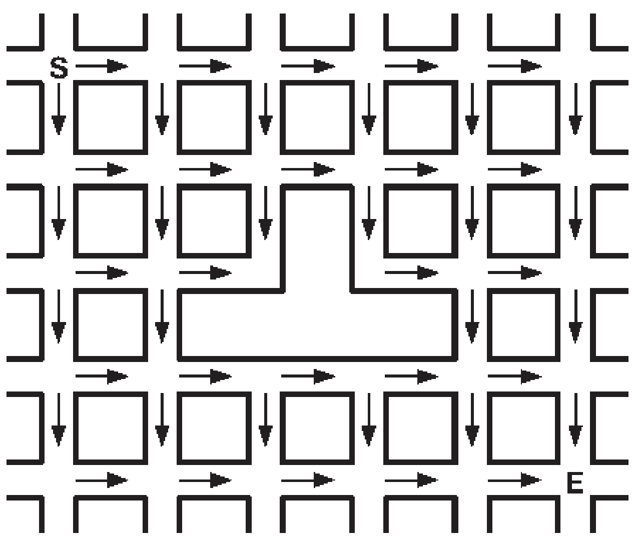

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

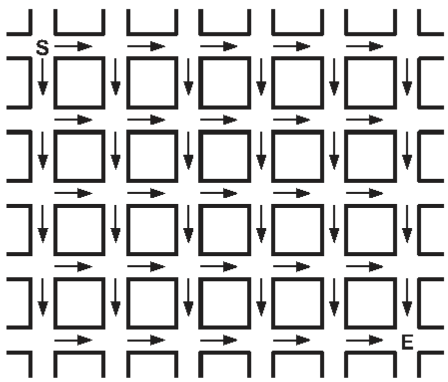

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

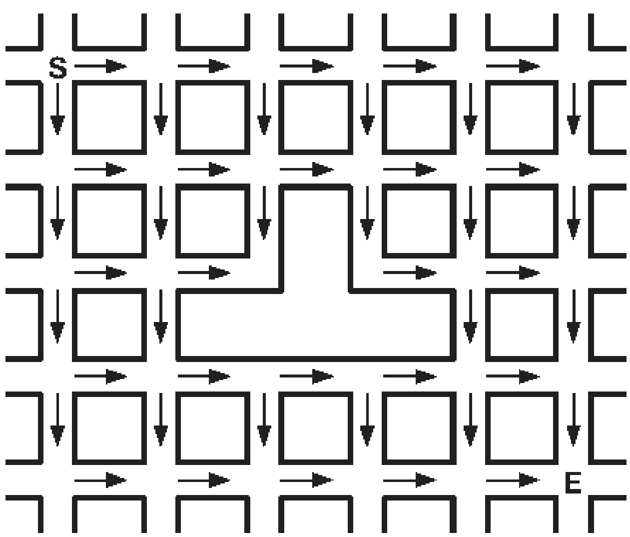

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

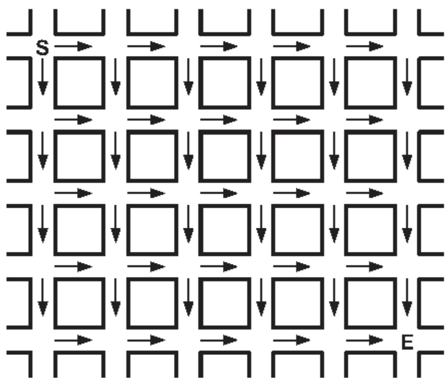

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

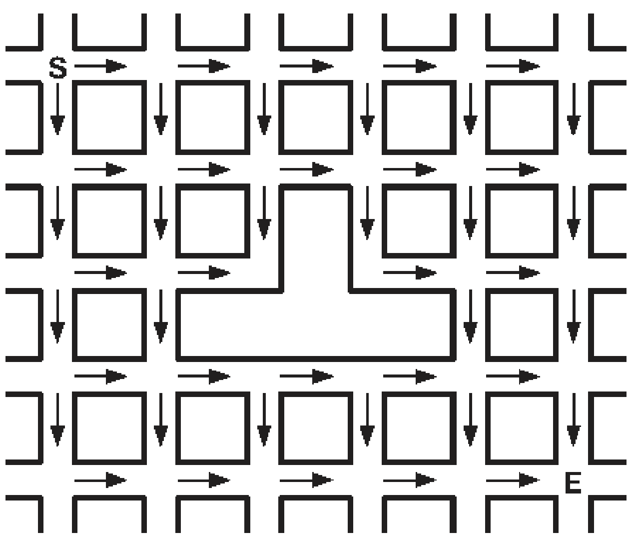

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

4056FE46-720F-4731-B047-DC3C10C23C40

SPRING2014_CSCI570_EXAM2

#175

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

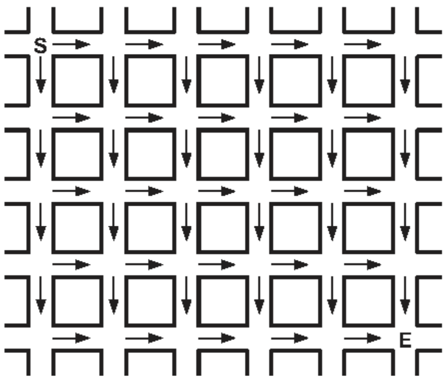

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

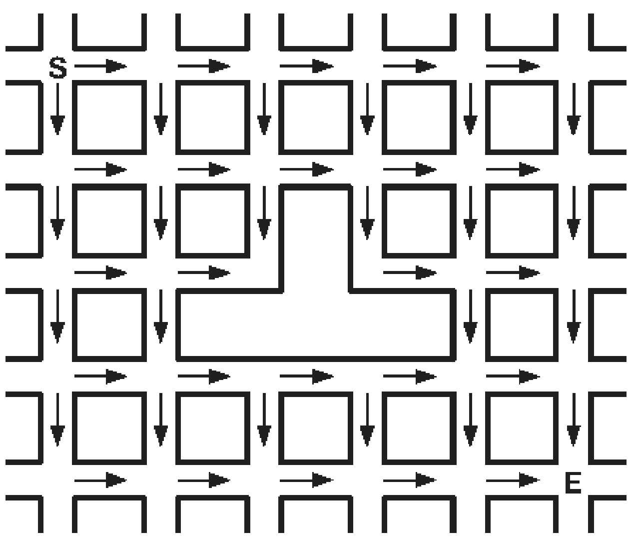

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \dots, p_n\}$ and the nights $\{d_1, d_2, \dots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \dots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that $n>2$, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that $1<i<n$, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

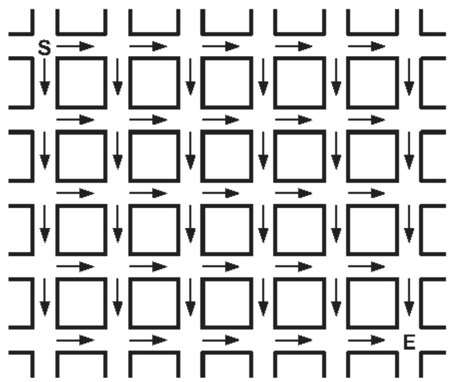

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

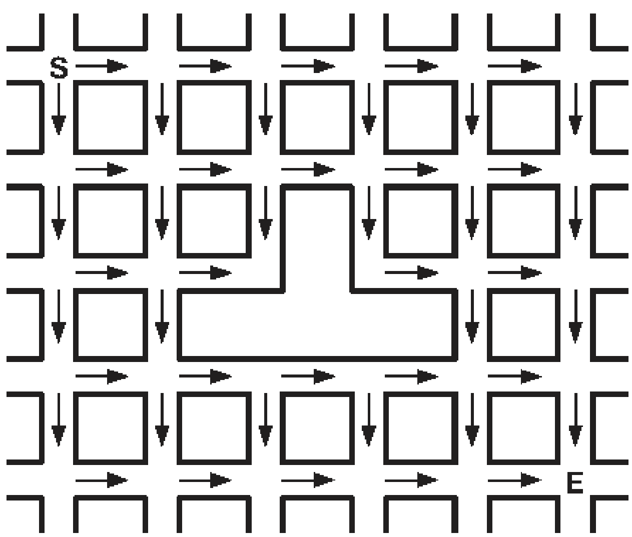

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

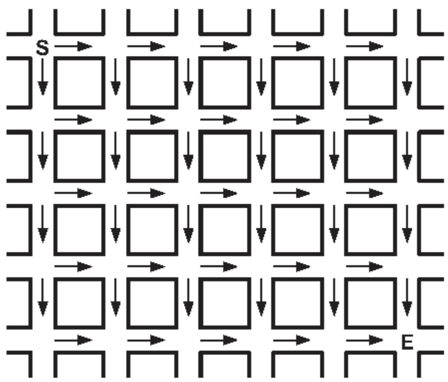

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

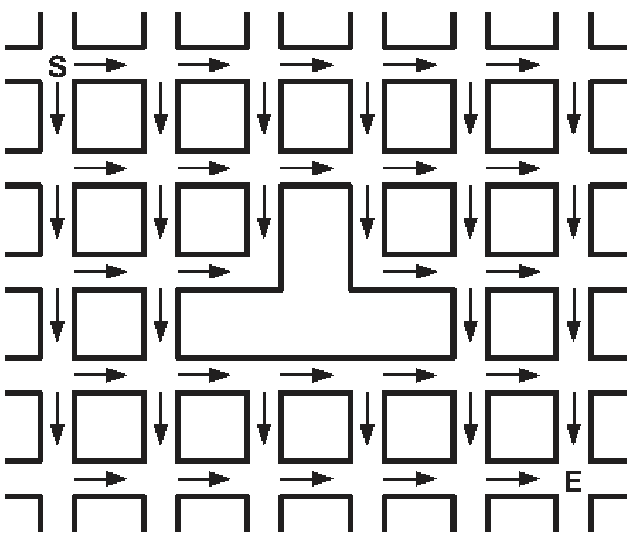

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

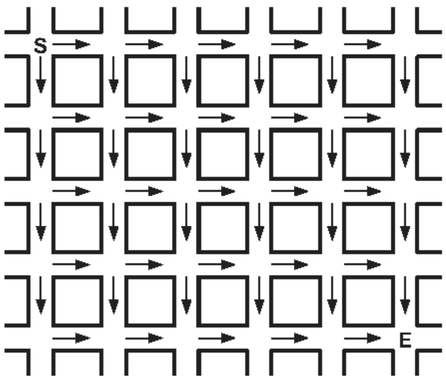

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

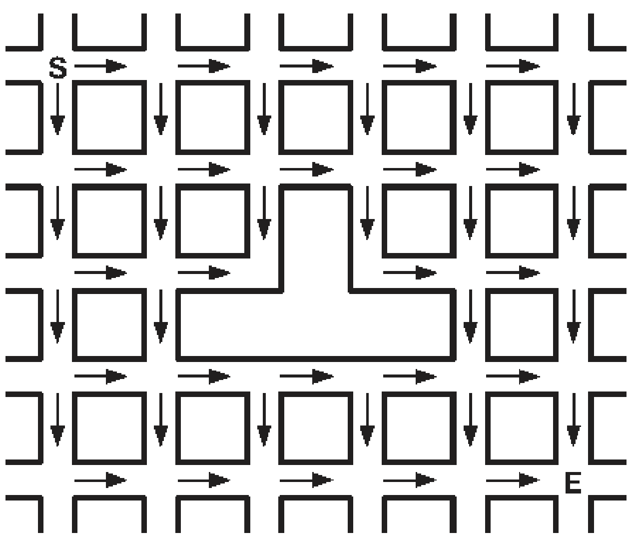

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

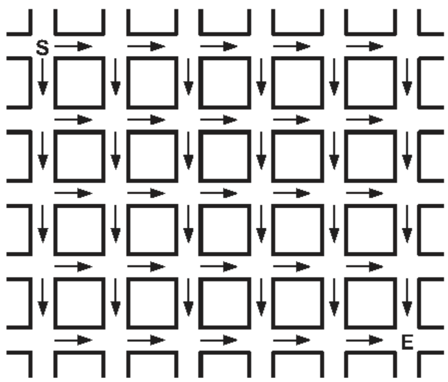

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

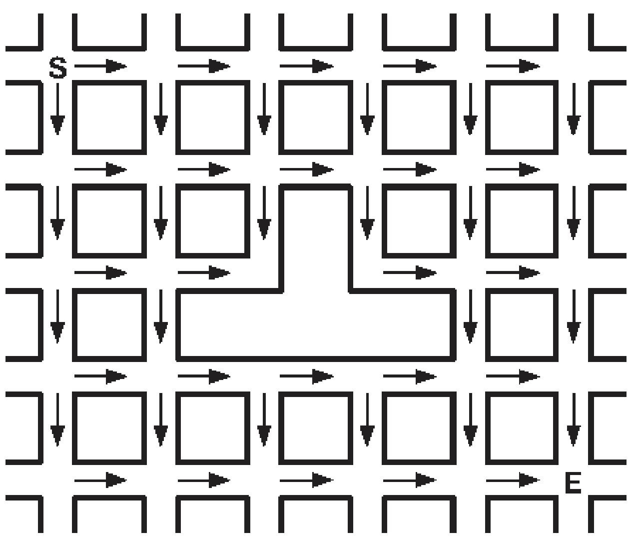

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts
Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.
Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

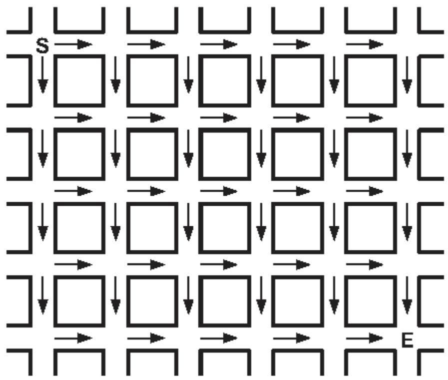

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

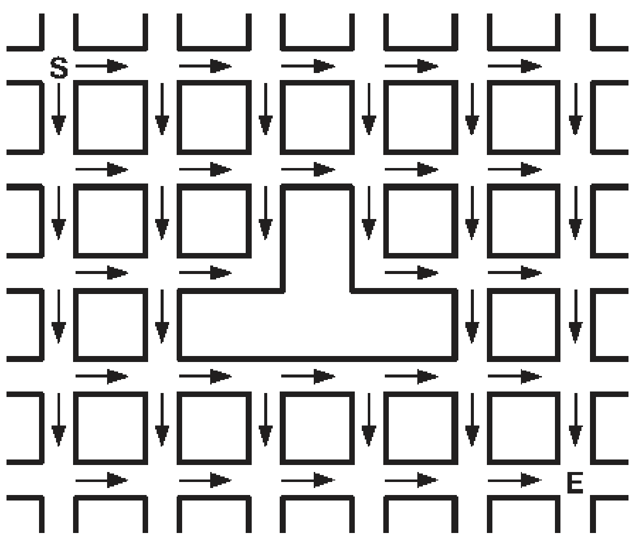

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

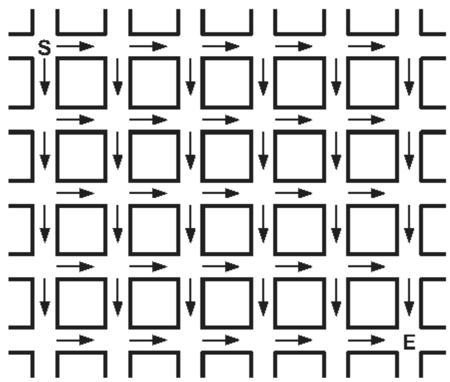

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

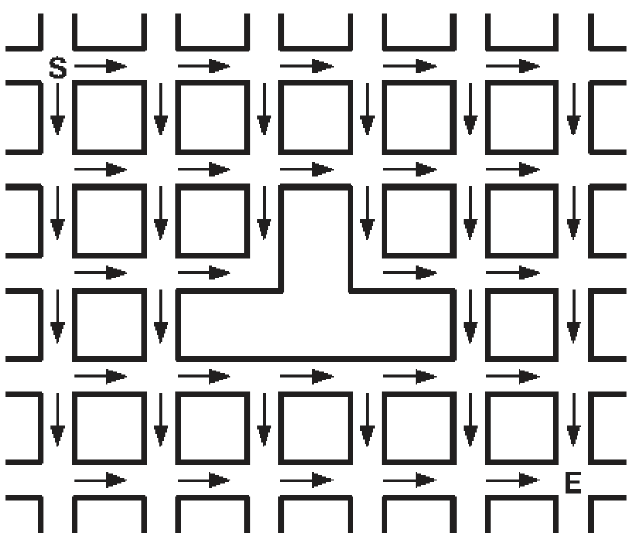

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

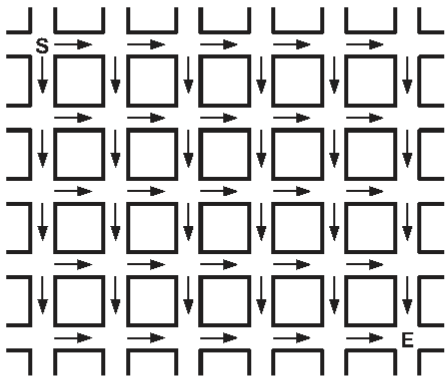

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

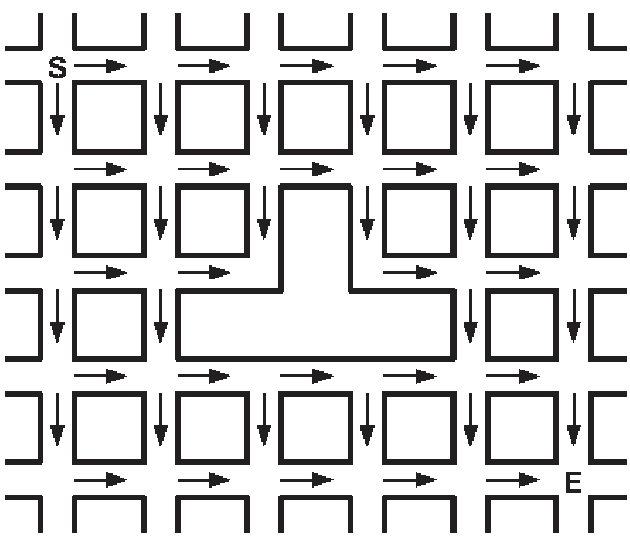

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

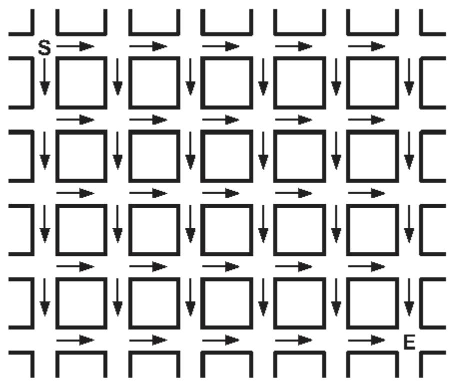

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

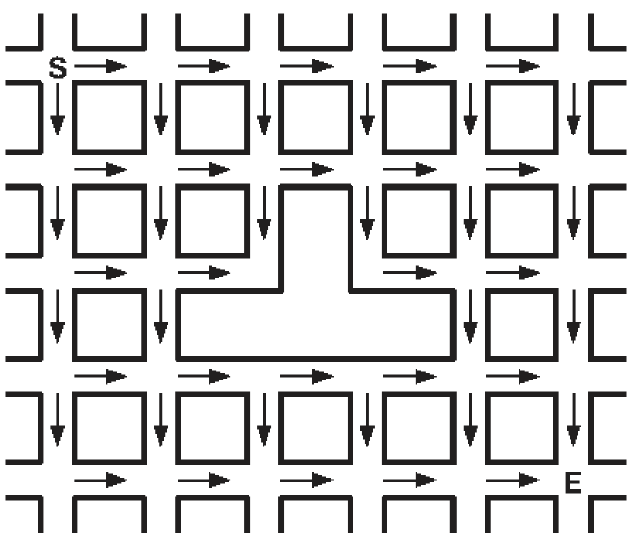

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \dots, p_n\}$ and the nights $\{d_1, d_2, \dots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \dots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

