UNIVERSITY OF MORATUWA

Faculty of Engineering

Department of Computer Science and Engineering

B. Sc. Engineering

Level 2 Semester 2 Examination 2009/10

CS 2020 – DATA STRUCTURES AND ALGORITHMS

Time allowed: 2 hours August 2010

INSTRUCTIONS TO CANDIDATES:

This paper contains **FOUR (4) questions** on 6 pages, including this page.

Answer **ALL** questions.

This is a **closed book** examination.

Mobile phones or any other communication devices are not permitted.

All questions carry equal marks.

This paper contributes to 70% of the total marks of the subject.

Clearly state the assumptions you made. If you have any doubt regarding the interpretation of the wording of a question, make your own decision, but clearly state it on the script.

Q1 **[25 marks]**

a) Which of the following problems can be solved using an algorithm? Explain why or why not for each problem.

   i. Build a house.

   ii. Given the date and time of your birth, find the date and time of your death.

   iii. Given the date and time of your birth, find the position of Saturn in the sky at that time.

   iv. Given two pictures of a person's face, identify if they are of the same person.

[4 marks]

b) Give an example of an Internet site which depends on algorithms, and explain how it uses them. [2 marks]

c) Algorithm A takes 10s to solve a problem for a given input, while algorithm B takes 30s with the same input. Which algorithm is better? Explain. [4 marks]

d) What operations does the stack data type support? [3 marks]

e) Show the implementation of a stack in pseudo-code using (a) an array and (b) a dynamic data structure. For each implementation, show the code for creating an empty stack, and for each operation supported by the data type. [12 marks]

Q2 **[25 marks]**

a) A heap is a special type of binary tree.
What are the properties of a max-heap? [3 marks]

b) Show, using a suitable diagram, a max-heap formed by the following numbers:
19 16 25 4 3 2 6 15 20 20 5 6 12.
Also show how the heap may be stored in an array. [5 marks]

c) What is the *height* of the heap in ii. above? [hint: a heap comprising a single node has height zero.] What is the maximum number of nodes which can be in a heap of that height? [2 marks]

Assume the array A of size n holds a binary tree and that the trees at LEFT(i) and RIGHT(i) are heaps, but that $A[i]$ might be smaller than its children. The procedure MAX_HEAPIFY(A, i) converts the tree rooted at i into a heap in time O(lg n).

d) Explain how a heap of size n in an array A may be converted to a sorted array using MAX_HEAPIFY, and show the pseudo-code for this procedure. [Assume that the procedure MAX_HEAPIFY(A, i) is already implemented.] [8 marks]

e) Explain, using pseudo-code, how you can sort 100 million 32-bit integers in linear time. [Hint: a 32-bit integer can be considered as four 8-bit base-256 digits.] [5 marks]

f) Assuming you have a computer which performs 1 million calculations per second, approximately how long would it take to sort the 100 million numbers? [You only need to give an approximate time, and can ignore operations which take a non-significant time.] [2 marks]

Q3 **[25 marks]**

a) Bellman-Ford algorithm and Dijkstra's Algorithm are two algorithms for calculating Single Source Shortest Path (SSSP).

i. Compare and contrast the two algorithms based on their generality and running time. [2 marks]

ii. Using one of the above algorithms, calculate the shortest path from A to all other vertices in the graph shown in following diagram (Figure Q3). Clearly indicate all the steps you followed. [7 marks]

