UNIVERSITY OF VICTORIA
EXAMINATIONS APRIL 1997
CSC225 S01
ALGORITHMS AND DATA STRUCTURES
Instructor: Valerie King

NAME: _____ REG. NO.: _____

TO BE ANSWERED ON THE EXAM

Students *must* count the number of pages in this examination paper before beginning to write, and report any discrepancy immediately to the invigilator.

This exam has 14 pages and this cover page.

INSTRUCTIONS:

- DURATION: 3 hours
- All answers to be written on examination paper. There are 11 questions. Points are indicated with each question
- This is a closed book exam. No aids are allowed

	Points	Mark
Q1	30	
Q2	12	
Q3	8	
Q4	10	
Q5	20	

Q6	20	
Q7	20	
Q8	20	
Q9	20	
Q10	20	
Q11	20	
Total	200	

1. (30 points) Use big-oh notation to give tight, simple upper bounds on these functions:

   A. $T(1) = 1$; $T(n) = 3 + T(n-1)$: _____.

   B. Worst case running time of selection sort on n numbers: _____

   C. Worst case running time of heapsort on n numbers: _____

   D. BEST case running time for doing n insertions into an initially empty binary search tree: _____

   E. $5n^3 \log_2 n + 2n^5 \log_4 n + 10n$: _____

   F. $\sum_{i=1}^{n} i$: _____

   G. The worst case time for evaluating an expression tree with n nodes: _____

   H. The worst case time for doing lookup in a hash table of size n with n keys if chaining is used: _____

   I. The worst case time for performing enqueue in a queue with n elements which is implemented as a linked list: _____

2. (12 points) In the problems below, circle the correct choice or choices.

   A. In which data stucture(s) can an insertion be done in constant time (in the worst case)
   a. ordered array
   b. unordered array
   c. binary search tree
   d. hash table with chaining

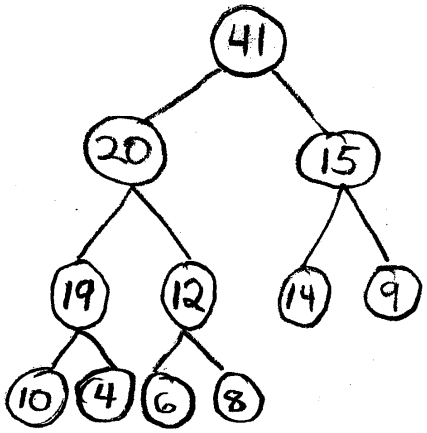
   B. In which of the following data structures(s) can all the keys contained within be output in sorted order in time $O(n)$ in the worst case? Assume each key has constant size.
   (a) heap
   (b) binary search tree
   (c) stack
   (d) queue
   (e) hash table

   C. The information theory lower bound for finding the median of n keys with a comparison based algorithm is $\Omega(\log n)$. (True or False).

3. (8 points) Draw a picture of a trie which stores the following words: dog, does, eat, every, ever, egg.

4. (10 points)

   (a) Redraw the heap below to show what it looks like after deletemax is executed:



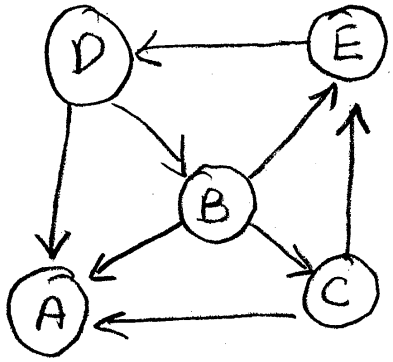
   (b) Circle above all keys which were involved in a comparison during the execution of deletemax.

5. (20 points) Let $R(h)$ be the maximum number of leaves in a tree of height h and branching factor k.

   (a) Express $R(h)$ as a recurrence relation. Don't forget to give a basis case.

   (b) Solve the recurrence relation by giving a tight big-oh upper bound on $R(h)$ and prove your solution is correct, using method A or method B from class, i.e., substitution (with induction proof) or guess and check (with induction proof).

6. (20 points)



Perform a depth-first search on the graph above. Assume that your algorithm considers nodes in ALPHABETICAL order and that adjacency lists are ordered alphabetically.

a. Label the nodes of the graph with the postorder numbering that results.

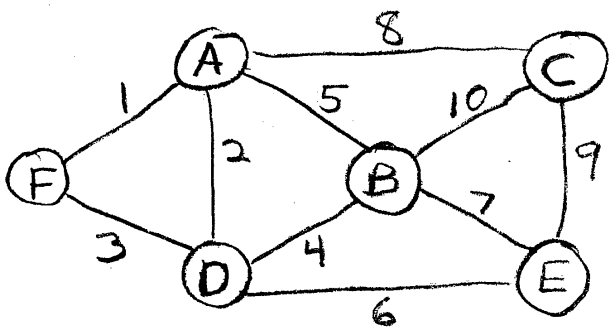
b. Draw the depth-first search tree below.

c. List the forward arcs.