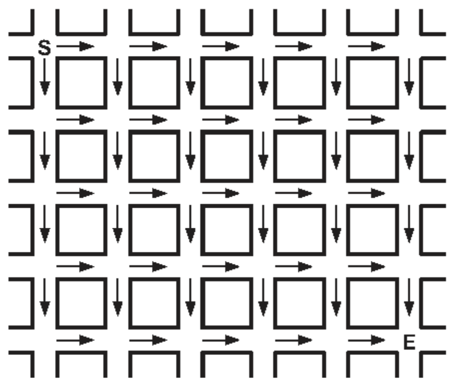

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

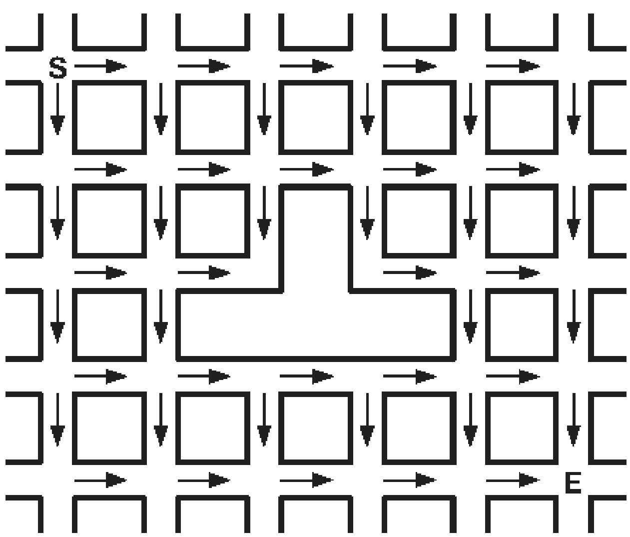

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

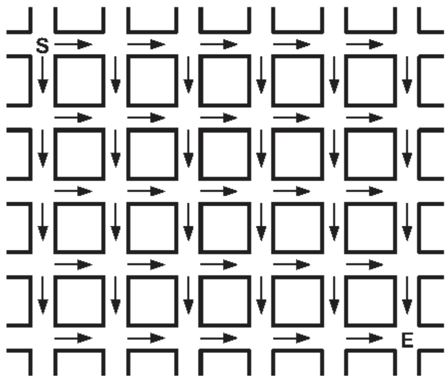

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

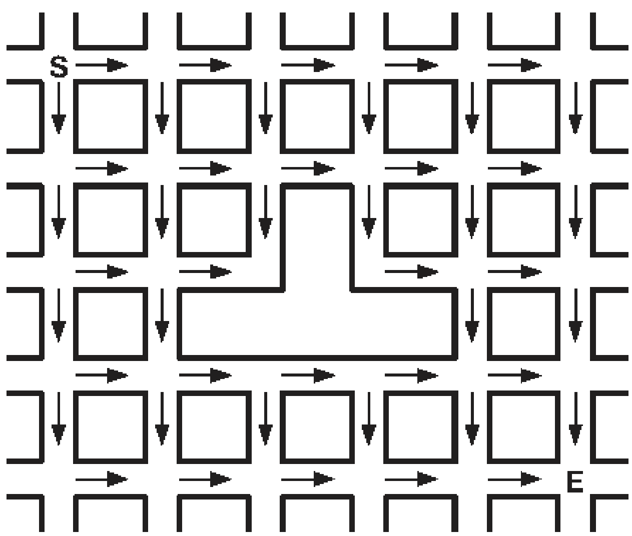

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

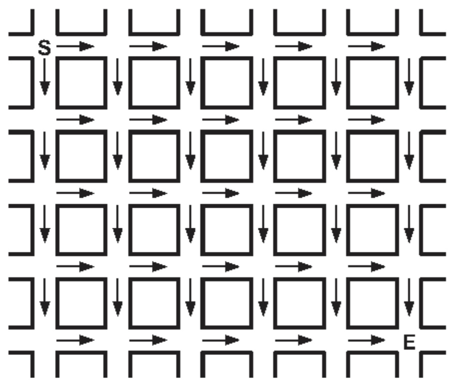

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

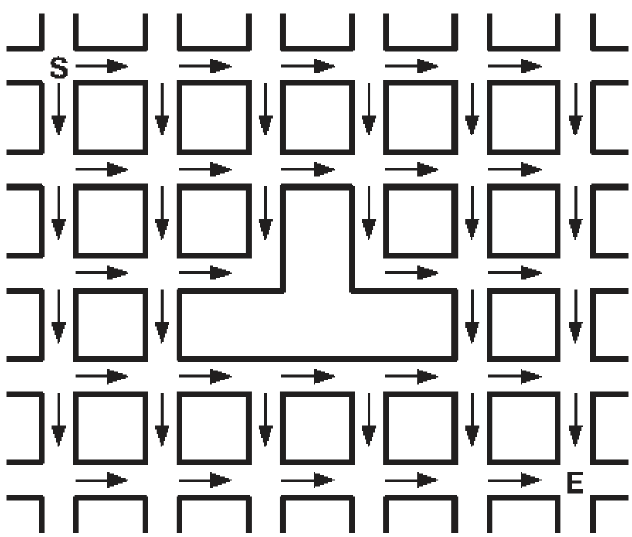

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

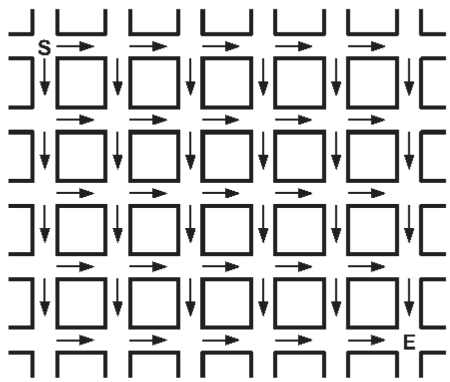

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

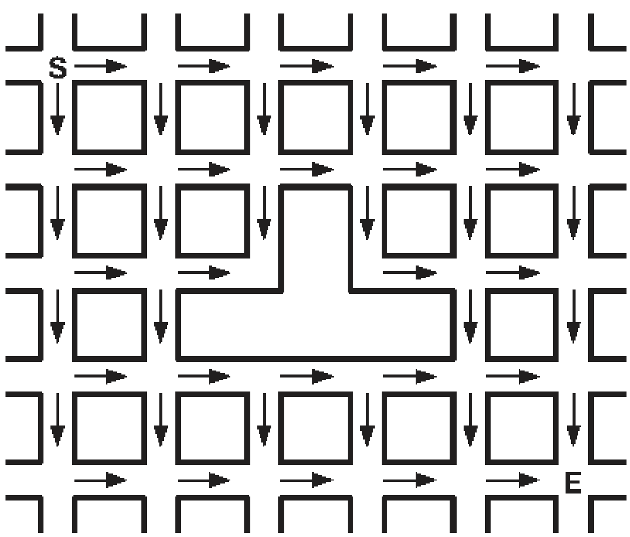

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

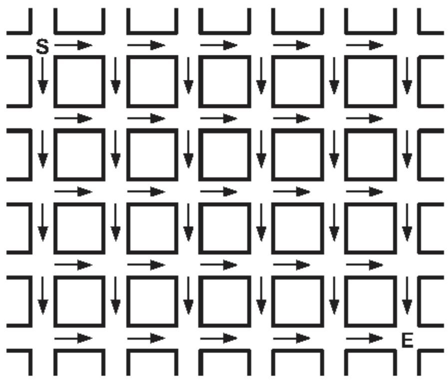

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

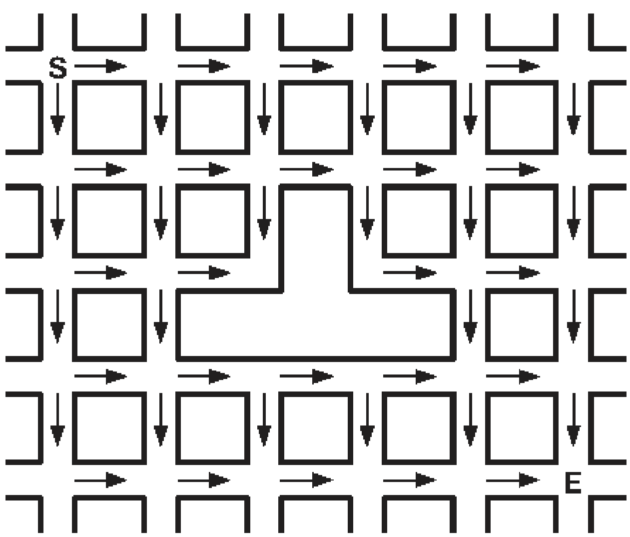

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

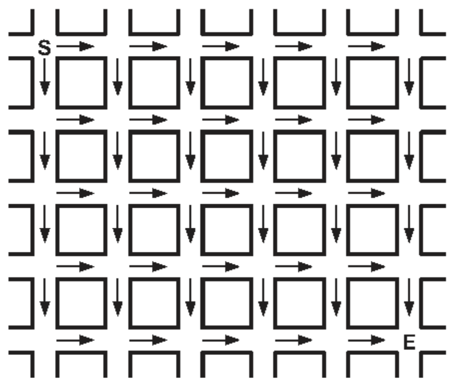

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

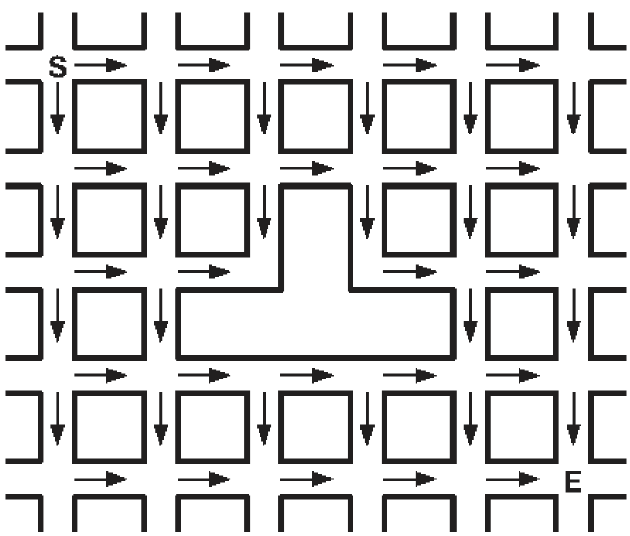

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

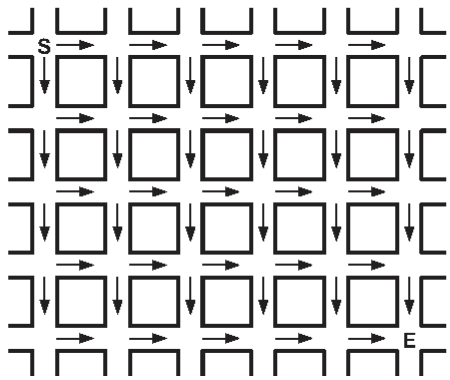

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

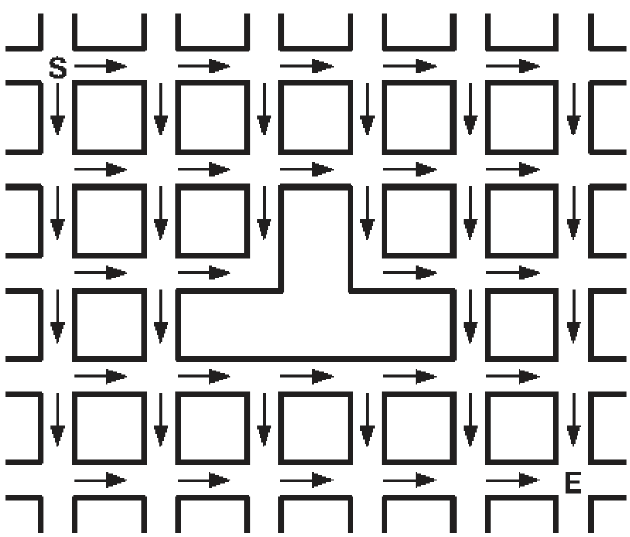

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

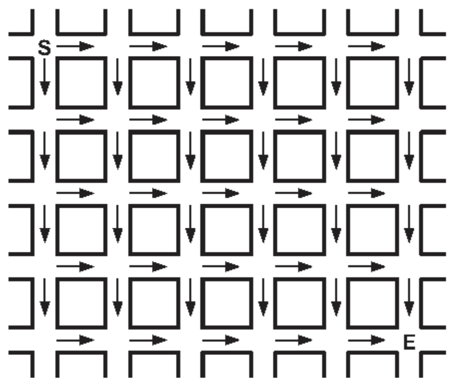

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

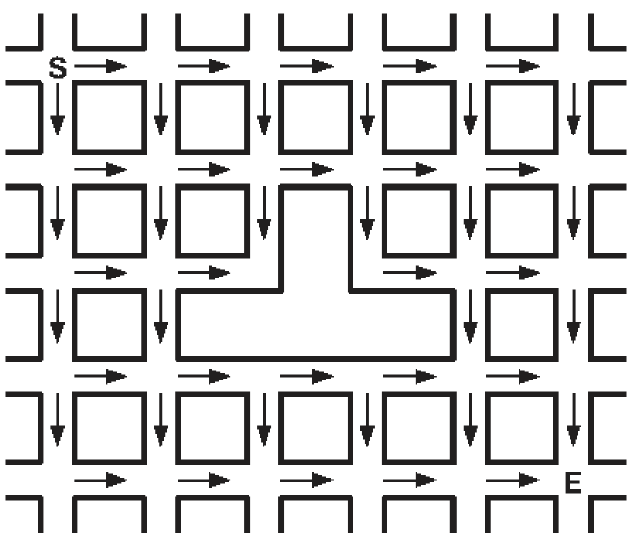

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

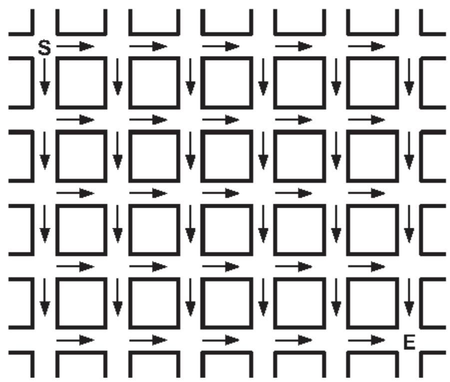

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

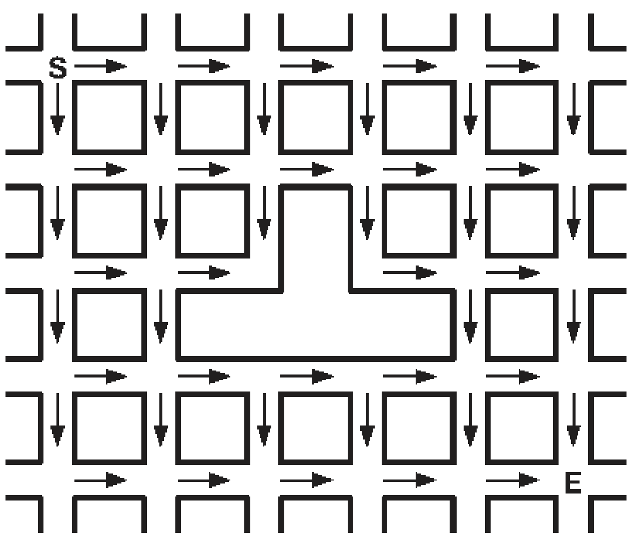

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

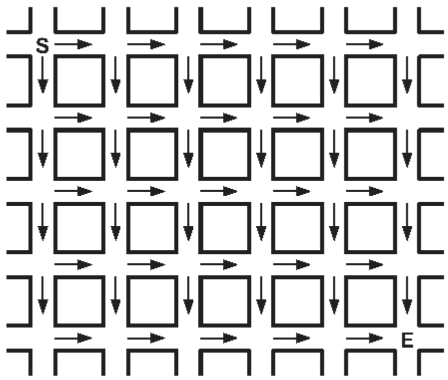

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

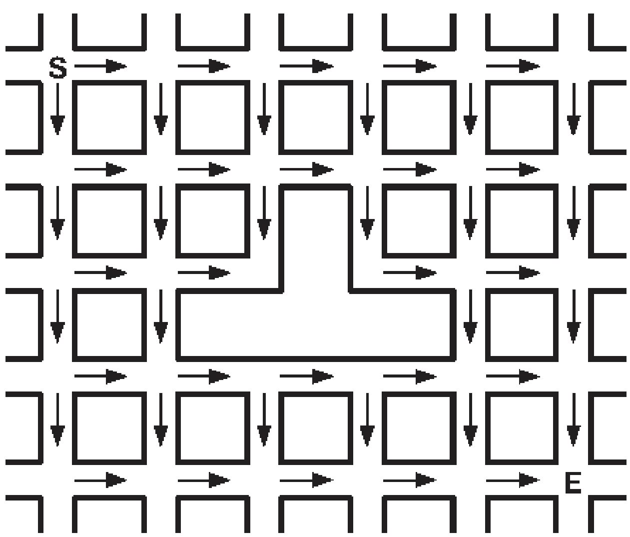

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

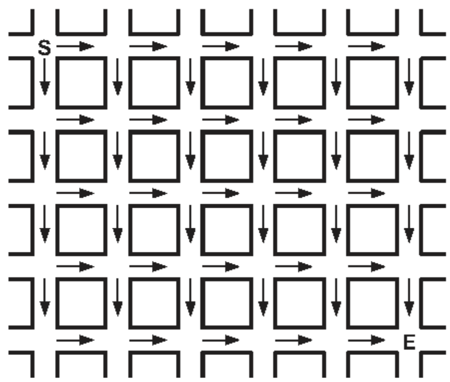

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

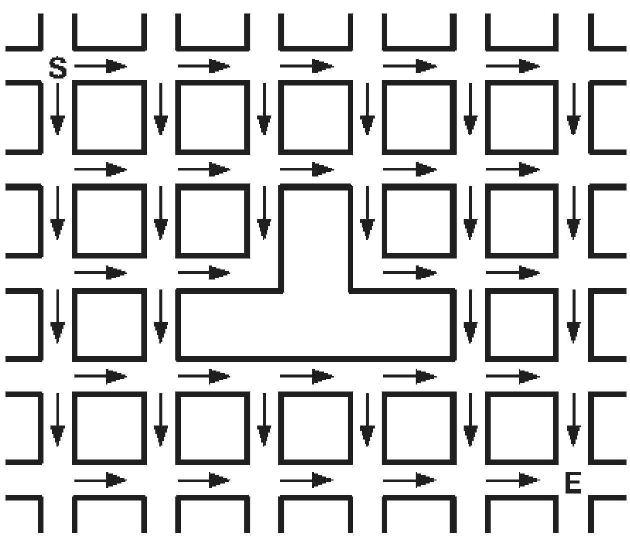

