UNIVERSITY OF MORATUWA

Faculty of Information Technology

B.Sc. (Hons) in Information Technology
B.Sc. (Hons) in Information Technology and Management
Level 2 – Semester 2 Examination
IN 2110 – Data Structures and Algorithms

Time Allowed: 3 hours | October 2019

INSTRUCTIONS TO CANDIDATES

1. This paper contains 5 questions on 6 Pages. (including this page)
2. The total marks obtainable for this examination is 100. The marks assigned for each question are included in square brackets.
3. This examination accounts for 70% of the module assessment.
4. This is a closed book examination.
5. Answer ALL questions.

ADDITIONAL MATERIAL

None.

Continued...

Question 01

(a) Given the following array as input, illustrate how the *Mergesort* algorithm performs. To illustrate the *Mergesort's* behavior, start with dividing of the array until the end condition of the recursive function is met and then show how the merge is performed.

3 8 4 10 1 5 6 9

[6 Marks]

(b) Consider the following array with nine (9) elements. Assuming that *Quicksort* will be used to sort this array in ascending order, select a value for the last element of the array (indicated by "?") such that the partitioning performed by *Quicksort* is most balanced. Explain why this makes *Quicksort* perform efficiently.

10 5 3 9 22 24 28 27 ?

[4 Marks]

(c) Consider the following method *find2C* which accepts an integer array of size n and an integer x. *find2C* returns true if the sum of any two consecutive numbers in the array equals x. What is the complexity of *find2C* method in *big-O* notation.

```
boolean find2C(int[] myArray, int x)
{
  for(int i=0; i<myArray.length-1; i++)
  {
    if(myArray[i] + myArray[i+1] == x)
    {
      return true;
    }
  }
  return false;
}
```

[4 Marks]

(d) *"Quick sort is always faster than the Insertion sort."*
State whether you agree or disagree with the above statement with justifications.

[6 Marks]

Question 02

(a) Write two (2) advantages of *arrays* over *linked lists*.

[2 Marks]

(b) Write an algorithm to search an item in a *linked list*.

[6 Marks]

Continued..

(c) A singly linked list is stored in an array as shown below. Each array element has 2 fields, which are the key value and the next node index (N/A means no value). The "Next" value indicates the index on the next node (in the list) stored in this array. The head of the list is shown above at index [0].

head--->

Index	Key	Next
[0]	10	[6]
[1]	N/A	N/A
[2]	87	[8]
[3]	9	NULL(-1)
[4]	31	[3]
[5]	10	[2]
[6]	8	[5]
[7]	N/A	N/A
[8]	90	[4]

(i) What is the index of the tail node of this singly linked list?

[2 Marks]

(ii) What is the index of the third (3^{rd}) node in this list?

[2 Marks]

(iii) A new node is inserted between the third (3^{rd}) and the fourth (4^{th}) node of this list and the content is stored in the index [7] of the array (the key value is 999). Clearly show all the changes in the array.

[8 Marks]

Question 03

(a) State whether the following statements are TRUE or FALSE with justifications.

(i) Suppose you have a list of names sorted in alphabetical order. The easiest way to print the names in reverse alphabetical order would be to use a *stack*.

(ii) *Breadth-first search* is best implemented using a *stack*.

[6 Marks]

(b) What would be the contents of *queue* Q1 and *queue* Q2 after the following code is executed and the following data are entered? The data entered are 5, 7, 12, 4, 0, 4, 6.

```
Q1 = CreateQueue
Q2 = CreateQueue
Loop (not end of file)
          Read number
          Enqueue (Q1, number)
          Enqueue (Q2, number)
          Loop (not empty Q1)
                    Dequeue (Q1, x)
                    Enqueue (Q2, x)
          End loop
End loop
```

[4 Marks]

Continued..

(c) You are given a predefined *stack* and a *queue*. The *stack* and the *queue* contain same element type (int) and hold the same maximum number of elements. The following functions are available for use:

```
public class Stack{
 public boolean empty(){};
 public void push(int n){};
 public int pop(){};
}
```

```
public class Queue{
 public boolean empty(){};
 public void enQueue(int n){};
 public int deQueue(){};
}
```

(i) Write a method to move all the nodes from a *stack* to a *queue* by only using the methods available in above Stack and Queue classes.

[5 Marks]

(ii) Write a method to return the number of items in the *queue*. There is no "size" function provided as part of the class. You are only allowed to use the functions given above.

[5 Marks]

Question 04

(a) If a *perfect (complete) binary tree* has n leaves and all levels are fully populated, how many nodes does the tree have in terms of n?

[4 Marks]

(b) Figure Q4 represents a Binary Search Tree (BST). Draw the final BST after each of the following operations performed one after the other.

(i) Remove 10 (ii) Add 33 (iii) Remove 16

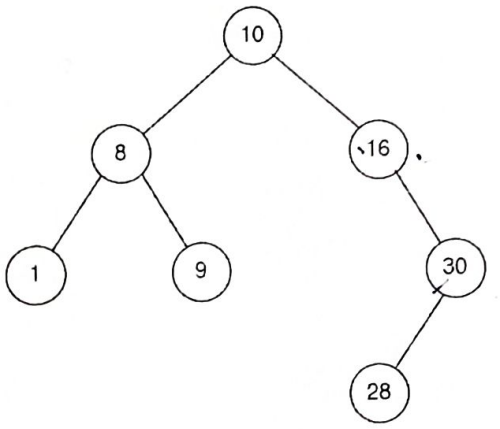

Figure Q4

[4 Marks]

(c) Write down the nodes in the order they are reached if you perform a *preorder* traversal for the tree shown in Figure Q4 starting with node 10.

[4 Marks]

(d) The integers 7, 1, 12, 8, 3, 0, –1, 9 are inserted in the same order into an initially empty BST. Draw the tree after the last insertion.

[4 Marks]

(e) Write the Java code that finds the element with smallest value in BST.

[4 Marks]

Continued..

Question 5

Consider the graph shown in Figure Q5 to answer part (a), part (b), and part (c).

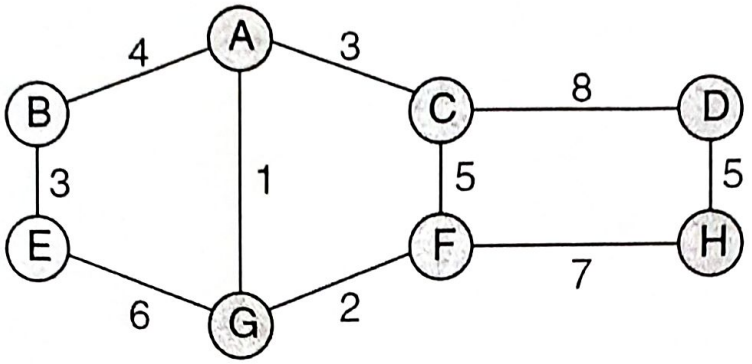

Figure Q5

(a) Write down the nodes in the order they are reached if you perform *depth-first traversal* of the graph in Figure Q5, starting from vertex A. Show the spanning tree.

[4 Marks]

(b) Write down the nodes in the order they are reached if you perform *breadth-first traversal* of the graph in Figure Q5, starting from vertex A. Show the spanning tree.

[4 Marks]

(c) Draw the adjacency matrix representation of the graph in Figure Q5.

[4 Marks]

(d) Suppose **T** is a tree and you are supposed to run *Depth First Search (DFS)* and *Breadth First Search (BFS)*_ starting from the same node **n**. Are the resulting trees the same? Justify your answer.

[8 Marks]

End of Paper