University of Swaziland
Final Examination
MAY 2012

Title of paper : Data structures

Course number : CS342

Time Allowed : Three(3) hours

Instructions :

- ***Each question carries 25 marks***
- ***Answer any four (4) questions from questions 1 to 6.***

This paper may not be opened until permission has been granted by the invigilator

Question 1

(a) Explain the meaning of the following terms.

(i) Data. 2 *marks*

(ii) Data structure. 2 *marks*

(iii)Abstraction 2 *marks*

(iv)Balanced tree 2 *marks*

(v) Recursive algorithm. *2 marks*

(b) **f(n) is O (g(n)).** Explain the meaning of this expression. 2 *marks*

(c) Using the big-oh notation estimate the running time of the following functions.

(i) f(N) = 1,000,000,000,000 *1 mark*

(ii) f(N) = N^2 + 1,000,000N + 10; *1 mark*

(iii)f(N) = 70000 LOG_2 (N)+ 2^N + 7000 *1 marks*

(iv)f(N) = 500 N LOG (N) + 100 if n > 10; 2 *marks*

10000 LOG N if N < = 10;

(d) Write the pseudocode of the binary search algorithm, clearly stating the preconditions and post-conditions. With the aid of an sample array containing not less than 13 values, trace the execution of a binary search algorithm. What is the running time of this algorithm?

4 *marks*

(e) Write the pseudocode of the selection sort algorithm, clearly stating the preconditions and post-conditions. What is the running time of this algorithm? Is there any other sorting algorithm that can do better than selection sort in terms of the running time, explain your answer?

4 *marks*

Question 2

Let *A* $[lo_1..hi_1, lo_2..hi_2]$ be a 2D array

(a) Write a C++ declaration of variable A as an 2D array of integers. 2 *marks*

(b) With the aid of an example, explain what is meant by row major order and column major order allocation for such a 2D array. Which one requires more memory? 5 *marks*

(c) Write an general expression that could be used to calculate the memory space required to store 1 column of array A given each element takes E_{size} bytes. 3 *marks*

(d) Write a general array mapping function, *Address (A[i,j])*, assuming column major order. Explain or show how you obtained this expression. 4 *marks*

(e) What is the big-oh time complexity for accessing element A[i,j]. 2 *marks*

(f) Assuming *A* $[lo_1..hi_1, lo_2..hi_2]$ is an array of employee records as defined below:

```
class Employee
{ string TaxId[10];
  string name[15];
  int age;
};
```

(i) Show a sample array A containing 3 such employee records. 2 *marks*

(ii) How much memory is required to store each record? Clearly state your assumptions about the size of each field of an employee record. 2 *marks*

(iii) Assume the base address array A is 500 and a C++ implementation of arrays, use the array mapping function obtained in (d) above to determine the first memory location used to store the name field value for the record stored in row 3 column 8. 5 *marks*

Question 3

Assuming a linked-list implementation of an ordered list,

(a) Draw a diagram of an ordered linked list with nodes containing the values 72.0, 10.2, 45.7, 16.5, 300.1 3 *marks*

(b) How much memory would be required to store the elements in the linked list in (a) above. What would be the difference if the elements we stored in an array? 3 *marks*

(c) Using C++ notation, define the structure of a **Node** in a linked list. 4 *marks*

(d) Write a C++ typedef that defines type **List** as a pointer to a **Node**, as defined in (b) above. 1 *marks*

(e) Write C++ functions that implement **init**, **isEmpty**, **add** and **show** operation on a ordered linked list as defined in (a) above. 10 *marks*

(f) Using the big-O notation, estimate the running times of the implementations given in (e) above. 4 *marks*

Question 4

(a) What is perfect binary tree? 2 *marks*

(b) Draw a picture of perfect binary search tree of height 3. How many leaf nodes are in this tree? 3 *marks*

(c) List and describe the operations of a stack data structure. 3 *marks*

(d) Using the example of tree obtained in (b) above, describe in-order tree traversal, and write a pseudocode of an algorithm that uses a stack to implement in-order tree traversal. Trace the execution of this algorithm using the sample tree in (b) above. 7 *marks*

(e) List and describe the operations of a queue data structure. 3 *marks*

(f) Using the example of tree obtained in (b) above, describe level-order tree traversal, and write a pseudocode of an algorithm that uses a queue to implement level-order tree traversal. Trace the execution of this algorithm using the sample tree in (b) above. 7 *marks*

Question 5

(a) What is a B-Tree of order b? *2 marks*

(b) Write the pseudocode for inserting into a B-tree of order b. *4 marks*

(c) Follow the pseudocode outlined in (a) above and construct a B-tree of order 5 containing the following values. Show all intermediate trees leading to your final answer.

20	98	10	200	400	15	250	315	30	75
650	100	150	70	170	800	50	80	100	40

Assume values are inserted in the given order. *13 marks*

(d) List all node values in the B-tree constructed above assuming pre-order traversal. What is the running time of this traversal? *3 marks*

(e) What is the height of a B-tree of order 19 containing 130000 values. *3 Marks*

Question 6

(a) Draw a picture of a sample directed graph G with 11 nodes and 19 edges. Each node must have at least 2 but not more than 3 neighbors. *3 marks*

(b) Show the adjacency list representation of the above graph G in (a) above. 2 *marks*

(c) Using the C++ Standard Template Library;

(i) Define a class **GraphNode** to represent a node in a graph with *NodeValue* and *AdjacentList* as its data members. The **node value** is of type *datatype and* **AdjacentList** is a **list** of values of type *datatype*. Define a constructor for this class. [***do not define any other member functions for this class***] 5 *marks*

(ii) Using your class definition in (i) above, define a class **Graph** with *Nodes* as its data member where *Nodes* is a **array** of GraphNodes.
[***do not define any member functions for this class***]
3 *marks*

(d) Based on your type definition in (a.) above, write C++ code for the following member functions of class Graph.

(i) *NodeIndex (u)* – returns the index of node u in a graph. 4 *marks*

(ii) *IsNode (u)* – returns true if u is a node in the graph and returns false otherwise. 2 *marks*

(iii) *AddEdge (u,v,)* – adds a new edge from node u to node v. 3 *marks*

(iv) *Neighbours (u)* – displays all the neighbors of node u (the adjacent list of node u) in a graph. 3 *marks*