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

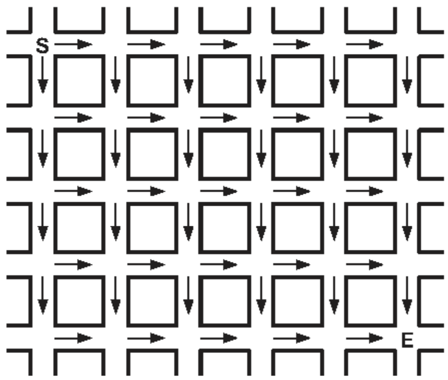

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

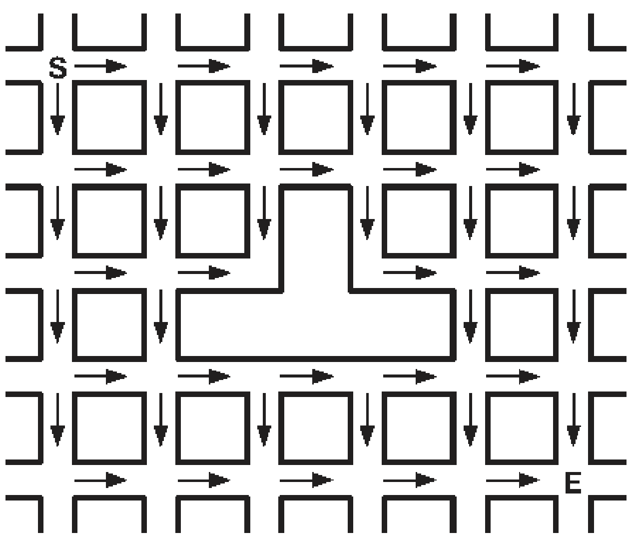

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

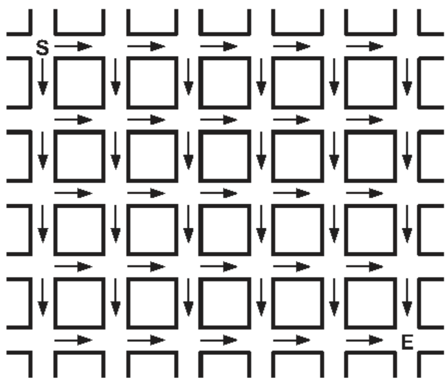

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

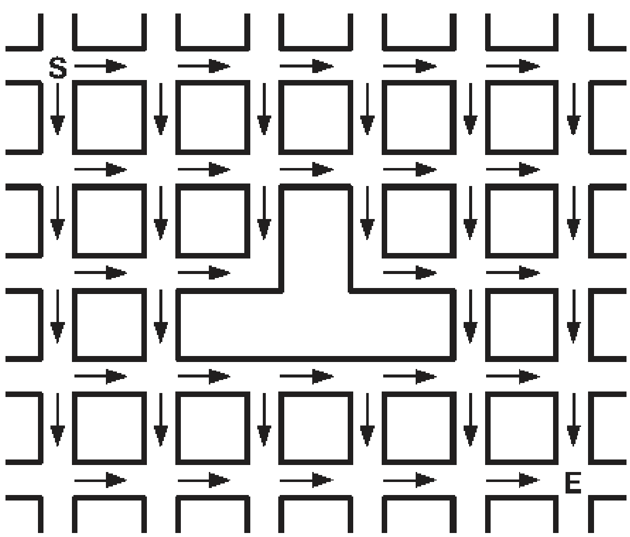

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

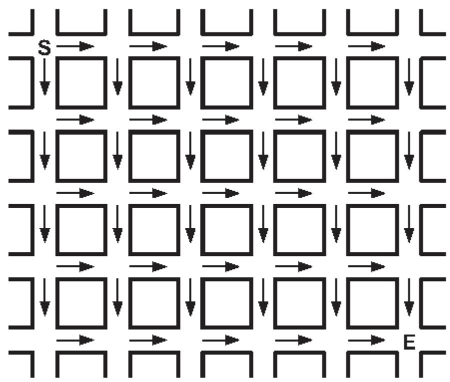

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

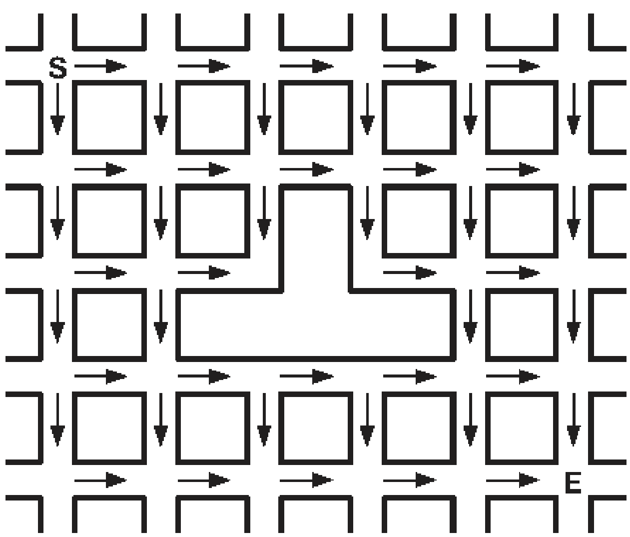

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

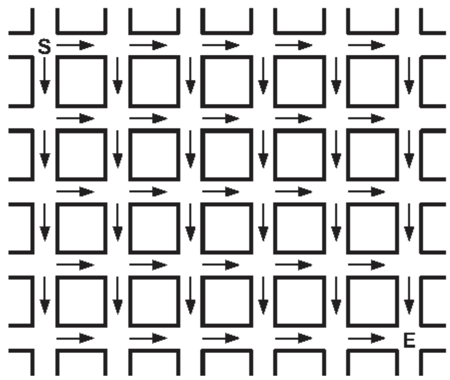

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

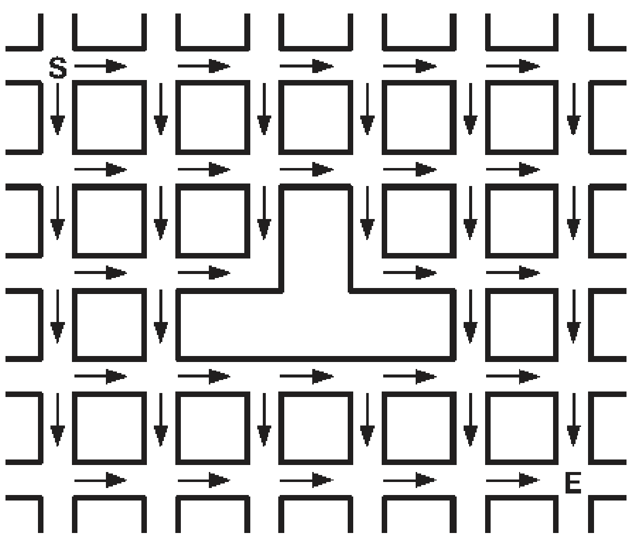

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

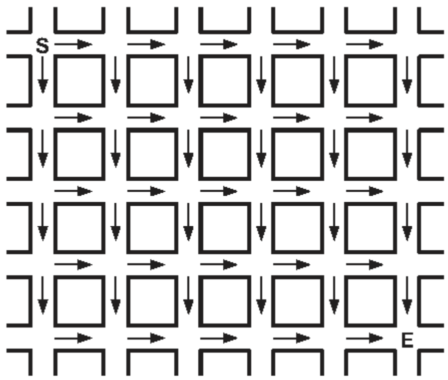

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

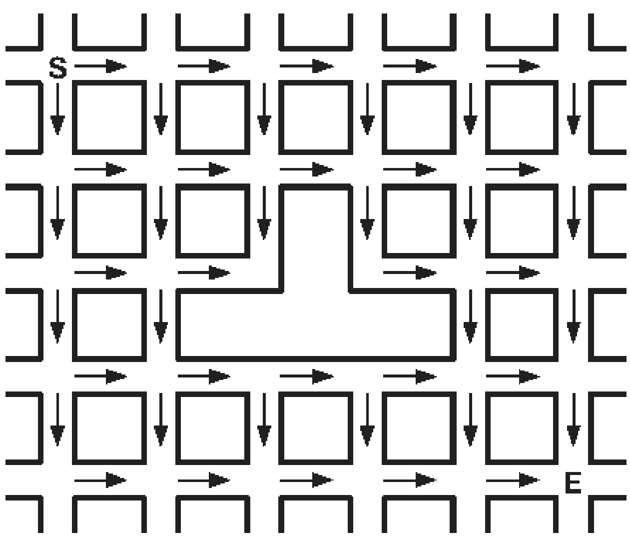

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

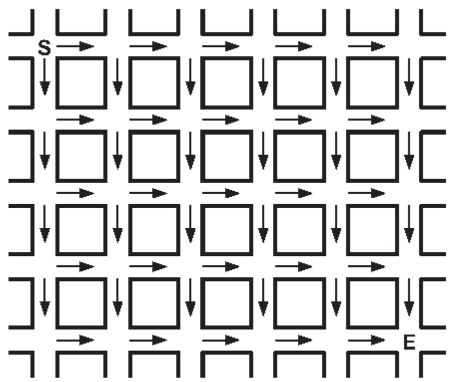

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

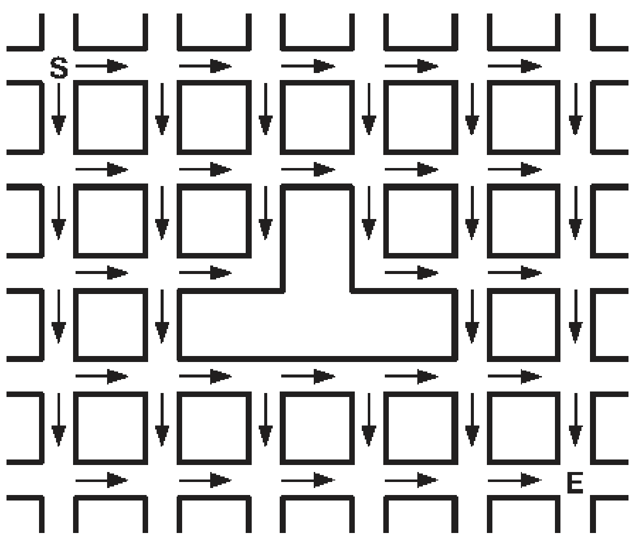

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

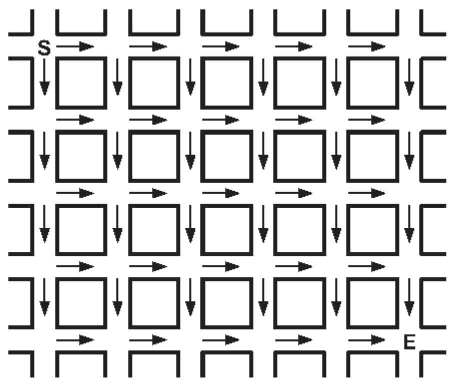

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

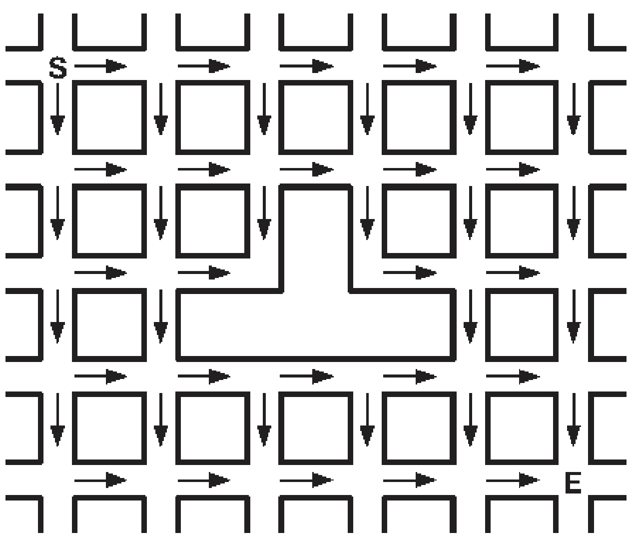

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

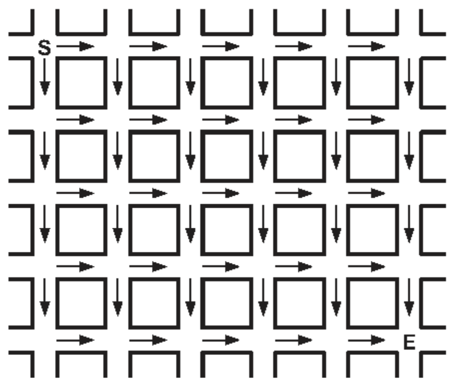

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

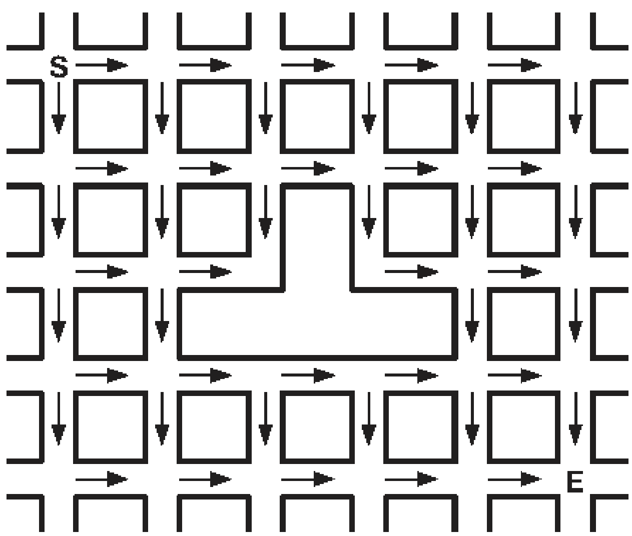

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

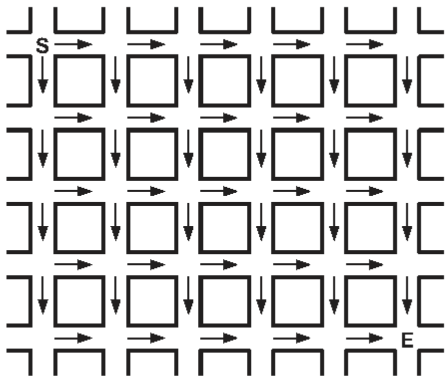

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

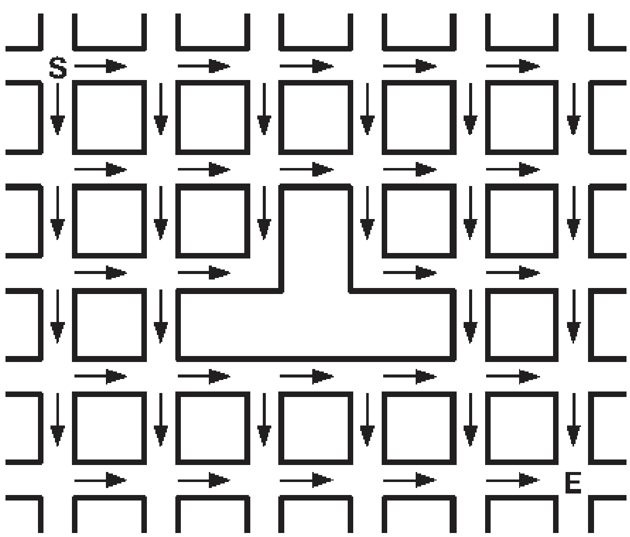

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

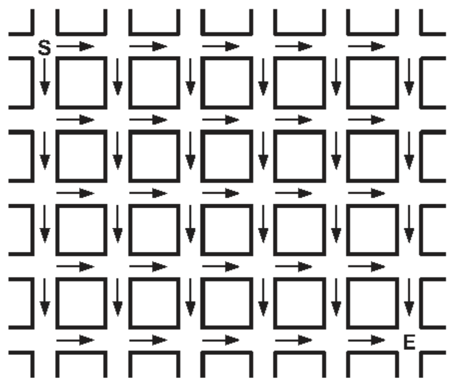

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

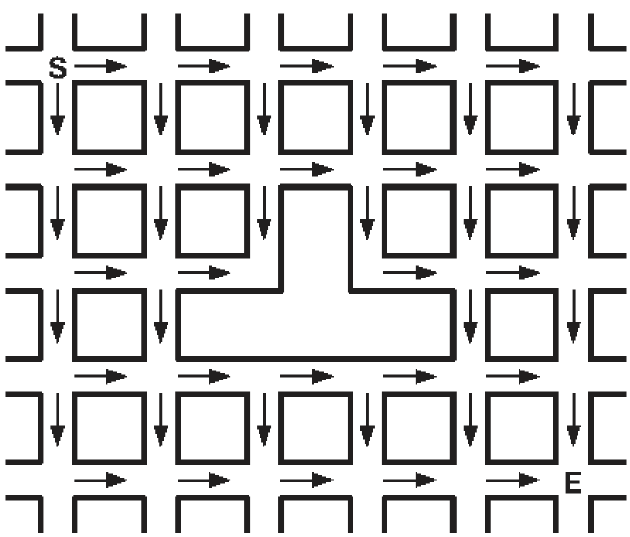

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and $k = 6$, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