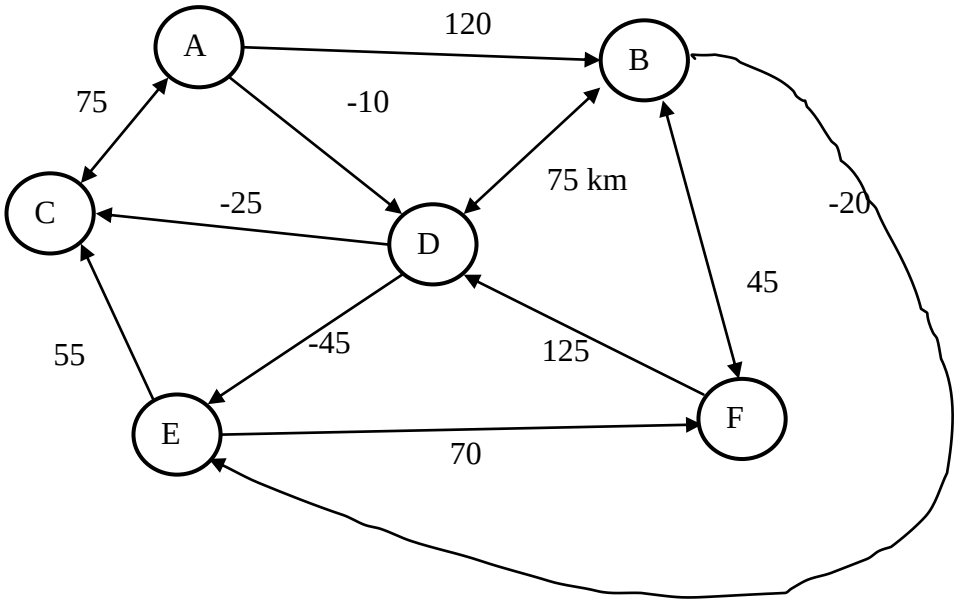


Figure Q3

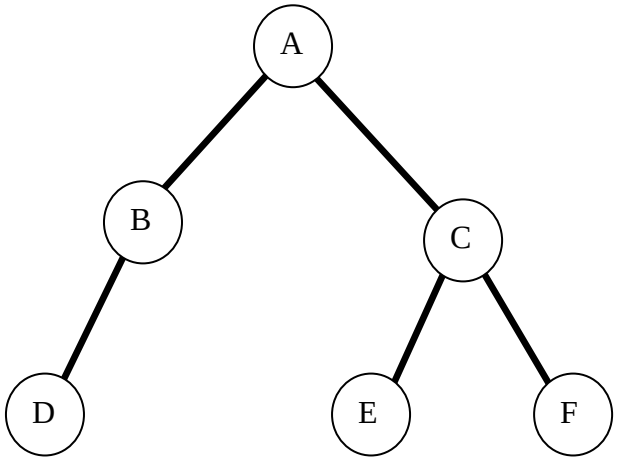
b) Breadth First Search (BFS) and Depth First Search (DFS) are two primitive graph search algorithms that have many applications.

i. Create a Depth First Forest for the graph shown in Figure Q3. Clearly indicate all the steps you followed. [3 marks]

ii. Is the above graph acyclic? Explain using the result you obtained in part (b) ii. [3 marks]

iii. Does DFS require any modifications when it is are applied to trees? Explain the required modifications or why no modification is required. [2 marks]

iv. Specify an order in which the nodes of the following tree will be finished if DFS (with any modifications that are specified above) is applied to it. Assume the links are undirected. [2 marks]



c) Consider the following three adjacency matrix representations of the graphs. For each graph, comment whether they;

- **can be** undirected graphs
- **can be** connected graphs

Please explain your answers. [6 *marks*]

i.

	0	1	2	3
0	0	1	0	1
1	1	0	0	0
2	0	0	0	0
3	1	0	0	0

ii.

	0	1	2	3	4	5
0	0	1	1	1	1	0
1	1	0	1	1	0	1
2	1	1	0	1	0	0
3	1	1	1	0	1	1
4	1	0	0	1	0	1
5	0	1	0	1	1	0

iii.

	0	1	2	3	4	5
0	0	1	1	1	1	0
1	1	0	1	0	1	0
2	0	1	0	1	0	1
3	0	1	1	0	1	1
4	1	0	0	0	0	1
5	0	1	0	1	1	0

Q4 **[25 marks]**

a) Solve the following recurrences using the recursion tree method.

i. $T(n) = T\left(\left\lfloor\frac{n}{2}\right\rfloor\right) + T\left(\left\lceil\frac{n}{2}\right\rceil\right) + n$ [2 marks]

ii. $T(n) = T\left(\left\lfloor\frac{n}{\max(\sqrt{n},2)}\right\rfloor\right) + n$ [4 marks]

b) Dynamic Programming is a technique which lets you solve certain set of problems efficiently compared to recursion and divide-and-conquer approaches. What is the property of the problem that enables the dynamic programming approach to complete the computation faster compared to recursion or divide-and-conquer approaches? How does the dynamic programming approach exploit that property to get the solution faster? [2 marks]

c) Consider the following 1-0 knapsack problem where you are given 8 objects with the following weights:

(4, 6, 2, 8, 4, 8, 6, 2)

Suppose the following profits are associated with the objects (in the same order):

(15, 80, 7, 70, 60, 75, 66, 12)

The knapsack has a weight capacity of 24. The objective is to fill the knapsack while maximizing the total profit.

Let x_i denotes the selecting or not selecting of the i^{th} item. x_i is 0 if the i^{th} item is not selected and x_i is 1 if i^{th} item is selected. Then, the problem can be stated as,

maximize $\sum_{i=1}^{n} p_i x_i$ subjected to the constraint $\sum_{i=1}^{n} w_i x_i \leq W$ where p_i, w_i and W represent profit of i^{th} item, weight of i^{th} item and the total weight limit respectively. Now if we let $P(i,k)$ as the maximum profit possible using items 1,2, ……., i and capacity k

a. Derive the recursive solution to the problem of selecting the optimal set of items to fill the knapsack. [2 marks]

b. Compute the solution to the problem using the dynamic programming approach in either top-down or bottom up approach. Specify the approach you used. [9 *marks*]

d) Consider a similar knapsack problem with the same set of items (with same profits and weights) and same weight limit. However, this time you are allowed to select a fraction of an item from **only one item**. You are free to select the item you want to select only a fraction.

a. Write an algorithm (pseudo code or the idea) that will solve the above problem with worst case time complexity of O(n). [2 *marks*]

b. Show that the worst case time complexity of the algorithm you developed in part a is O(n). [2 *marks*]

c. Find the optimal solution to the problem using the algorithm you developed in part a. [2 *marks*]