Sessional based on CSE 2201

Linear Time Sorting

Problem. Counting Sort

1. Generate N random integers within range 0 to 10000 in a file named input.txt.
2. Implement bubble sort to sort the numbers (from input.txt) and count the time.
3. Implement counting sort to sort the numbers (from input.txt) and count the time.

Increase the value of N and plot the performance curve for sufficiently large N to see distinguishable performance. Write your comments on the findings.

Report in the following format:

Time required to sort N numbers		
N	Bubble sort	Counting sort
10000		
50000		
100000		
50000		

Submit your code and report in a single pdf named your_roll.pdf to CR.

Deadline: 3 July 2021

Backtracking

Problem. N Queens

Using backtracking, solve the N queens problem. For any N taken as input, your code should find out the goal nodes as well as the bounding nodes (from where no more nodes are checked along that path and backtracking occurred). Each state/node is represent by following style.

Solution: 2 4 1 3 means placing at 2nd column of 1st row, 4th column of 2nd row ... gives a solution.

Backtrack From Node: 1 3 1 means after placing at 1st column of 1st row, 3rd column of 2nd row and 1st column of 3rd row checking is skipped along that path.

Sample Input: 4 Sample Output: Backtrack From Node: 1 1 Backtrack From Node: 1 2 Backtrack From Node: 1 3 1 Backtrack From Node: 1 3 2 Backtrack From Node: 1 3 3 Backtrack From Node: 1 3 4 Backtrack From Node: 1 4 1 Backtrack From Node: 1 4 2 1 Backtrack From Node: 1 4 2 2 Backtrack From Node: 1 4 2 3 Backtrack From Node: 1 4 2 4 Backtrack From Node: 1 4 3 Backtrack From Node: 1 4 4 Backtrack From Node: 2 1 Backtrack From Node: 2 2 Backtrack From Node: 2 3 Backtrack From Node: 2 4 1 1 Backtrack From Node: 2 4 1 2 Solution: 2 4 1 3 Backtrack From Node: 2 4 2 Backtrack From Node: 2 4 3	Backtrack From Node: 2 4 4 Backtrack From Node: 3 1 1 Backtrack From Node: 3 1 2 Backtrack From Node: 3 1 3 Backtrack From Node: 3 1 4 1 Solution: 3 1 4 2 Backtrack From Node: 3 2 Backtrack From Node: 3 3 Backtrack From Node: 3 4 Backtrack From Node: 4 1 1 Backtrack From Node: 4 1 2 Backtrack From Node: 4 1 3 1 Backtrack From Node: 4 1 3 2 Backtrack From Node: 4 1 3 3 Backtrack From Node: 4 1 3 4 Backtrack From Node: 4 1 4 Backtrack From Node: 4 2 1 Backtrack From Node: 4 2 2 Backtrack From Node: 4 2 3 Backtrack From Node: 4 2 4 Backtrack From Node: 4 3 Backtrack From Node: 4 4

Submit your code and report in a single pdf named your_roll.pdf to CR.

Deadline: 30 July 2021

Greedy Algorithm

Problem. DijKstra Algorithm

Generate random integers V (total vertices) and E (total edges) followed by E edges (u, v) and corresponding weights for an undirected graph.

For example, following lines show V=3 vertices and E=2 edges. Vertex 1 is connected to vertex 2 with weight 5. Vertex 3 is connected to vertex 2 with weight 8.

Sample Input:

3 2

1 2 5

3 2 8

For V=100, find the shortest path from any specified source node to any specified goal node. Represent the graph using Adjacency list as well as Adjacency matrix. Represent the priority queue using Binary heap as well as unsorted array.

Report in the following format:

Time to find shortest path		
Priority queue data structure	Adjacency list	Adjacency matrix
Binary heap		
Unsorted array		

Submit your code and report in a single pdf named your_roll.pdf to CR.

Deadline: 7 August 2021

Problem . Kruskal's Algorithm

Generate random integers V (total vertices) and E (total edges) followed by E edges (u,v) and corresponding weights for an undirected graph.

For example, following lines show V=3 vertices and E=2 edges. Vertex 1 is connected to vertex 2 with weight 5. Vertex 3 is connected to vertex 2 with weight 8.

Sample Input:

3 2

1 2 5

3 2 8

Find the minimum spanning tree using Kruskal's algorithm. Consider the disjoint set implementation using path compression.
Report in the following format:

V	E	Time (Path compression/Rank based approach)	Time(Without path compression/rank approach)	Solution (Total cost)

Submit your code and report in a single pdf named your_roll.pdf to CR.

Deadline: 7 August 2021

Dynamic Programming

Problem. Longest Common Subsequence
Find the Longest Common Subsequence (**LCS**) between two strings using dynamic programming.

Sample input: ABCDGH AEDFHR

Sample output: 3

Submit your code and report in a single pdf named your_roll.pdf to CR.

Deadline: 7 August 2021

NP hard problem using approximation algorithm

Problem . Minimum vertex cover

Generate random integers V (total vertices) and E (total edges) followed by E edges (u,v) and corresponding weights for an undirected graph.

For example, following lines show V=3 vertices and E=2 edges. Vertex 1 is connected to vertex 2 with weight 5. Vertex 3 is connected to vertex 2 with weight 8.

Sample Input:

3 2

1 2 5

3 2 8

For V=1000, find the minimum vertex cover from the graph using some approximation algorithm and mention the corresponding time. Show the theoretical bound(K) of your approximation algorithm.

Report in the following format:

V	Solution	Time
10		
50		
100		

Submit your code and report in a single pdf named your_roll.pdf to CR.

Deadline: 7 August 2021