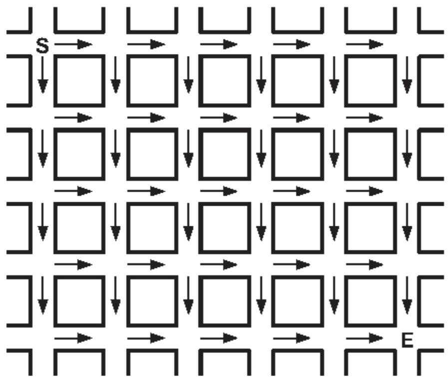

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

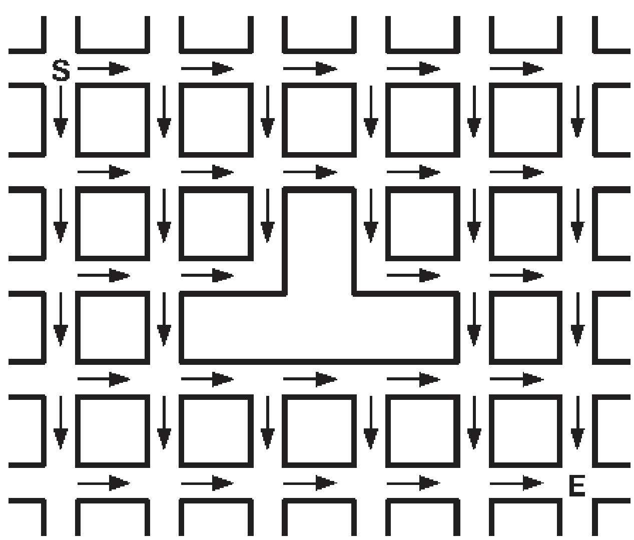

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

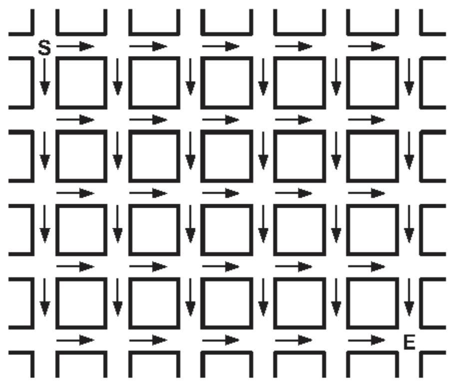

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

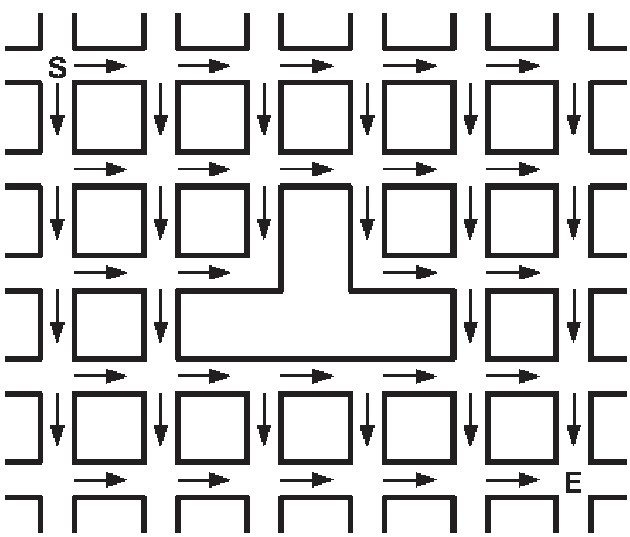

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

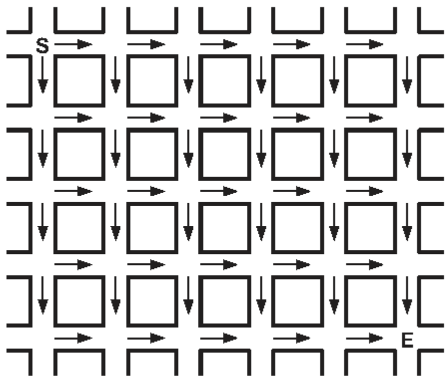

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

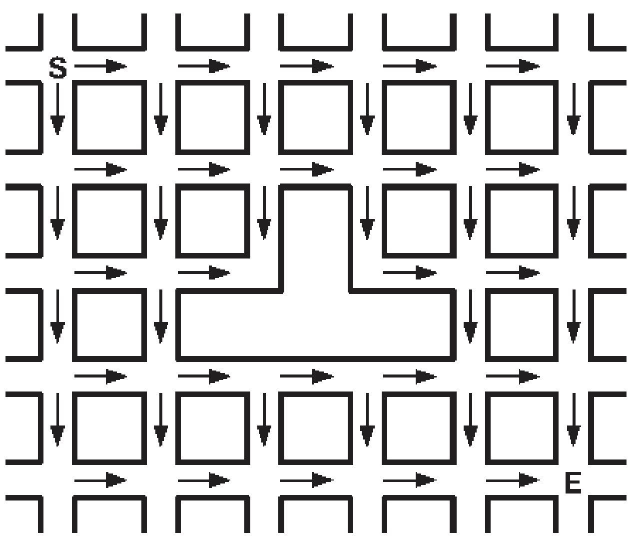

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

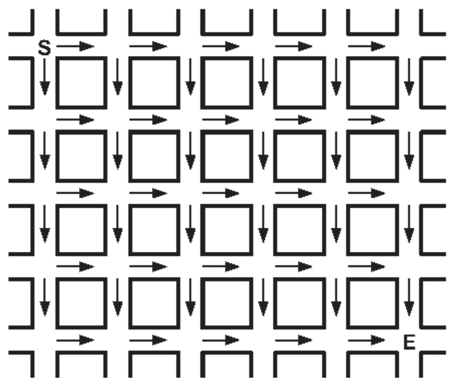

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

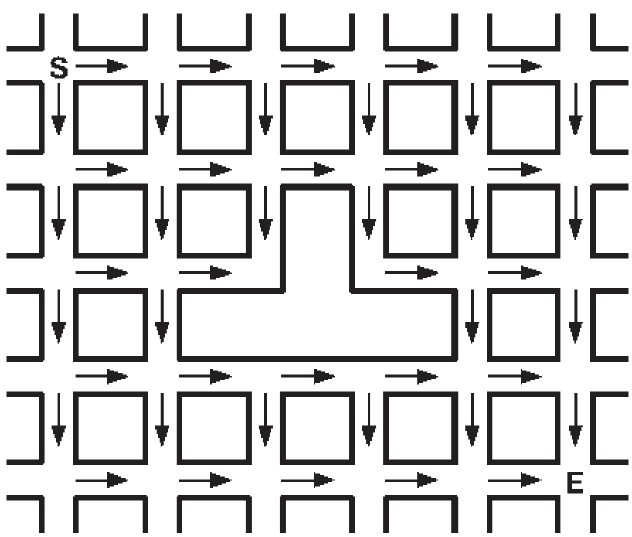

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

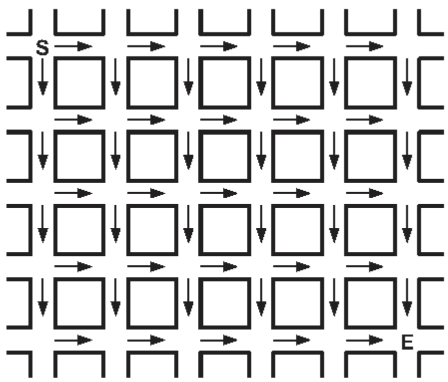

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

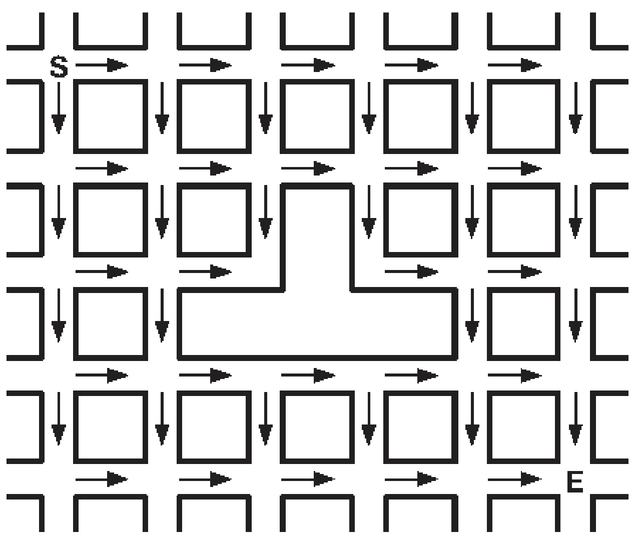

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and $k = 6$, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

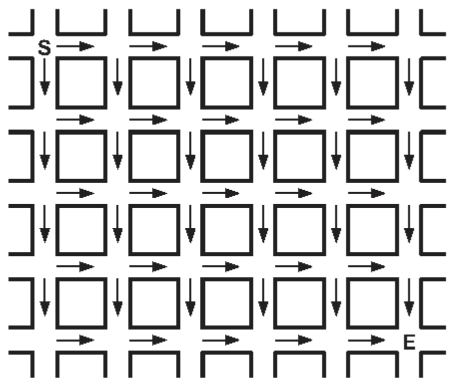

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

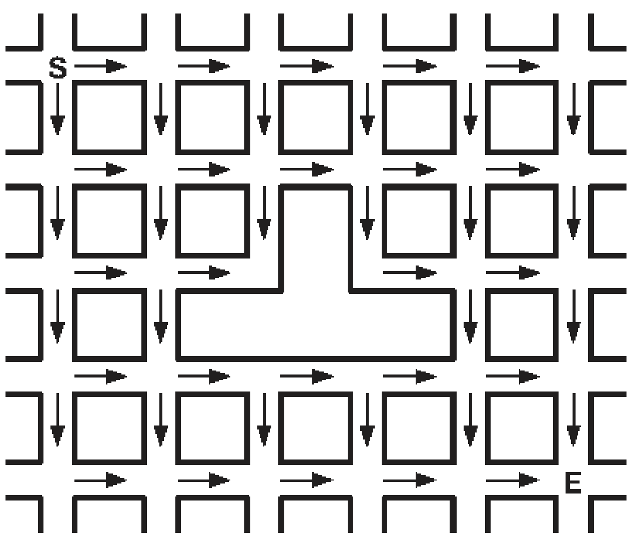

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts
Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.
Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

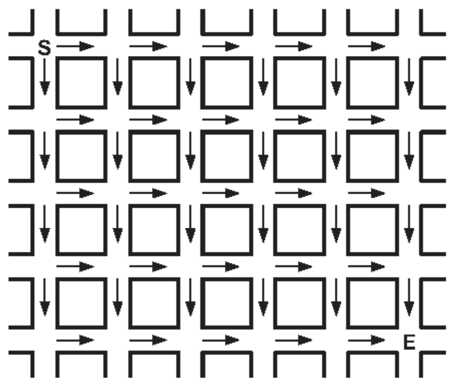

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

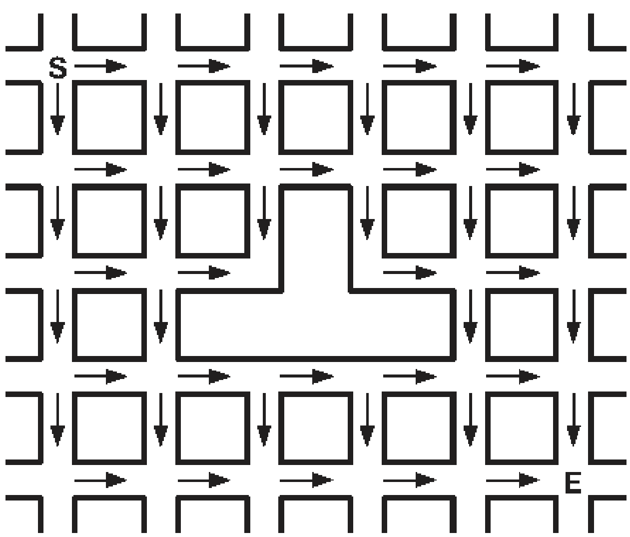

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

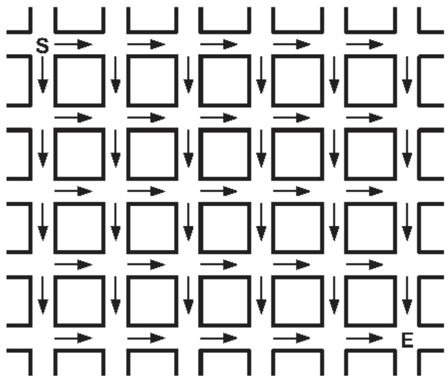

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

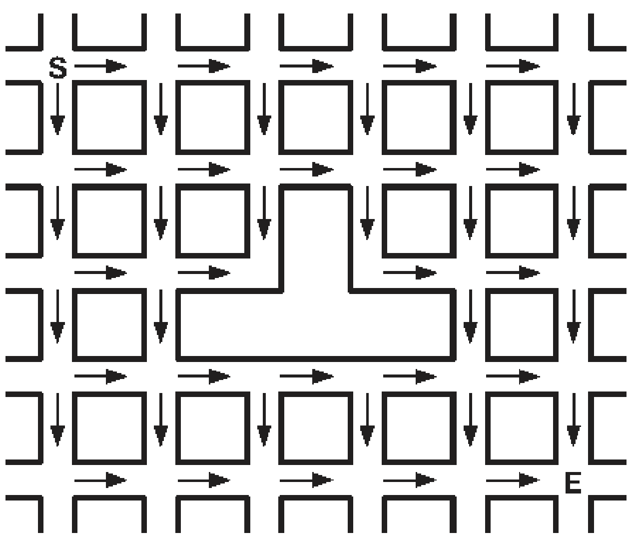

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

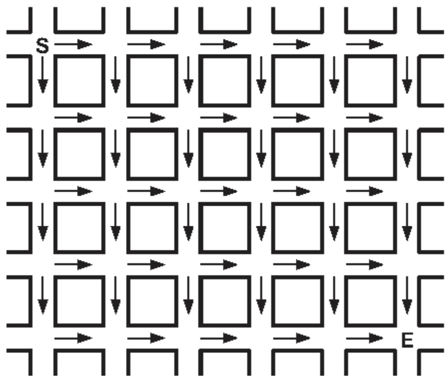

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

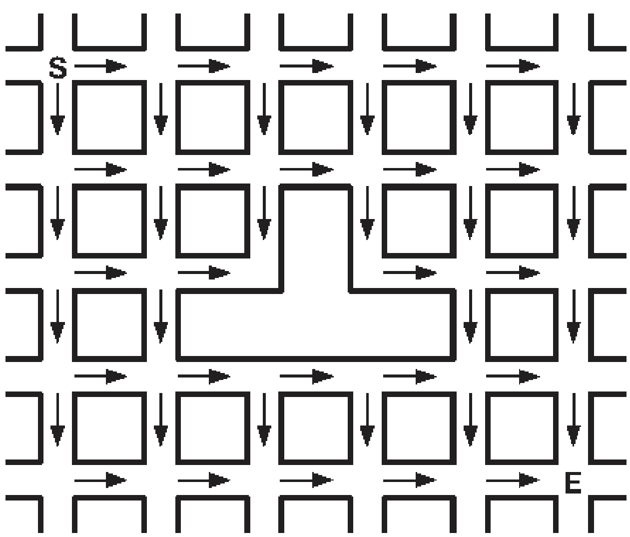

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and $k = 6$, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