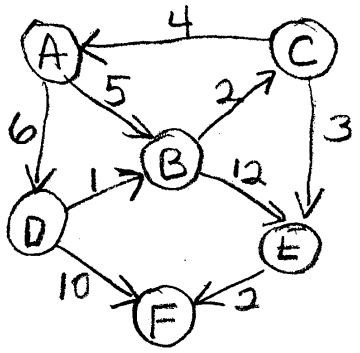
d. List the cross arcs.

e. List the backward arcs.

7. (20 points) Perform Kruskal's algorithm for computing a minimum spanning tree on the graph below. List the edges that are in the MST in order of how they were added to the tree.



8. (20 points)



Compute shortest paths from node A, using Dijkstra's algorithm on the above graph.

(a) Finish filling in this chart giving the value of $dist(v)$ for each node v and mark the set of settled nodes with a * for each stage of the algorithm.

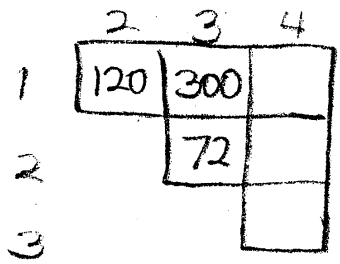
A	0*	0*			
B	5	5*			
C	∞	7			
D	6	6			
E	∞	17			
F	∞	∞			

(b) Suppose, in the implementation of Dijkstra's algorithm, we implement our priority queue as an unsorted list stored in an array. Each node contains a pointer to its position in the priority queue. Give a tight big-oh upper bound on the running time of the algorithm in terms of m and n, where m is the number of arcs and n is the number of nodes. Explain your answer.

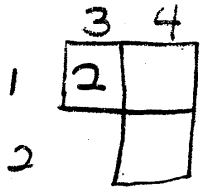
(c) Suppose we implement our priority queue on a sorted list stored as a linked list. What is the worst case asymptotic running time of Dijkstra's algorithm in terms of m and n? Explain your answer.

9. (20 points) Given a matrix chain $A_1A_2A_3A_4$, where the dimensions of A_1 are 10 x 4; A_2's are 4 x 3; A_3's are 3 x 6; and A_4's are 6 x 2, we would like to find the optimal way to parenthesize the chain so as to do the fewest scalar multiplications.

   (a) Finish filling in the tables below. Entry $m[i][j]$ of table M is the number of scalar multiplications needed to compute the product of $A_i...A_j$. Entry $s[i][j]$ of table S is equal to k if the cheapest way to compute the product is by breaking it into the two chains $A_i...A_k$ and and $A_{k+1}...A_j$ and computing their products first.



m



S

(b) What is the best way to parenthesize the chain?
$A_1 \ A_2 \ A_3 \ A_4$.

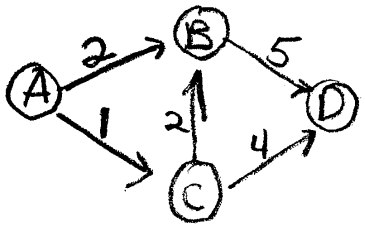
(c) How many scalar multiplications are needed?

10. (20 points)

(a) Give a high level description of an algorithm which finds the length of the LONGEST path from every node u to any other node in an ACYCLIC graph in time $O(m)$, where m is the number of arcs in the graph.

Hint: this is similar to a homework problem. Start by numbering the nodes in REVERSE topological ordering, and consider the nodes in that order.

Example:



Answers:

A: 8
B: 5
C: 7
D: 0

(b) Give the appropriate loop invariant for your algorithm to prove its correctness.

(c) Prove your loop invariant holds.

(d) Explain why your algorithm has $O(m+n)$ running time.

11. (20 points) Suppose a directed graph G with m arcs and n nodes represents a highway network where each edge is a one-way road. Let $wt[i][j]$ denote the maximum permissible vehicle weight for the road from city i to city j. We wish to find for each pair of cities (i, j), the weight of the heaviest vehicle that can be sent from i to j. Below are the recurrence relations on which is based the Floyd algorithm for all pairs shortest path problem. (This is consistent with the notation used in the text.) We let $dist[i][j]^k$ denote the length of the shortest k-path from i to j.

$$dist[i][j]^{-1} = arc[i][j]$$
$$dist[i][j]^k = min\{dist[i][j]^{k-1}, dist[i][k]^{k-1} + dist[k][j]^{k-1}\}, \text{ for } k \geq 0.$$

You are to modify them so that they are adapted for the purpose of carrying out the desired computation.

(a) Call your new function *maxwt* and give the formal definition of $maxwt[i][j]^k$, for $k \geq -1$. (I.e. Describe in words the quantity given by this function.)

(b) Give the recurrence relation for $maxwt[i][j]^k$. Don't forget the basis case.

(c) What is the running time of your dynamic programming algorithm for computing this problem? Explain.

END