The Open University of Sri Lanka
Faculty of Natural Sciences
B. Sc/ B. Ed Degree Programme

Department	: Computer Science
Level	: 03
Name of the Examination	: Final Examination
Course Code and Title	: CSU3302/CPU1142 – Data Structures and Algorithms
Academic Year	: 2019/2020
Date	: 15.11.2020
Time	: 9.30 am -11.30 am
Duration	: Two (02) Hours

General Instructions

1. Read all instructions carefully before answering the questions.
2. This question paper consists of **six (06)** questions in **seven (07)** pages.
3. Answer any **four (04)** questions only. All questions carry equal marks.
4. Answer for each question should commence from a new page.
5. Draw fully labelled diagrams where necessary.
6. Involvement in any activity that is considered as an exam offense will lead to punishment.
7. Use blue or black ink to answer the questions.
8. Clearly state your index number in your answer script.

Answer FOUR Questions ONLY.

Q1.

a) i. What is a Data Structure?

ii. List down the three (03) basic operations that can be performed with Data Structures.

b) What is a List?

c) Distinguish between Array Implementation of List and Pointer Implementation of List.

d) Use the following C language declaration of Array Implementation of List to answer the questions from d (i) to d (iii).

```
#define nodes 100
struct node
{
            int info;
            int next;
}
struct node strArray [nodes];
```

i. Graphically explain the functionality of the above C code.

ii. Write C program code for the following functions:

`int getNode ( )` - Removes a node from the available list and returns a pointer to it.

`void insertAfter (int p,int x)` - Inserts variable x into a node following the pointer (p).

iii. Write the function `freenode` which accepts a pointer to a node and returns that node to the available list.

Q2.

a) What is a Stack? Explain using an example.

b) Write codes in C programming language to do the following.

i. Create a data structure to implement a **stack.** The structure should contain a variable to store the **top** position of the **stack** and an **array** to hold numeric data of type **int** in the **stack.** Declare a stack with the name "**mystack**".

ii. Write a function in C to **insert** a data item into "**mystack**". Use **void StackInsert(int x)** as the function header (check stack overflow condition before you insert a data item).

iii. Write a function in C to **delete** the data item at the top from "**mystack**". Use **int StackDelete** ( ) as the function header. (Check the stack underflow condition before you delete a data item).

c) What are the two basic operations associated with a Queue?

d) Following is a function in C programming language for some operation.

```
int function()
{
int x;
if(q. front == q.rear)
{
printf("queue underflow \n");
exit(1);
}
q. front = (q. front + 1)% MAXSIZE;
x = q.items[q. front];
return x;
}
```

By analyzing the above function, answer the following questions.

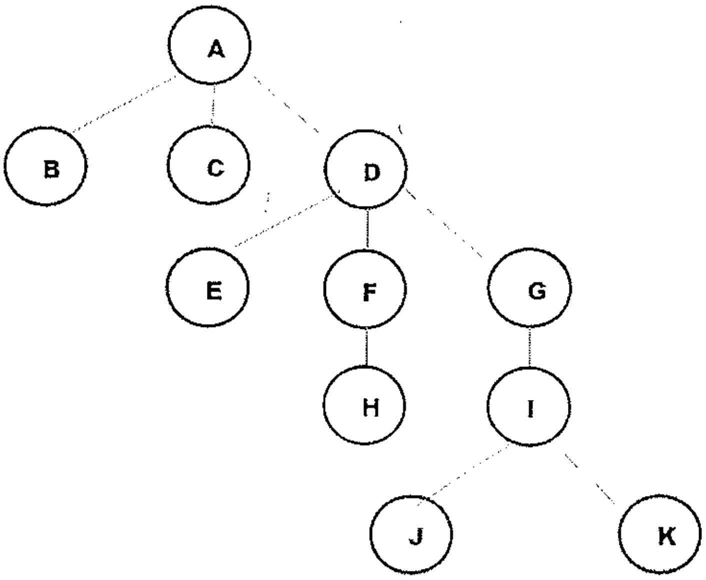
i) What is the data structure that this function belongs to?

ii) What type of implementation method is used to create the data structure?

iii) Which type of operation can be performed using the above function?