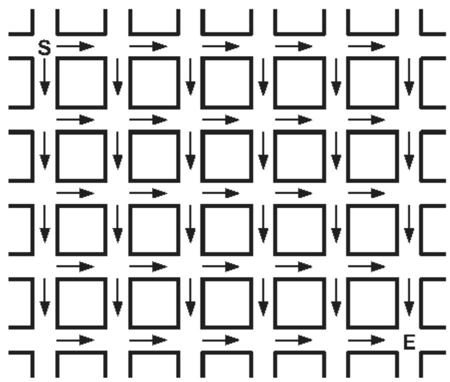

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

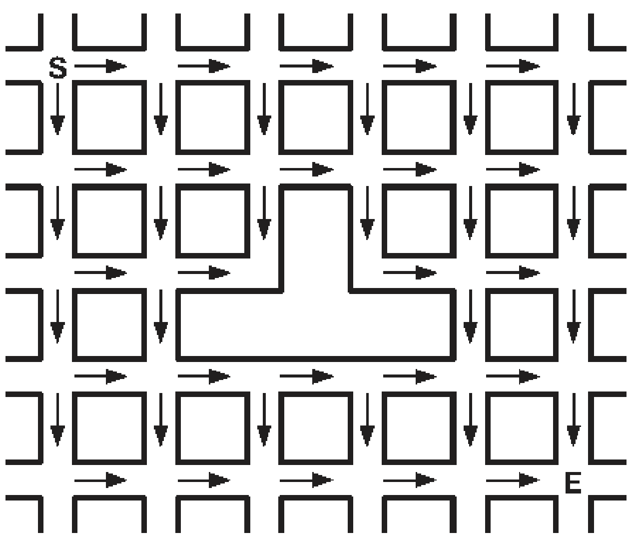

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

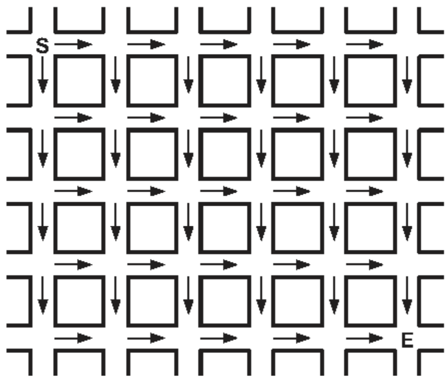

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

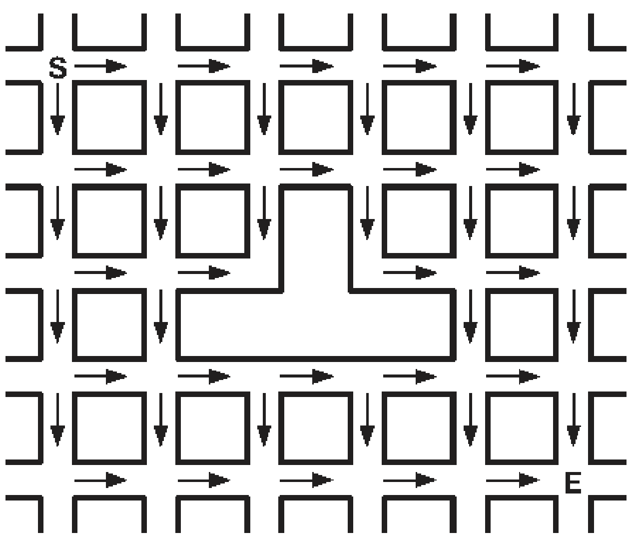

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

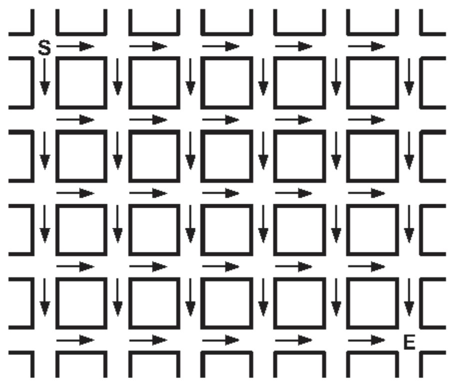

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

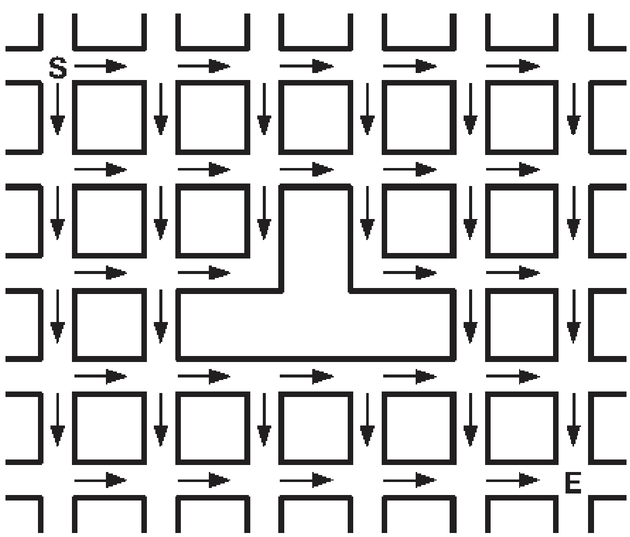

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

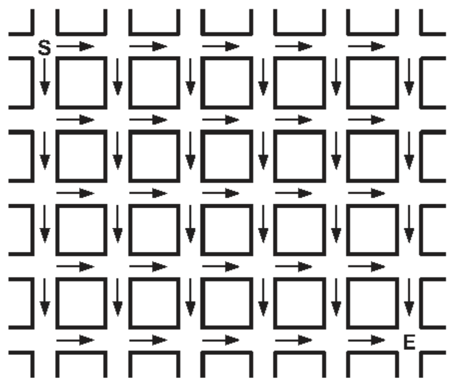

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

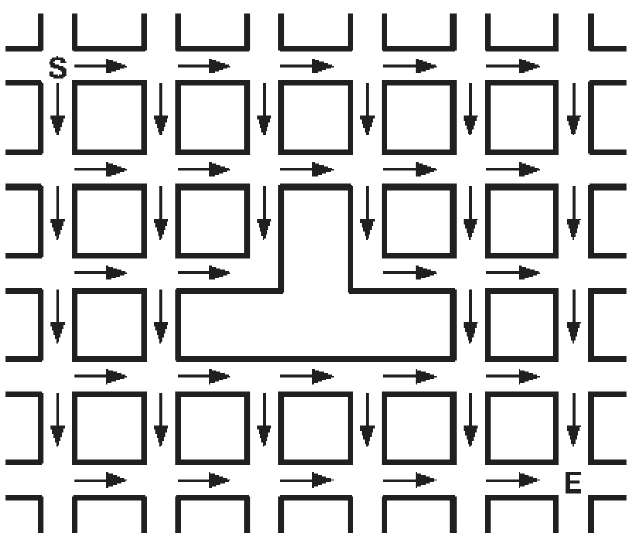

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

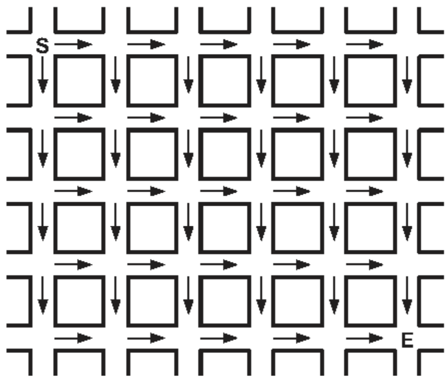

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

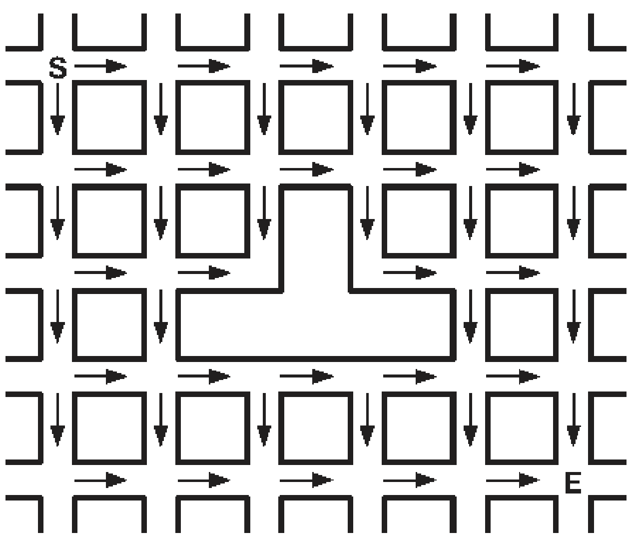

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

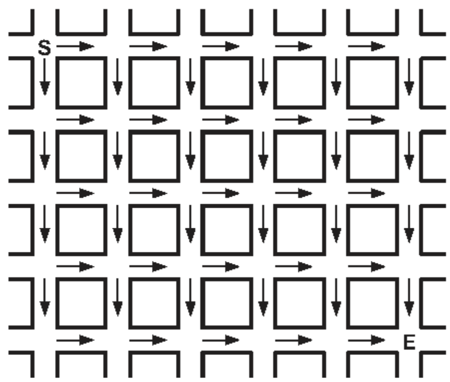

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

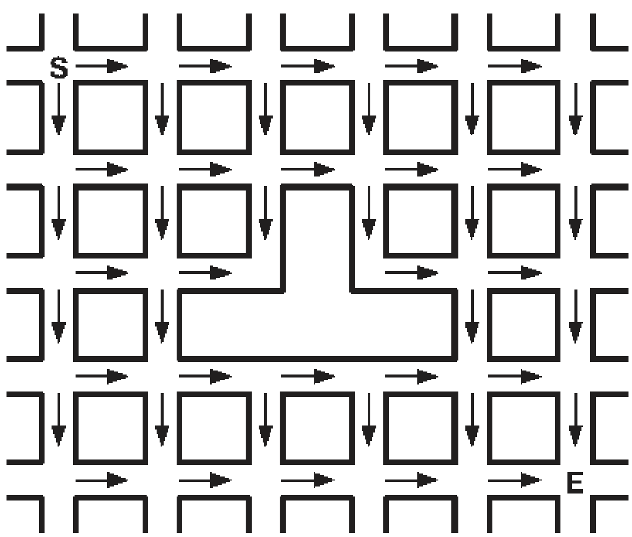

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

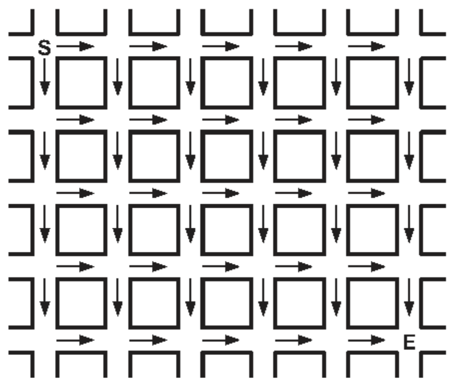

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

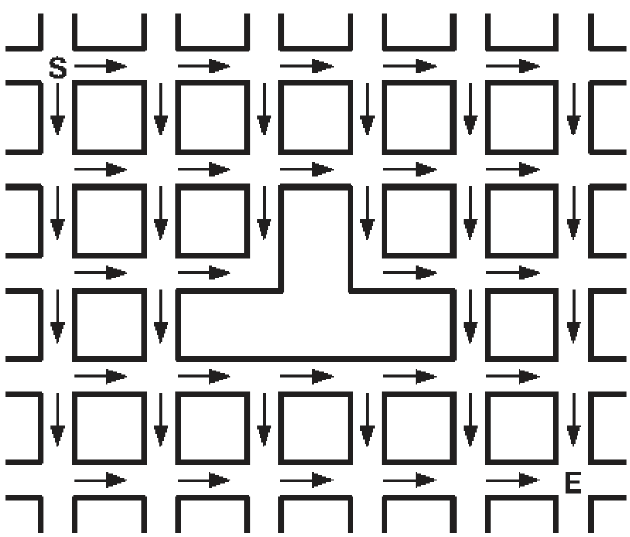

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

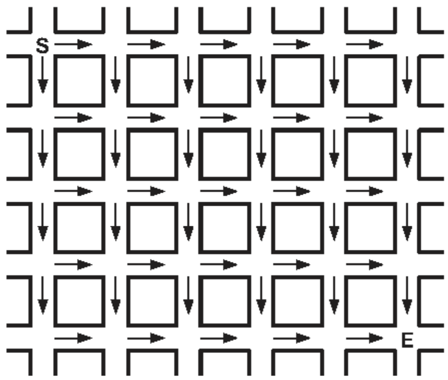

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

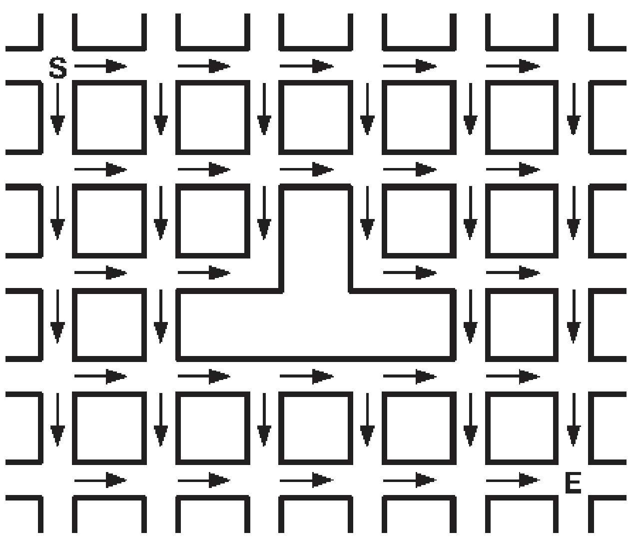

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

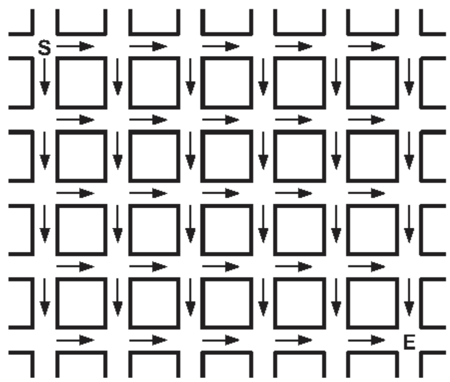

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

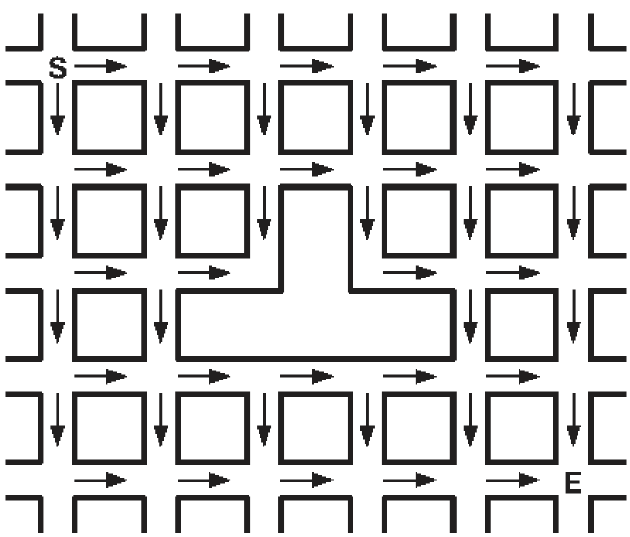

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

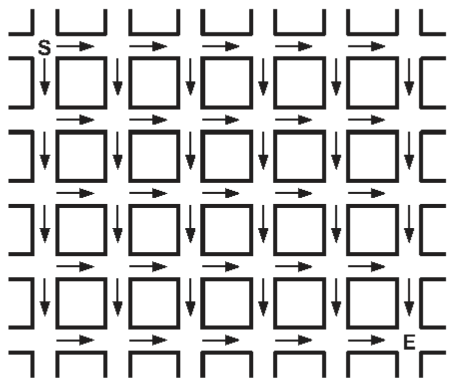

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

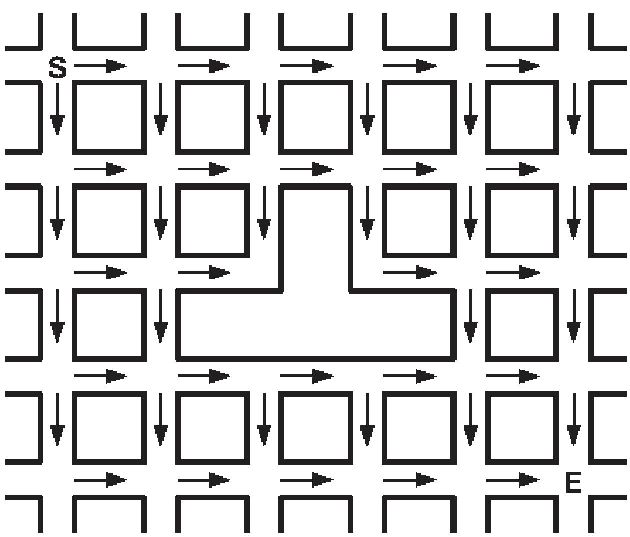

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

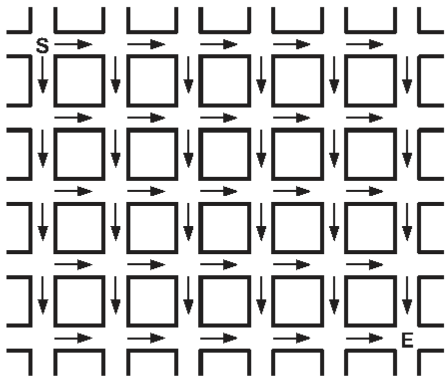

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

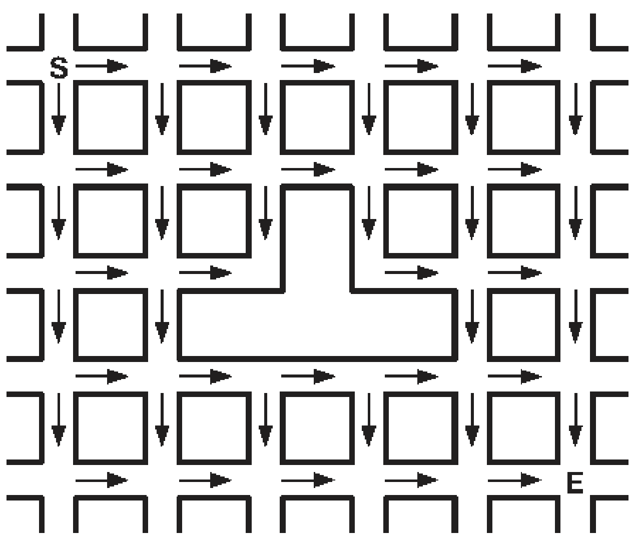

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

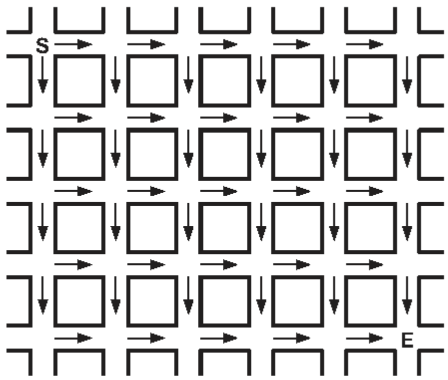

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

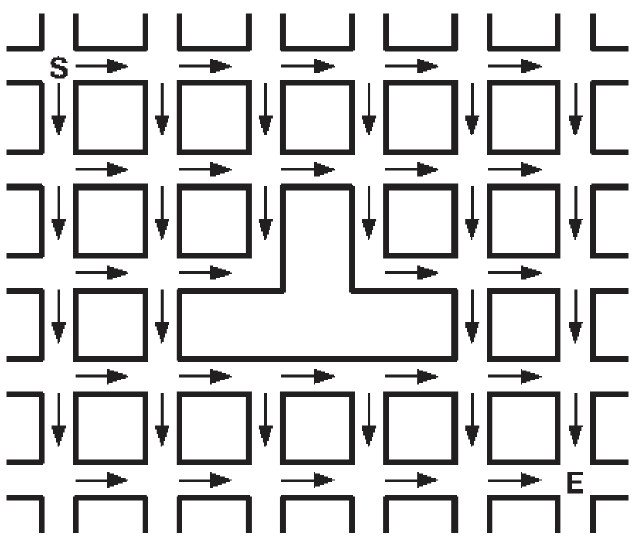

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

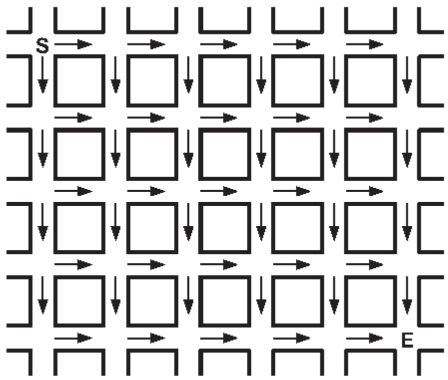

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

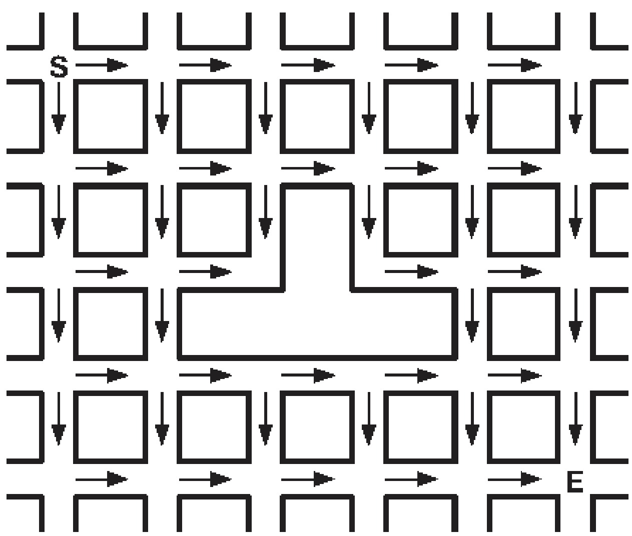

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

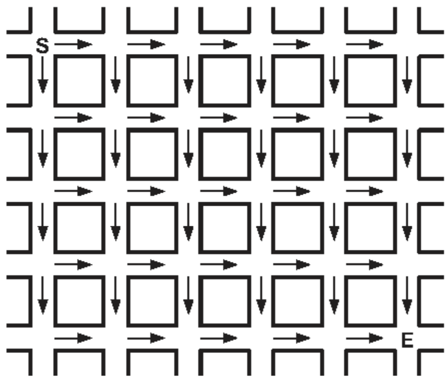

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

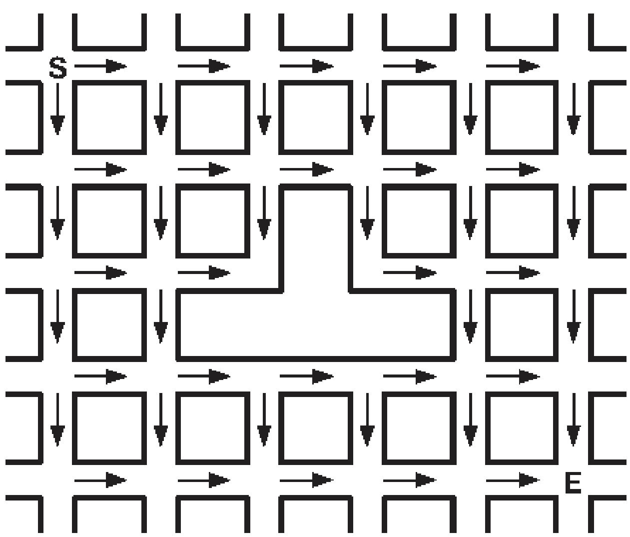

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array $A[1...n]$ of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most $O(\log n)$ pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

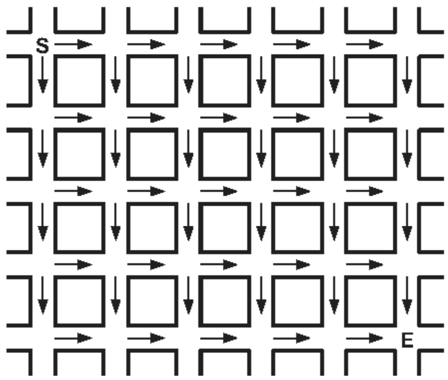

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

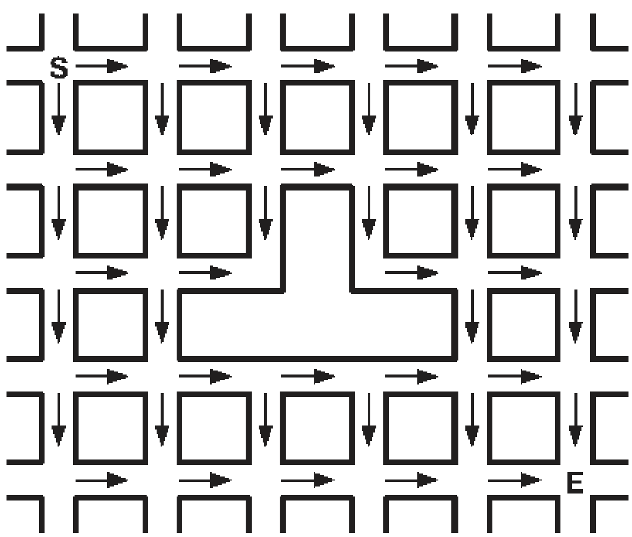

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

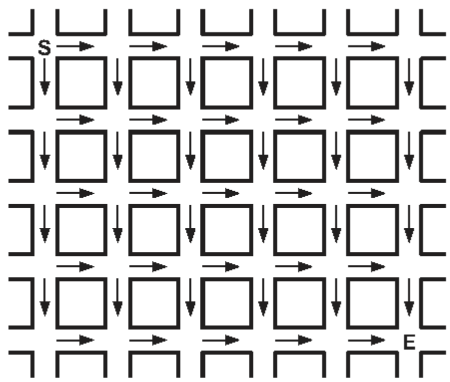

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

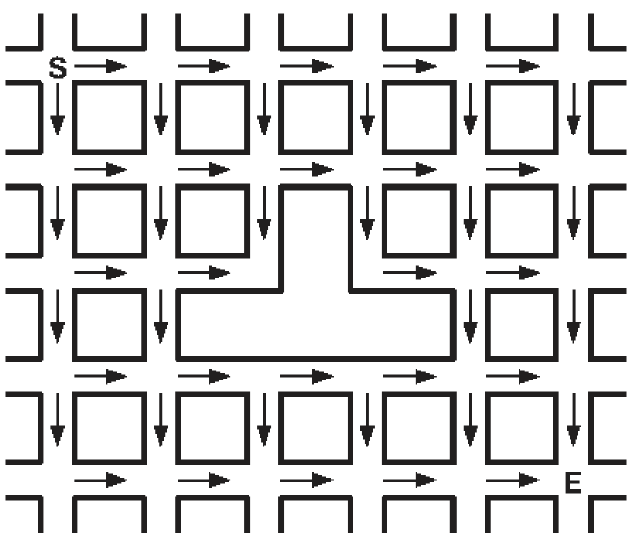

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

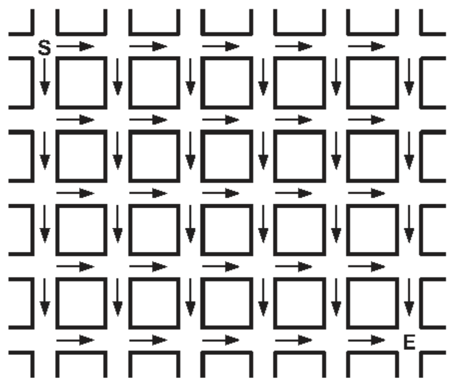

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

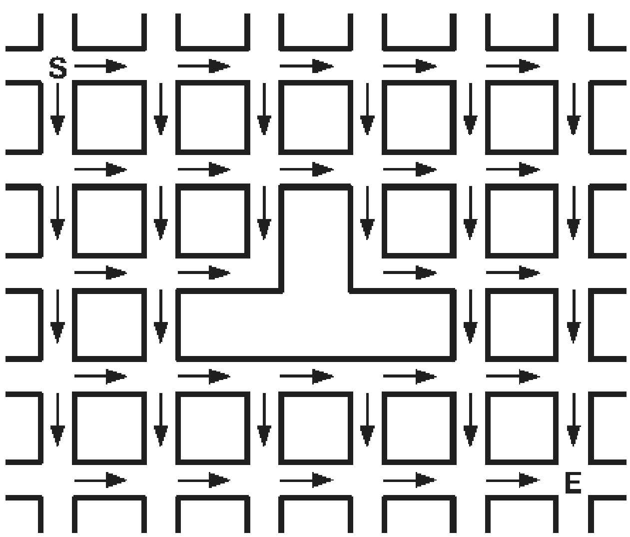

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

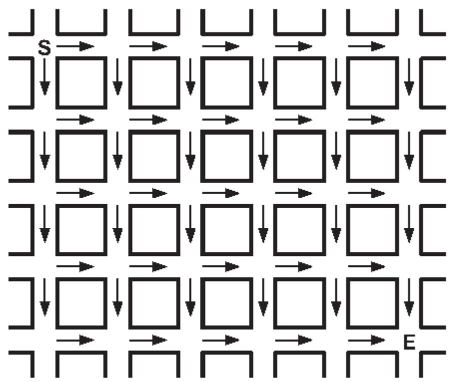

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

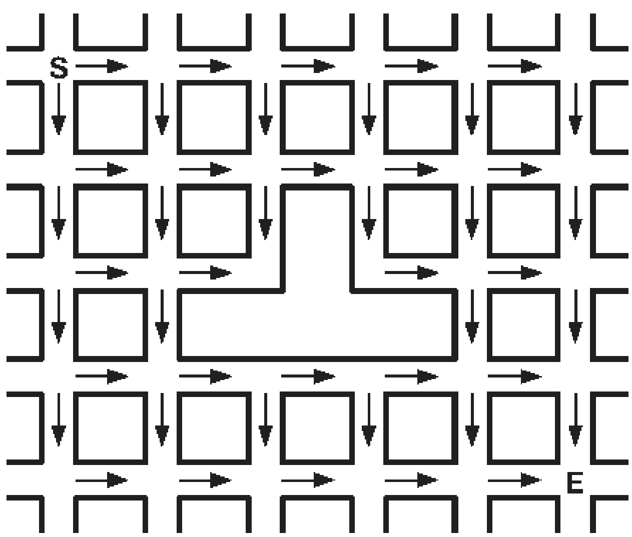

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

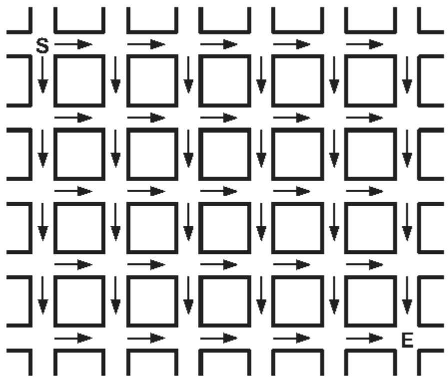

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

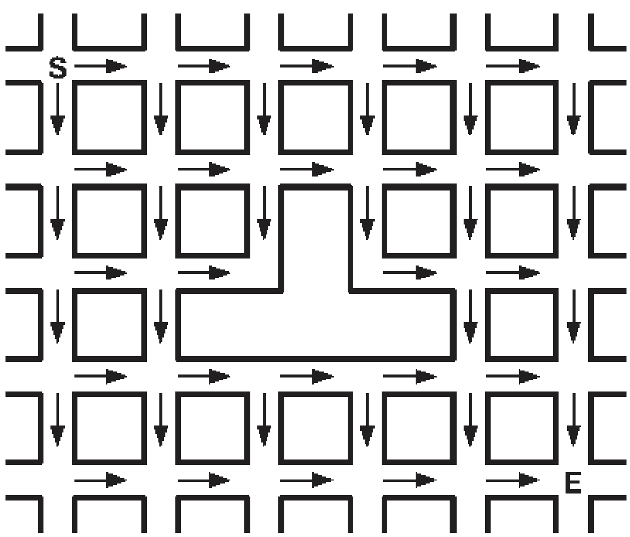

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: _______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

