National Exams December 2016

04-Soft-A1, Algorithms and Data Structures

3 hours duration

NOTES:

1. If doubt exists as to the interpretation of any question, the candidate is urged to submit with the answer paper, a clear statement of any assumptions made.

2. This is an CLOSED BOOK EXAM. Any Casio or Sharp approved calculator is permitted into the exam.

3. The exam consists of seven questions; pick five of your choice and the first five questions **as they appear in your answer book** will be marked. Each question is of equal value.

Marking Scheme

All Questions are equal 20 points.

1. 20 marks total (4 items, 5 points each)
2. 20 marks total (5 items 4 points each)
3. 20 marks total (4 items, 5 points each)
4. 20 marks total it's the ADT and 3 access functions so 4 5 points each
5. 20 marks total: 10 points for ADT definition, 5 each for the two access functions.
6. 20 marks total (4 items, 5 points each)
7. 20 marks total (5 items 4 points each)

NOTE: "To implement", you can choose any programming language like C, C++, Java, or Python. You can also use clean consistent pseudo code.

1) Binary Tree

An implementation of a binary tree storing integer number is given. (The signatures, also called prototypes showed below, are given in C, but you can choose another adequate language if you prefer; and then replace pointers with objects if needed).

BTree is the pointer type to the tree. The provided interface functions include:

- **BTree get_left_child(BTree tree)** returns a pointer to the left subtree, and Null if there is no such child.
- **BTree get_right_child(BTree tree)** returns a pointer to the right subtree, and Null is if there is no such child.
- **void delete_node(BTree tree)** if the pointed tree has no children it is deleted, else an error is printed and the program terminated.
- **int is_empty(BTree tree)** return 0 if the pointed tree is empty, else it returns 1.
- **int value(BTree tree)** returns the value stored at the root of the tree.

Implement the following 3 functions using the above functions.
(No direct manipulation of the tree data type allowed)

1. A function which computes the sum of all entries in the BTree
   int compute_sum(BTree tree).

2. A function which deletes all leaf nodes of the BTree:
   BTree delete_leafs(BTree tree).
   (A leaf node is a node with no children)

3. A function computing the depth of the binary tree
   int compute_depth(BTree tree)

2) Max Heap

We use an array to implement a max heap.

1. Describe in 1-2 sentences:
   a. How to compute the index of the parent of a node with index *n*?
   b. How to compute the index of the right child of a node with index *n*?
2. Describe in 2-3 sentences the process of adding a node to a max heap
3. Describe in 2-3 sentences the process of deleting a node from a max heap.
4. Illustrate each step by a figure the addition of 4, 6, 3, 7, and 1 to a max heap, starting with an empty heap.
5. What is the complexity of adding a node to a max heap, what is the complexity of deleting a node from a max heap? Justify you answer in 1-2 sentences.

3) Graph Traversal and Spanning Trees

1. Describe in a few sentences each of the two searching algorithms:
   Depth First Search and *Breadth First Search*.
2. What is the relation between these searching algorithms and spanning trees?
3. Describe in a few sentences how Kruskal algorithm finds a minimal cost spanning tree.

4) Waiting

For a public office we are asked to create a software system clients in the order they arrive in the office. For this, they get a number at the entrance. The maximum capacity of the waiting room is 100 people.
Define an ADT (Abstract Data Type) for the waiting list.
There are 3 functions to implement:

1. **int Get_number**() gives a unique number (1-100) to a new entering customer, it returns an Error if the waiting room is full
2. **int get_next()** gives the number of the next customer to be served
3. **int num_waiting**() (0-100) gives the number of people waiting.

Define the data type, and implement the three functions in a language of your choice.

5) Double Linked List

Define the ADT of a double linked list storing integer numbers. Implement the function **void add_front(int data)** and **int delete_end(void)** in a language of your choice.

Note: An ADT consists of the types and data structures used, and a list of all interface functions stating their argument types and their return type.
Draw a "picture" illustrating the two procedures if that helps you.

6) Merge Sort

1. Describe the merging of two sorted lists in 3-4 sentences.
2. Implement the function
   merge(int a[MAX],int b[MAX],int c[MAX+MAX])
   merging the arrays *a* and *b* into array *c* of size MAX.
3. Describe the Merge Sort algorithm in 3-4 sentences.
4. Show in pictures illustrating the steps needed to sort the numbers
   9, 5, 2, 6, 1, 7, 8, 3
   with Merge Sort.

7) Short Questions:

Give a short 2-3 sentences answer to each question.

1. How to determine if a linked list has a loop
2. Explain the concept of a hash table
3. What is a non-deterministic polynomial problem, NP-problem? Make sure you explain the two properties: non-deterministic and polynomial.
4. Explain the concept of a greedy algorithm?
5. Explain the tradeoff between space and time complexity in choosing a data structure and algorithm to search for strings in a large ASCII document.