Q3.

a) Consider the following Tree to answer the questions from a) i to a) vi



i. Mark the levels of the Tree.

ii. What are the Sub Trees of the node D?

iii. What are the descendants of node G?

iv. What is the depth of the Tree?

v. What is the degree of the node D and node G?

vi. What is the degree of the Tree?

b) Construct a Binary Search Tree for the following set of integers.

24, 27, 12, 32, 7, 49, 8, 16, 33, 3, 11, 21, 10, 15, 53, 65

c) What will be the output, when you traverse the Tree in the following orders?

i. Preorder (NLR)

ii. Inorder (LNR)

iii. Postorder (LRN)

00532

Q4.

a) "Graphs are non-linear and non-hierarchical data structures."

Differentiate graphs from linear data structures and hierarchical data structures.

b) The adjacency matrix of a directed graph is given below.

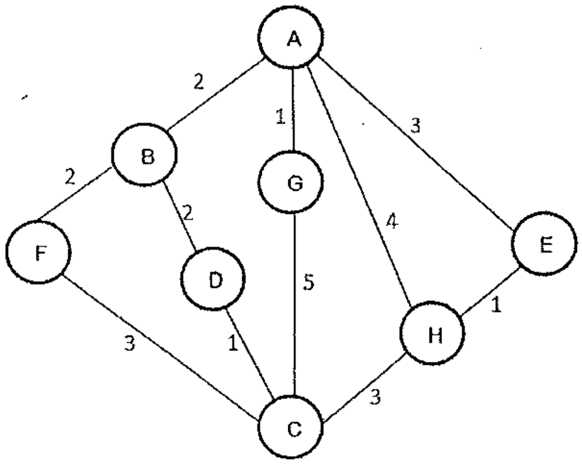
	A	B	C	D	E	F
A	0	1	0	0	0	0
B	0	0	1	0	1	0
C	0	0	0	1	0	0
D	1	1	0	0	0	0
E	0	0	0	0	0	1
F	0	1	0	0	0	0

i. Draw a directed graph that corresponds to the above given adjacency matrix

ii. Write down the edges in the graph.

iii. State, giving reasons, whether the graph you have drawn is a strongly connected graph or a weakly connected graph.

c)



i) Write the breadth first traversal of the graph given above, starting from A.

ii) Give all possible paths from vertex A to vertex C and identify the shortest path.

Q5.

a) What does it mean by an algorithm? Mention two examples for types of algorithms

b) What do you mean by the complexity of an algorithm?

c) What are the factors affecting the running time of a program?

d) In computer science what is meant by '**recursion**'?

e) What are the two main types of recursions?

f) Look at the following piece of C code regarding recursion.

```
int fact(int n)
{
   if (n == 0)
    return 1;
   else
    return n * fact (n-1);
}
```

Showing the steps clearly, employee the above function for the n=4 situation. What is the final output value?

g) Write a recursive function to print the square values of numbers from a given positive number down to 1. The output should be as follows, if the given number is 5.

```
square value of 5 = 25
square value of 4 = 16
square value of 3 = 9
square value of 2 = 4
square value of 1 = 1
```

Hint: $n^2 = (n-1)^2 + 2 \times n - 1$

Q6.

a) The techniques of sorting can be divided into two categories.
Mention and briefly explain them

b) How many passes are required to sort *n* number of integers by using the bubble sort?

c) Explain how the Bubble sort algorithm works on the following set of integers, when sorting them in the ascending order.

2, 1, 6, 4, 2, 3, 5

d) Consider the following Bubble sort algorithm.

```
void bubbleSort (int a[ ], int n)
{
        for (i=0;  i<n-1;  i++) {
             for (j=0;  j<n-1-i;  j++)
                  if (a[j+1] < a[j]) {
                        tmp = a[j];
                        a[j] = a[j+1];
                        a[j+1] = tmp;
                  }
             }
}
```

The above implementation has the following inner for loop:

```
for (j=0; j<n-1-i; j++)
```

Consider the effect of replacing this with the following statement:

```
for (j= n-1; j>i; j--)
```

Write the modified algorithm after replacing `for (j= n-1; j>i; j--)` to the inner for loop to get the same result.

e) Compare the *Bubble sort* algorithm with the *Quick sort* algorithm.