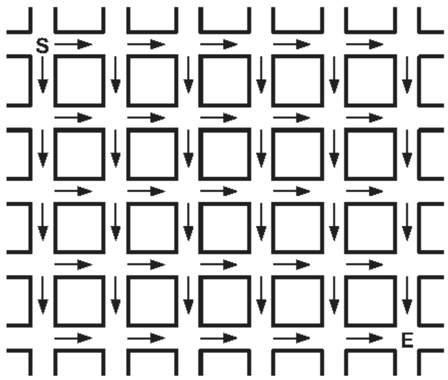

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

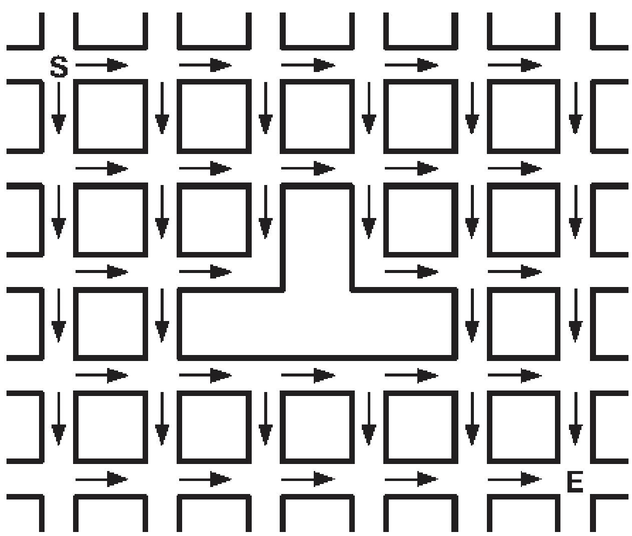

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms

Spring 2014

Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

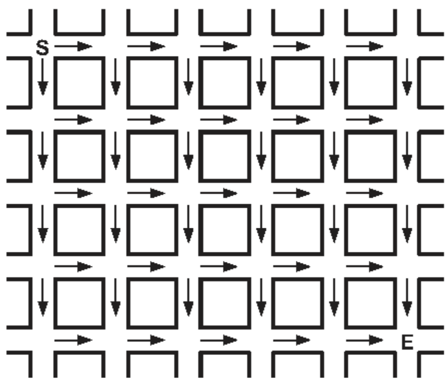

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

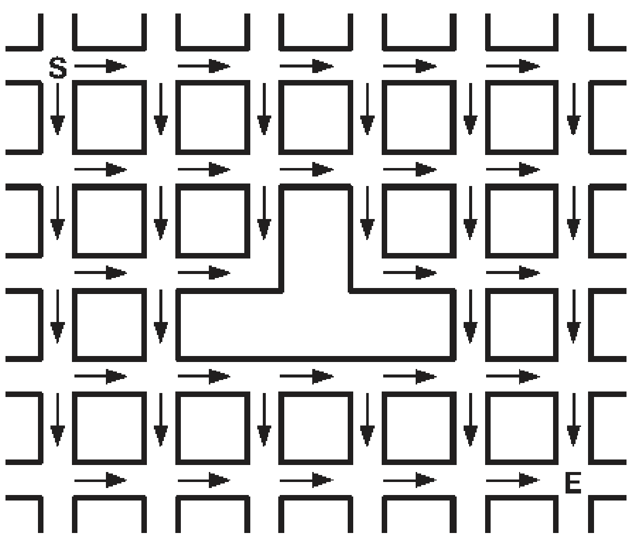

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ____________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

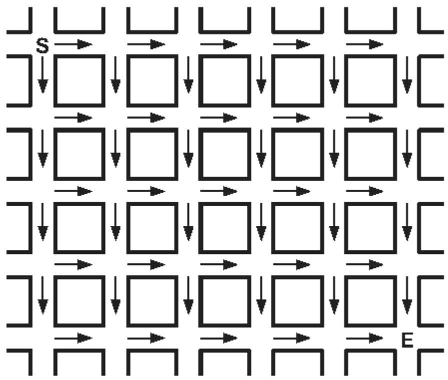

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

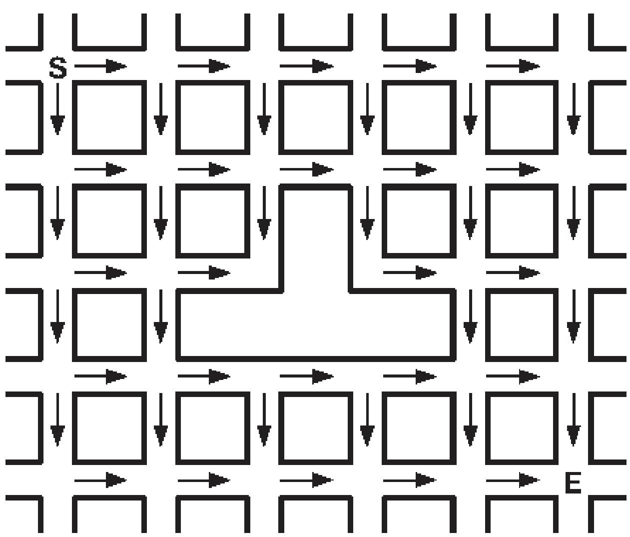

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

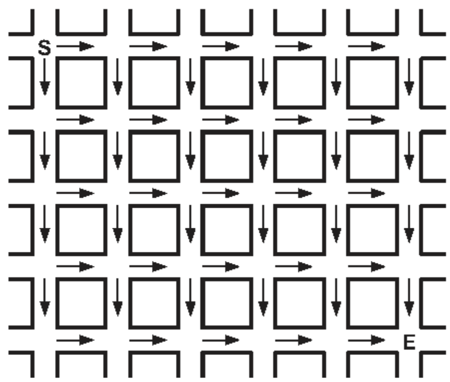

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

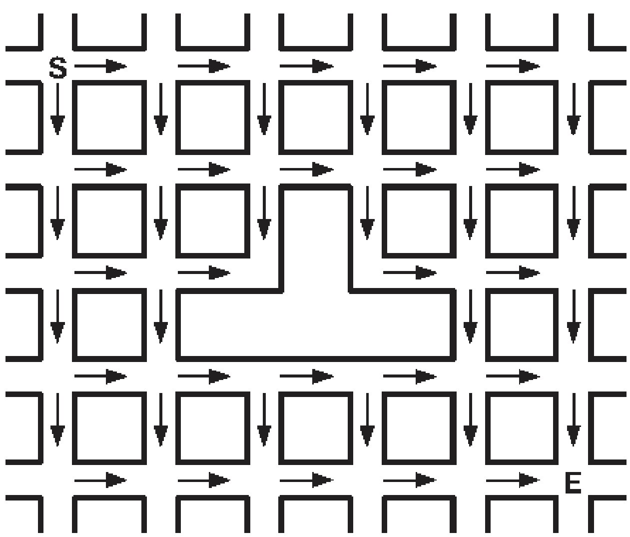

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

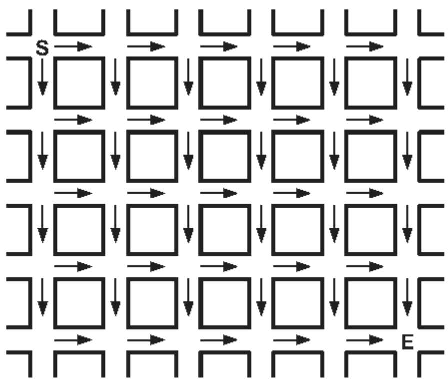

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

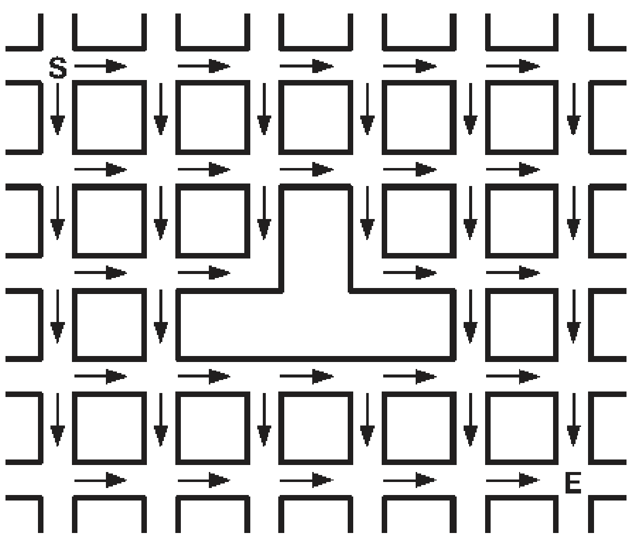

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

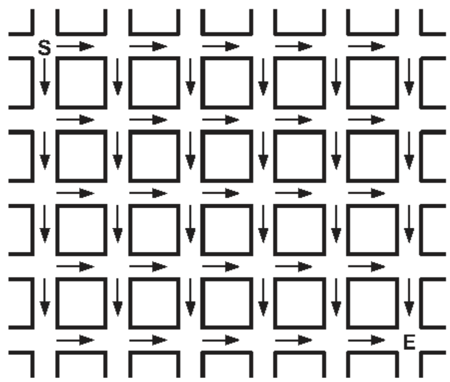

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

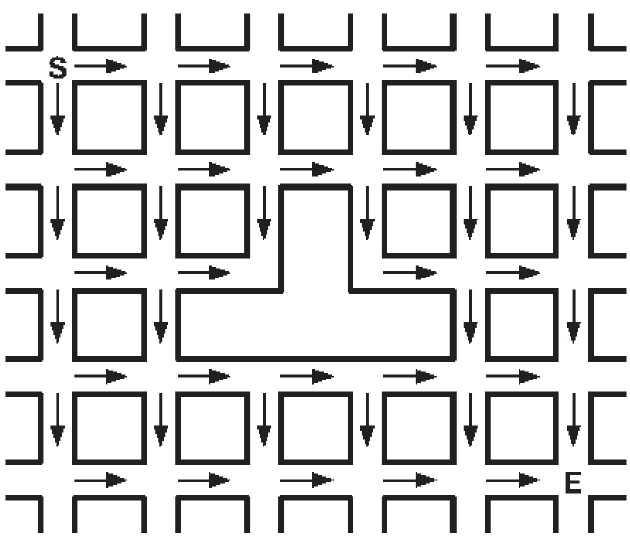

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

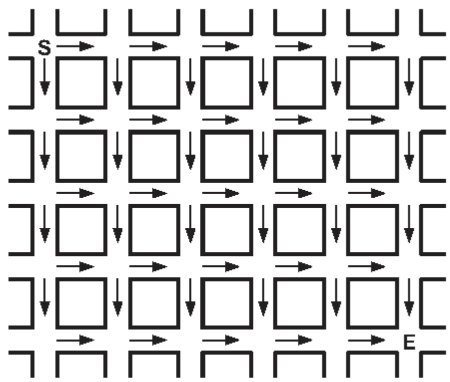

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

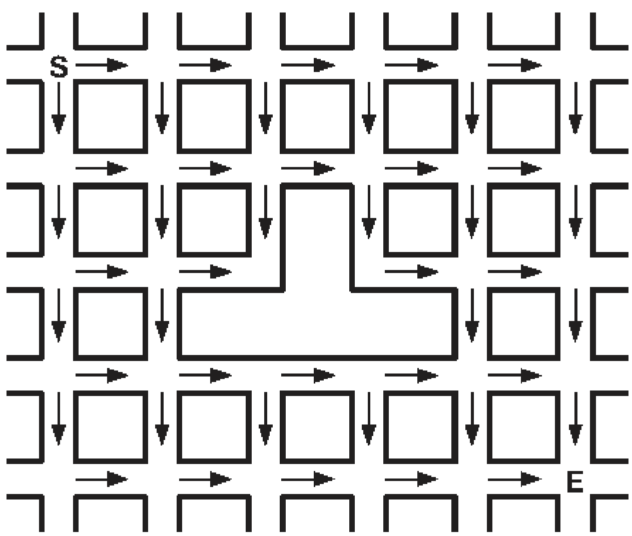

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts
You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

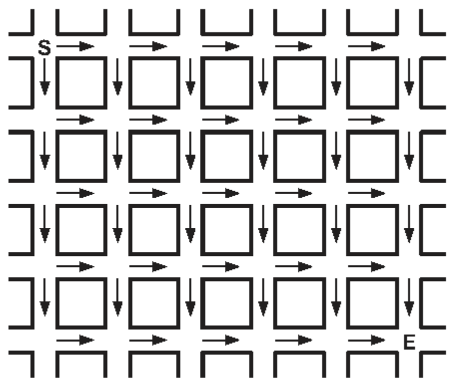

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

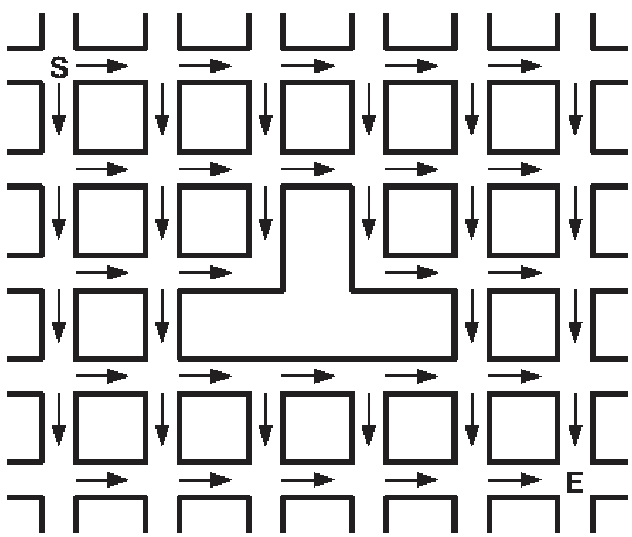

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

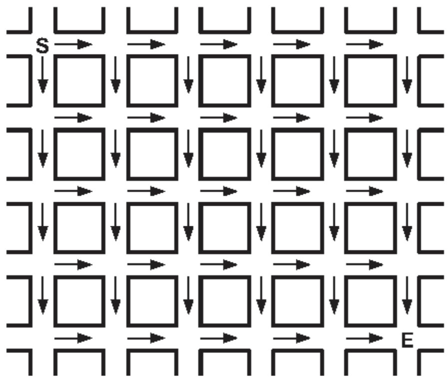

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

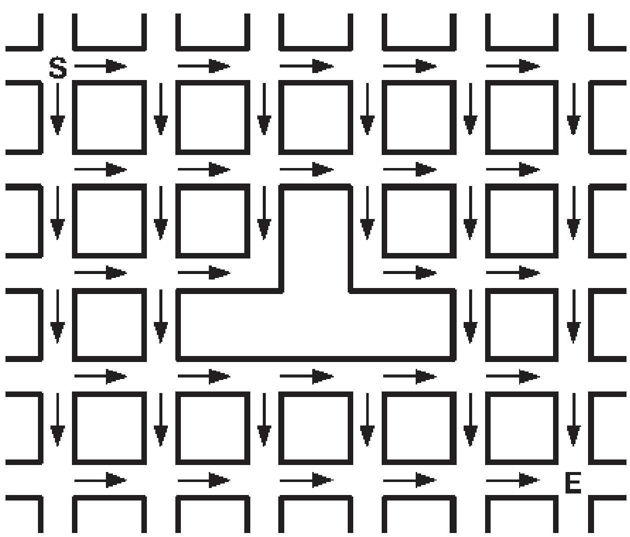

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that $n>2$, $A[1]>=A[2]$ and $A[n]>=A[n-1]$. A local minimum of the array A is defined as an index i such that $1<i<n$, $A[i-1]>=A[i]$ and $A[i+1]>=A[i]$.

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

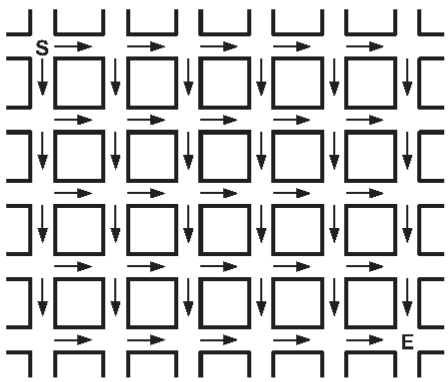

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

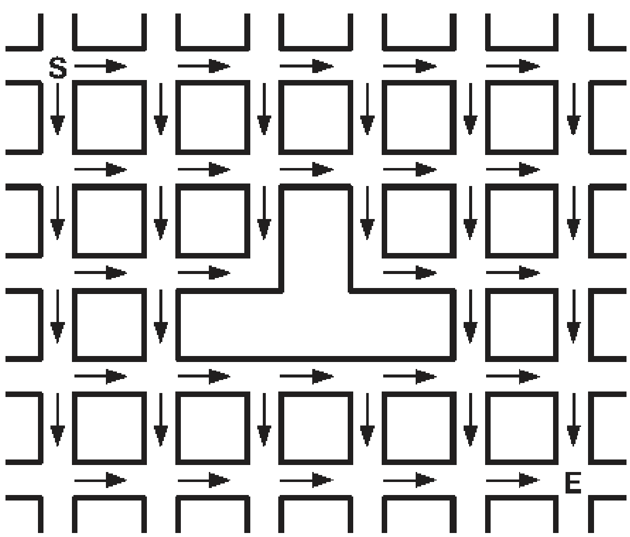

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

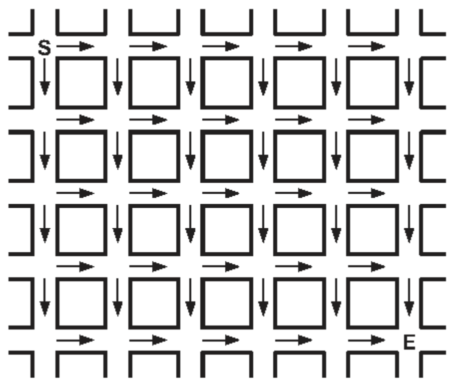

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

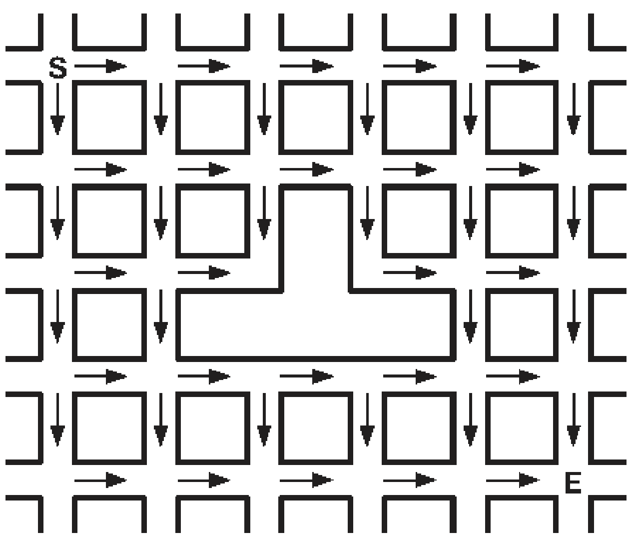

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

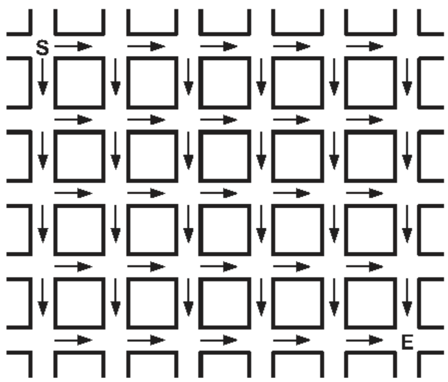

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

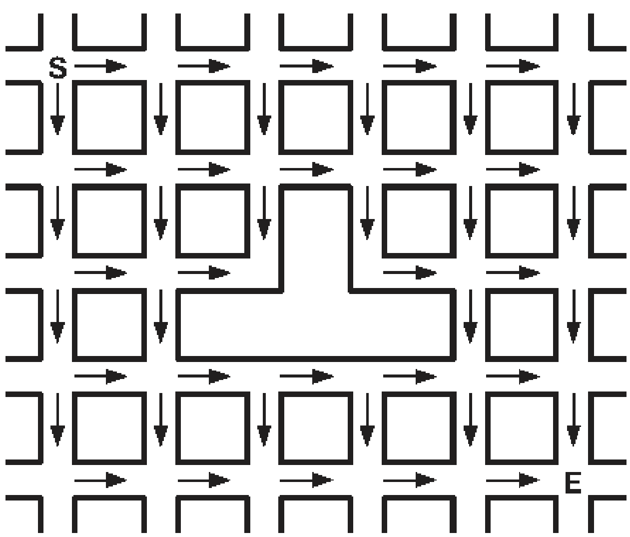

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

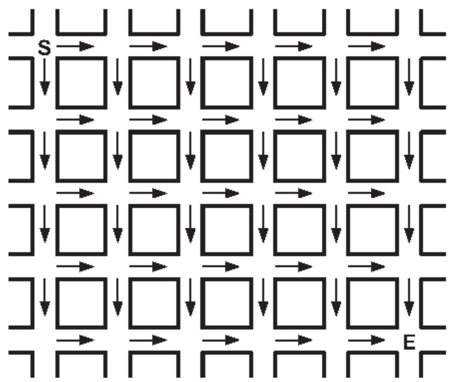

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

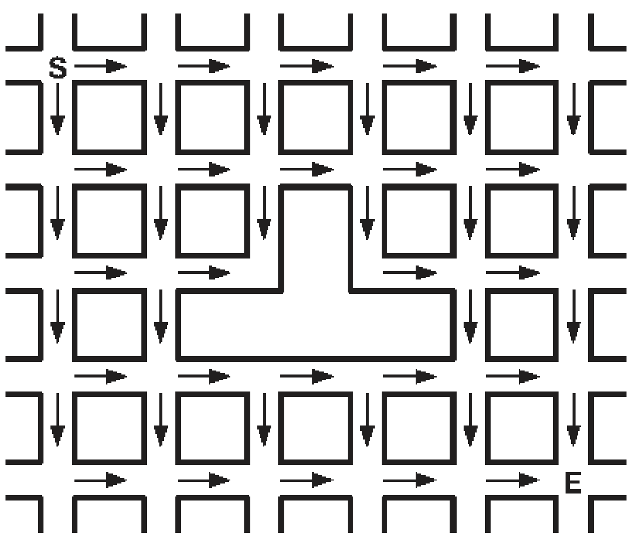

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ___________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

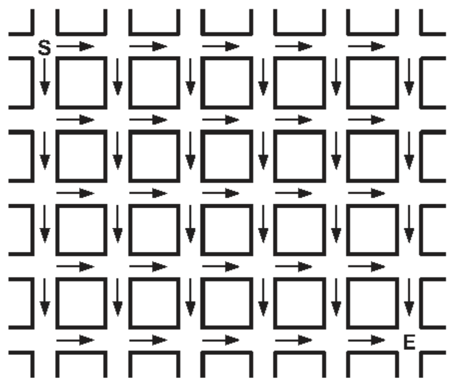

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

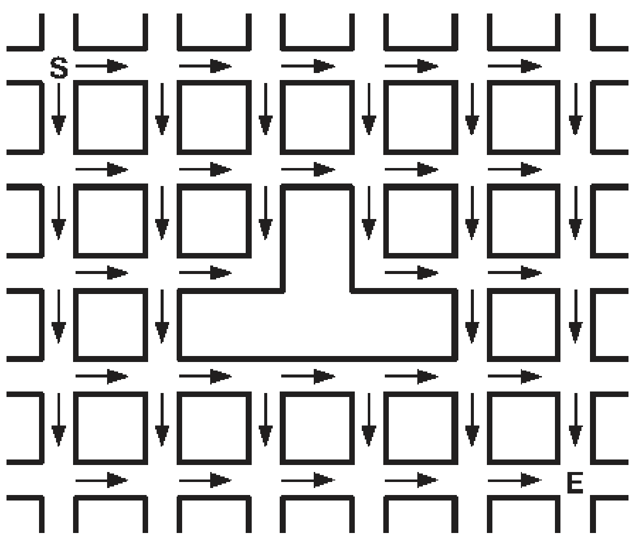

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

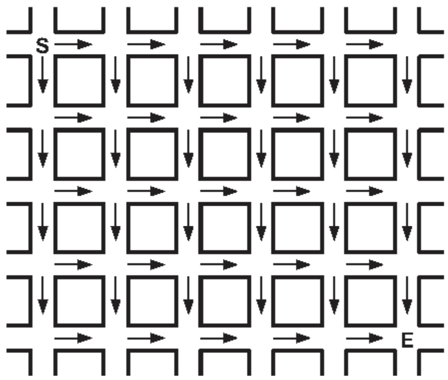

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

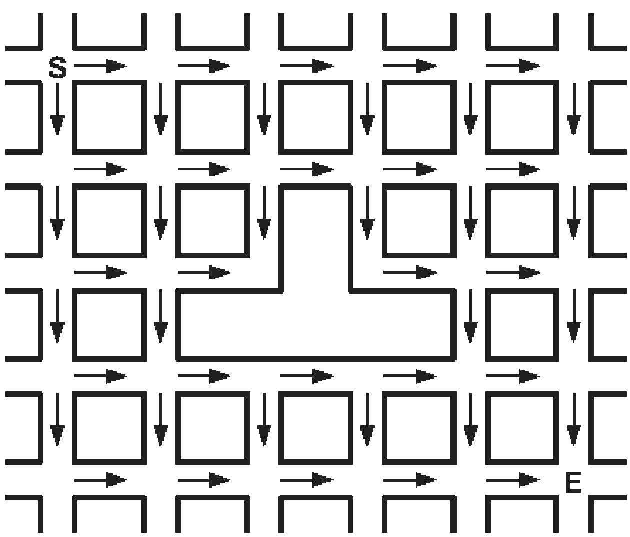

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

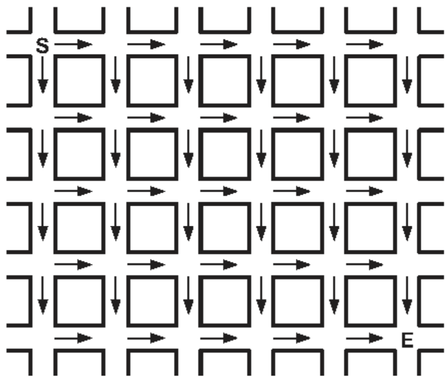

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

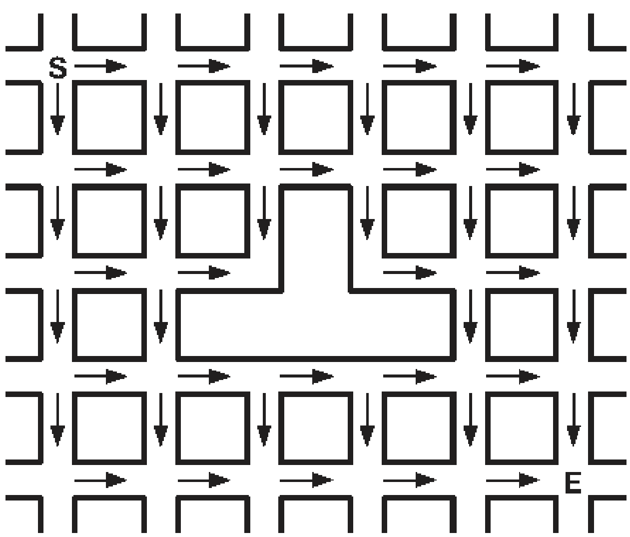

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts
Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.
Input: $x_1, x_2, \ldots, x_n$, k, v.
Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

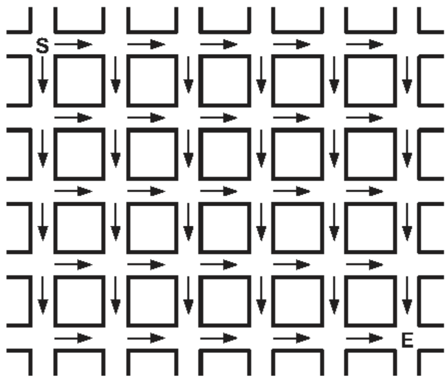

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

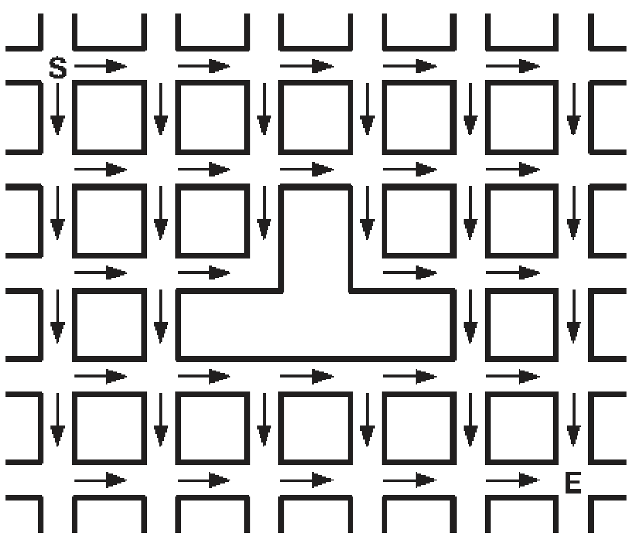

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

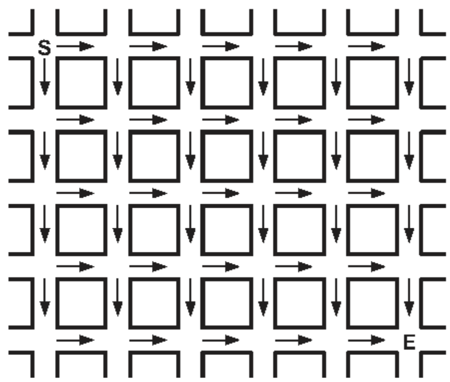

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

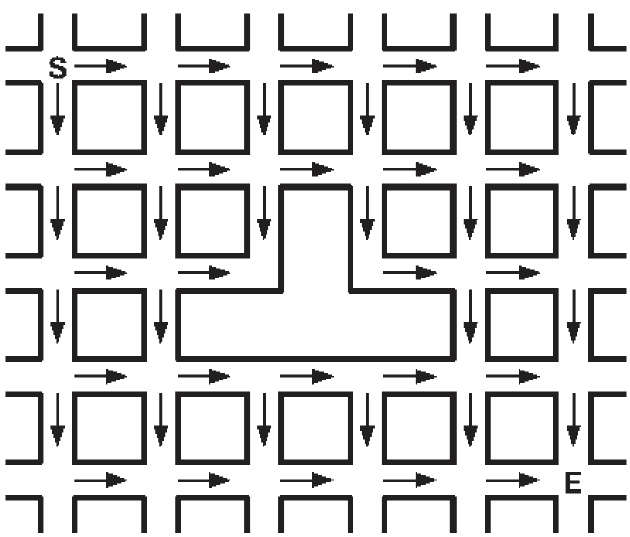

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)+56$, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

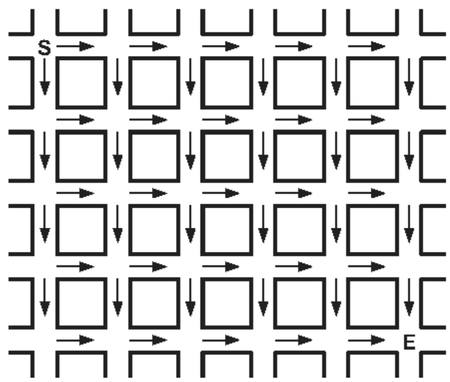

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

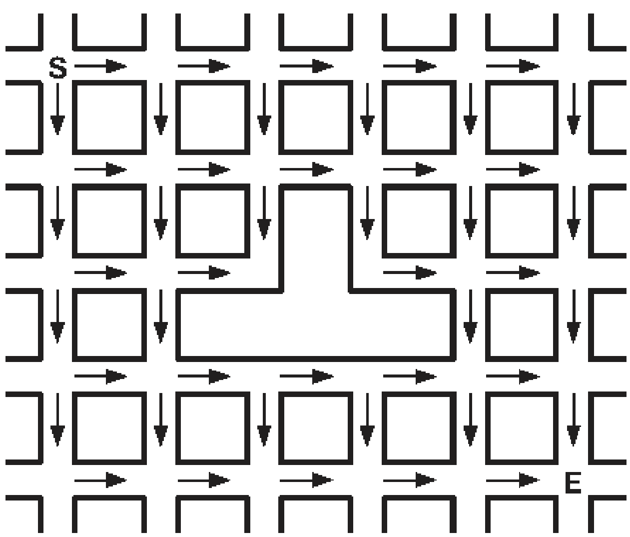

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

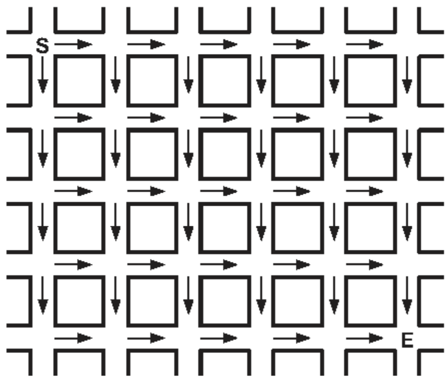

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

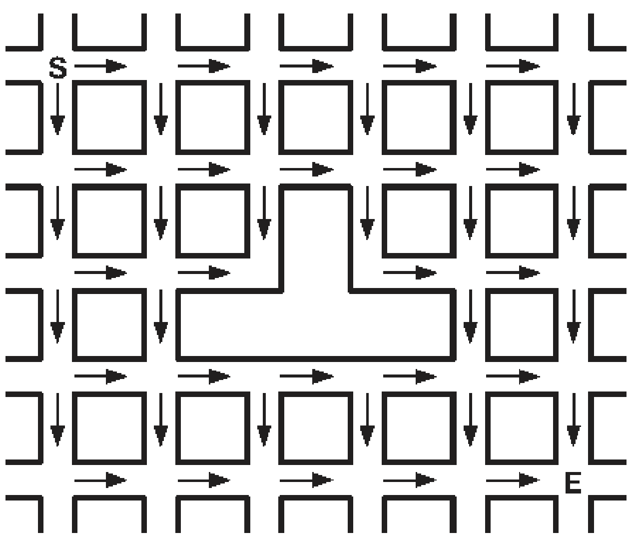

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

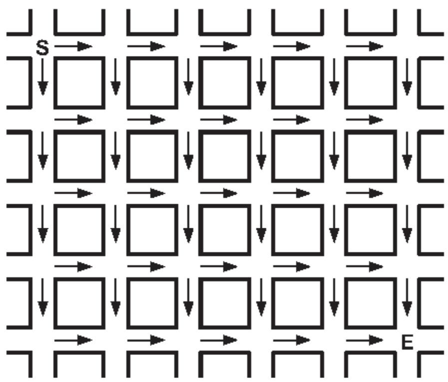

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

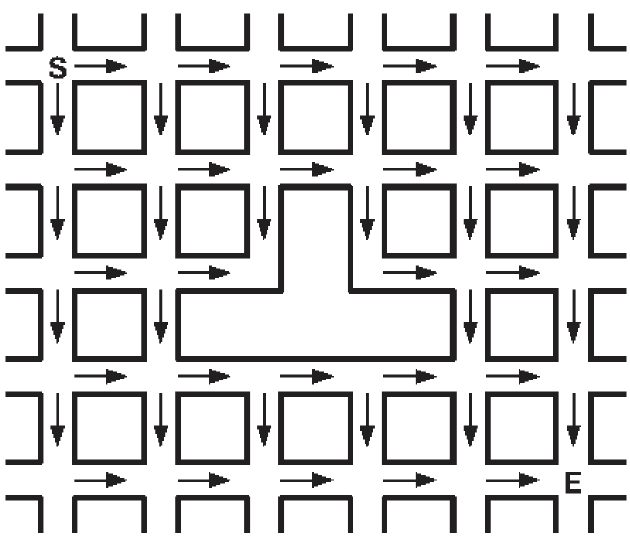

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $S_i \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

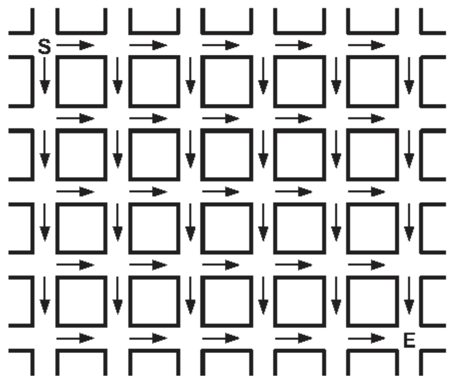

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

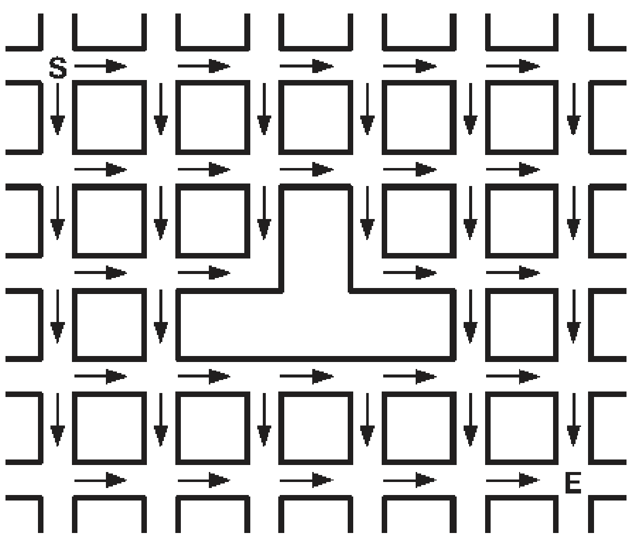

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

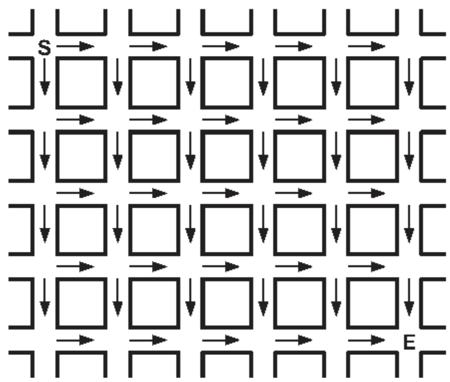

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

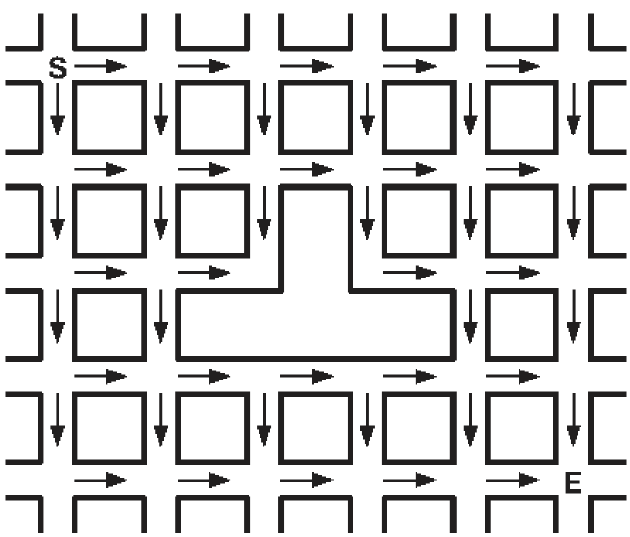

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts

Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]

Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]

Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]

For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]

Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]

In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]

In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]

If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]

Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]

Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]

The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

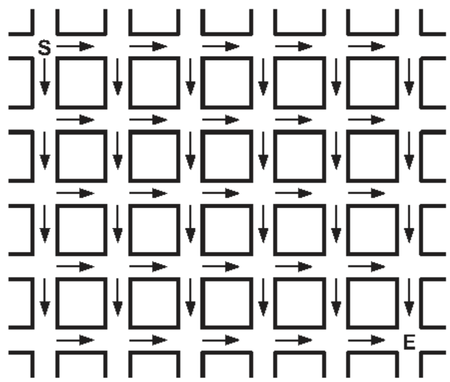

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

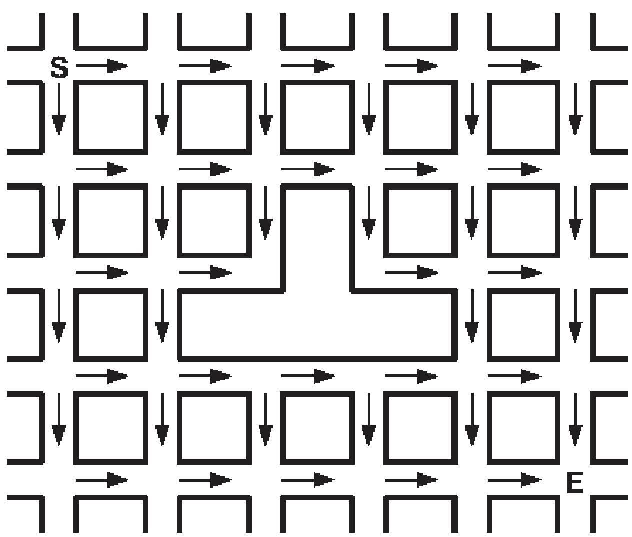

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

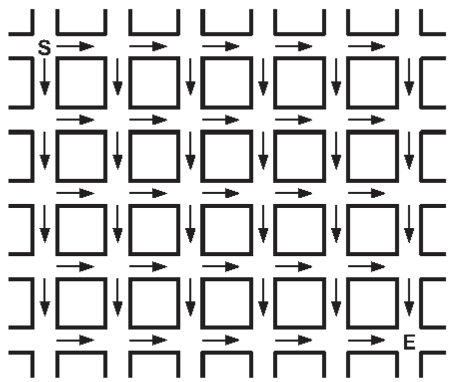

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

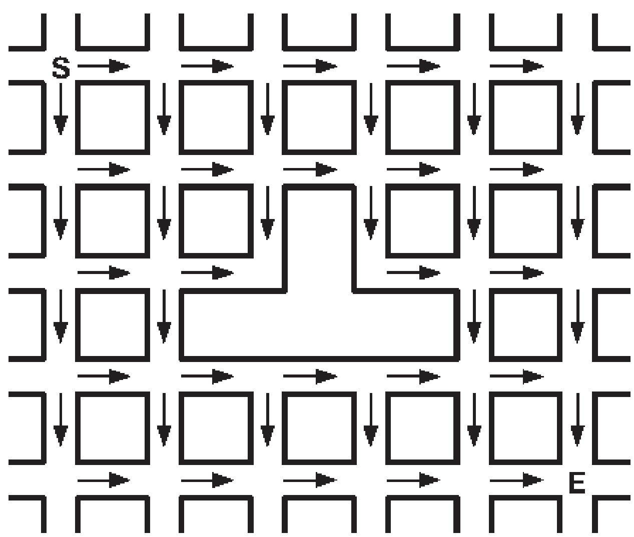

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

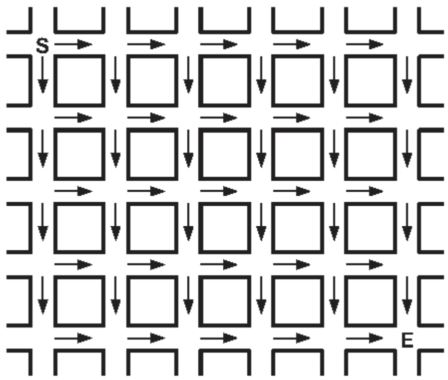

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

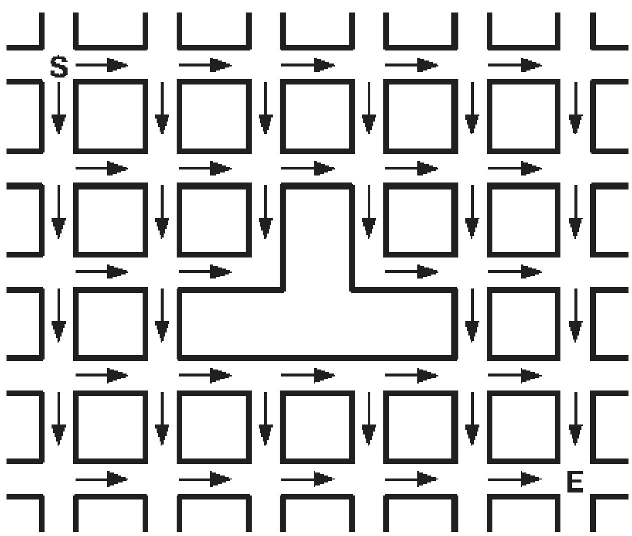

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ________________________
Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations x_1, x_2, …, x_n, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: x_1, x_2, …, x_n, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations x_1, x_2, …, x_n? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

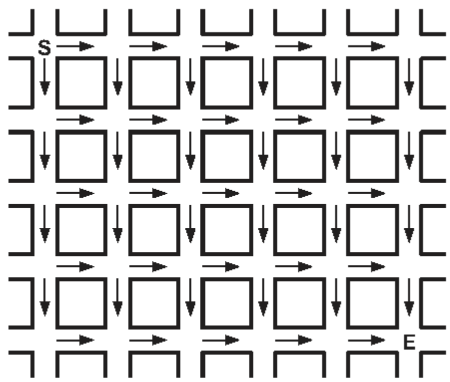

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

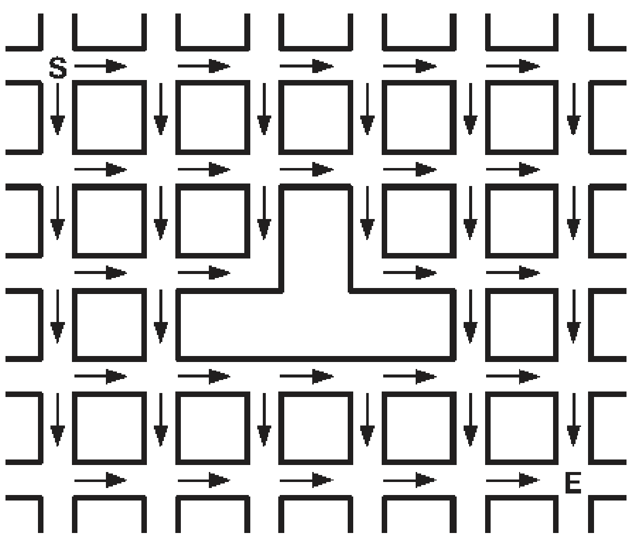

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts
The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

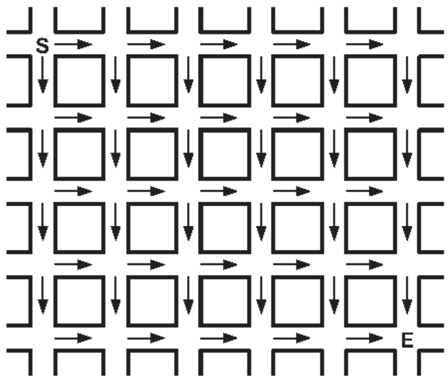

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to “smaller” subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

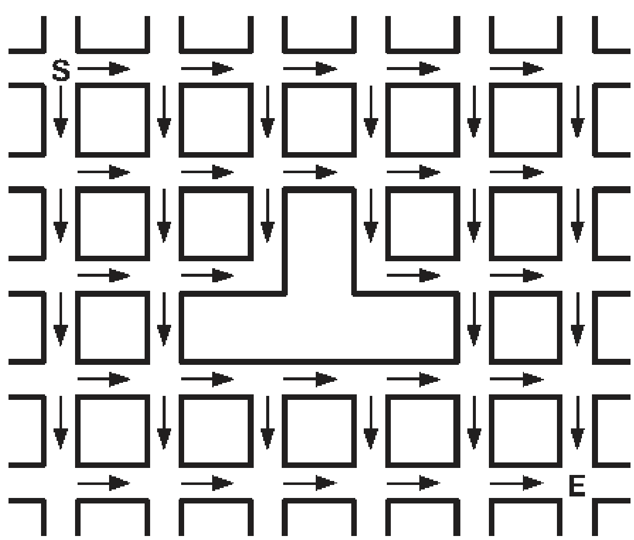

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________
Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

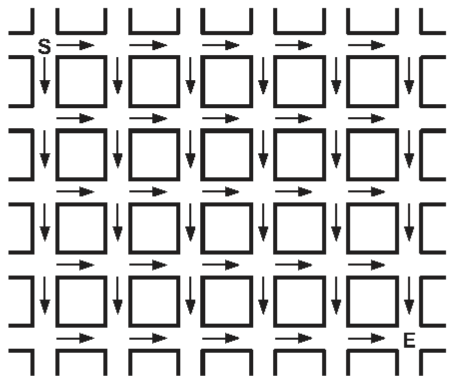

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

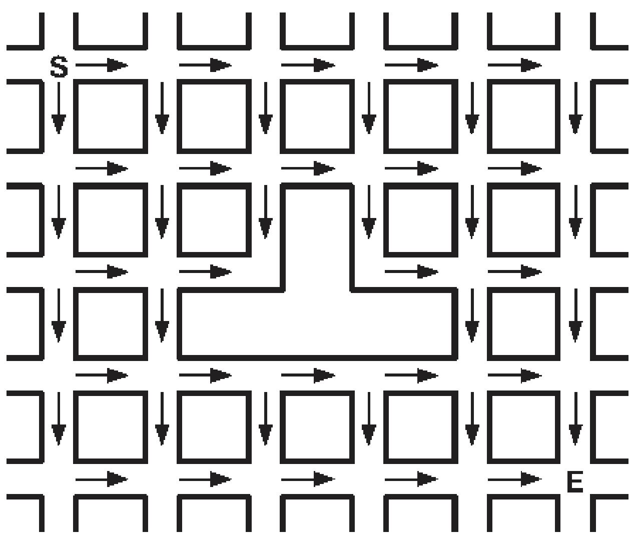

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570

Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: ____________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights Si $\subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

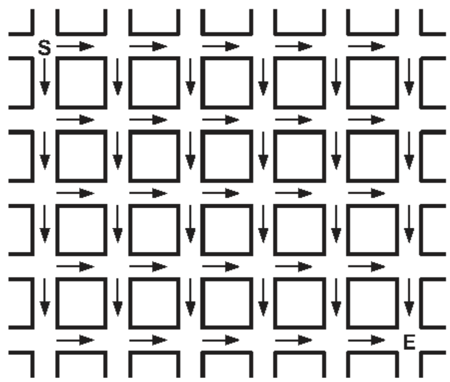

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

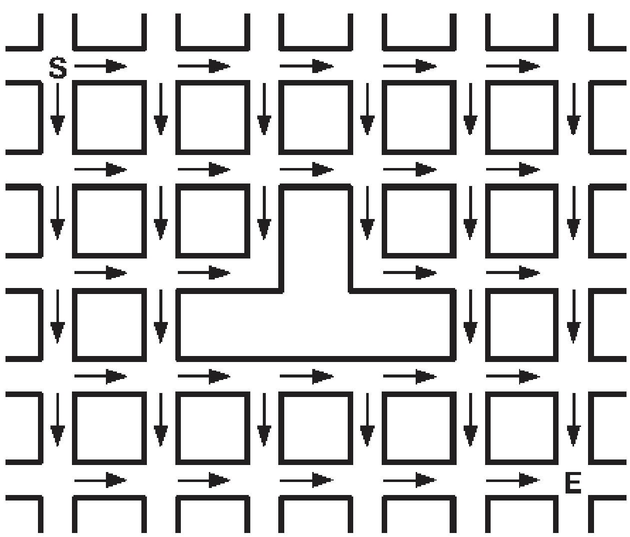

Figure B.

6) 15 pts

A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space

CS570
Analysis of Algorithms
Spring 2014
Exam II

Name: ______________________

Student ID: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	15	
Problem 4	20	
Problem 5	15	
Problem 6	15	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$

[TRUE/FALSE]
Maximum value of an s-t flow could be less than the capacity of a given s-t cut in a flow network.

[TRUE/FALSE]
For any edge e that is part of the minimum cut in G, if we increase the capacity of that edge by any integer k>1, then that edge will no longer be part of the minimum cut.

[TRUE/FALSE]
Any problem that can be solved using dynamic programming has a polynomial worst case time complexity with respect to its input size.

[TRUE/FALSE]
In a dynamic programming solution, the space requirement is always at least as big as the number of unique sub problems

[TRUE/FALSE]
In the divide and conquer algorithm to compute the closest pair among a given set of points on the plane, if the sorted order of the points on both X and Y axis are given as an added input, then the running time of the algorithm improves to O(n).

[TRUE/FALSE]
If f is a max s-t flow of a flow network G with source s and sink t, then the value of the min s-t cut in the residual graph G_f is 0.

[TRUE/FALSE]
Bellman-Ford algorithm can solve the shortest path problem in graphs with negative cost edges in polynomial time.

[TRUE/FALSE]
Given a directed graph G=(V,E) and the edge costs, if every edge has a cost of either 1 or -1 then we can determine if it has a negative cost cycle in O(|V|^3) time..

[TRUE/FALSE]
The space efficient version of the solution to the sequence alignment problem (discussed in class), was a divide and conquer based solution where the divide step was performed using dynamic programming.

2) 15 pts

Given an unlimited supply of coins of denominations $x_1, x_2, \ldots, x_n$, we wish to make change for a value v using at most k coins; that is, we wish to find a set of k coins whose total value is v. This might not be possible: for instance, if the denominations are 5 and 10 and k = 6, then we can make change for 55 but not for 65. Give an efficient algorithm for the following problem.

Input: $x_1, x_2, \ldots, x_n$, k, v.

Question: Is it possible to make change for v using at most k coins, of denominations $x_1, x_2, \ldots, x_n$? (No proof of correctness required)

3) 15 pts

Suppose you live with n-1 other people in an off-campus cooperative apartment. Over the next n nights, each of you is supposed to cook dinner for the entire group exactly once, so that someone different cooks on each night. Due to scheduling constraints (concerts, sports, etc), each person is unable to cook on certain nights, so deciding on who is cooking on each night appears to be a tricky task. Suppose we label the people $\{p_1, p_2, \ldots, p_n\}$ and the nights $\{d_1, d_2, \ldots, d_n\}$. Then for each person pi, there is a set of nights $Si \subseteq \{d_1, d_2, \ldots, d_n\}$ where p_i is unable to cook. A feasible dinner schedule is an assignment of each person in the co-op to a different night, so that each person cooks on exactly one night, there is someone cooking on each night, and if p_i cooks on night d_j then $d_j \notin S_j$.

Describe an algorithm to determine if there is a feasible dinner schedule or not. What is the running time of your procedure?

4) 20 pts

The input to your algorithm is an array A[1...n] of n numbers. You are further assured that n>2, A[1]>=A[2] and A[n]>=A[n-1]. A local minimum of the array A is defined as an index i such that 1<i<n, A[i-1]>=A[i] and A[i+1]>=A[i].

a) Prove that a local minimum of A always exists. (5 pts)

b) Design an algorithm that computes a local minimum while making at most O(log n) pairwise comparisons between elements of A. (no proof required) (15 pts)

5) 15 pts

You are in Downtown of a city and all the streets are one-way streets. You can only go east (right) on the east-west (left-right) streets, and you can only go south (down) on the north-south (up-down) streets.

a) In **Figure A** below, how many **unique ways** are there to go from the intersection marked **S** (coordinate (0,0)) to the intersection marked **E** (coordinate (5,4))? *(Hint: You may want to do part (b) first.)* (5 pts)

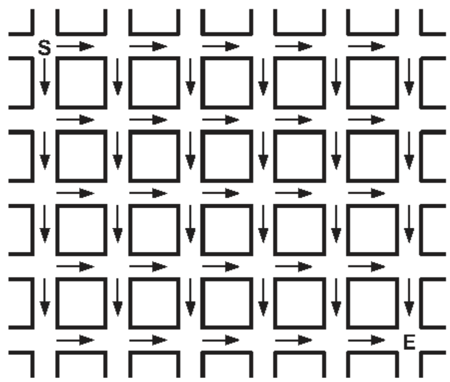

Figure A.

b) Continuing from part (a), formulate the solution to this problem as a dynamic programming problem, i.e., you need to define what OPT(. . .) is and write down the **recurrence equation** that expresses OPT(. . .) in terms of an **optimal** solution to "smaller" subproblems. Please make sure that you include all the boundary conditions and clearly define your notations you use. *(Hint: this is not an optimization problem.)* (5 pts)

c) Continuing from part (a), in **Figure B** above, how many **unique ways** are there to go from the intersection marked **S** to the intersection marked **E**? You can solve the problem numerically and directly on the figure *(Hint: beware of dead ends.)* (5 pts)

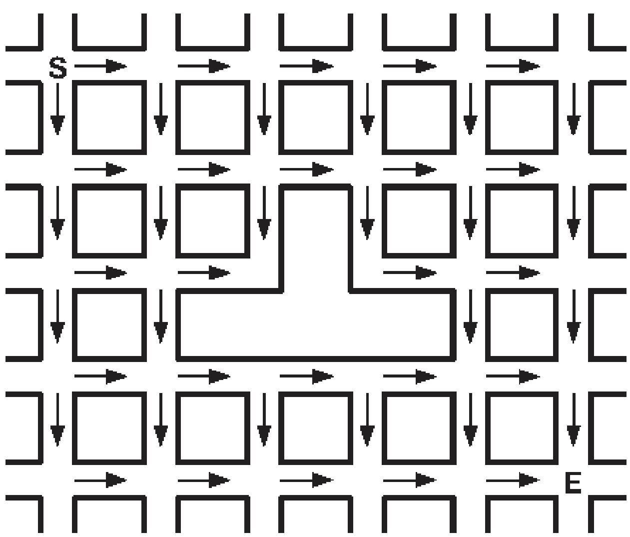

Figure B.

6) 15 pts
A spammer is located at one node q in an undirected communication network G and email users are located at a given set of nodes denoted by S. Let c_{uv} denote the effort required to install a spam filter for the network edge (u, v). The problem is to determine the minimal effort required to isolate the spammer from the email users using the spam filters. Give a polynomial-time algorithm to solve this problem. Justify your algorithm.

Additional Space