Semester 2 Examinations, 2021/2022

Exam Code(s)	2BCT1, 2BLE1, 2BP1, 1EM1, 1OA2
Exam(s)	Second BSc in Computer Science & Information Technology Second BE in Electrical & Electronic Engineering Second BE in Electronic & Computer Engineering
Module Code(s)	CT2109
Module(s)	Object Oriented Programming: Data Structures and Algorithms
Paper No.	7
External Examiner(s)	Dr Ramona Trestian
Internal Examiner(s)	Prof. Michael Madden *Dr. Frank Glavin
Instructions*:*	Answer any **three** questions. All questions carry equal marks.
Duration	2 hours
No. of Pages	6
Discipline(s)	Computer Science
Course Co-ordinator(s)	Dr. Colm O'Riordan
Requirements	None

Question 1

a) Your friend has recently started studying Computer Science. They are finding some aspects of the course difficult and have sent you the following message:

"*Hey, yes we recently started studying something called Dynamic Programming. I am not fully sure how it works as I missed a few classes. Could you explain it to me using simple examples that are easy to understand!? Keep in mind that I am really new to Computer Science so the simpler the better... Thanks!!*"

Reply to your friend and explain how Dynamic Programming works in simple terms using appropriate examples.

[6 Marks]

b) Describe, in your own words, why you think algorithm analysis is important. Outline the benefits and drawbacks of evaluating an algorithm "*on paper*" versus coding up the solution for comparative purposes.

[6 Marks]

c) Outline the differences between developing an *iterative* versus a *recursive* solution to a problem. Provide Java code for both an iterative and recursive solution to calculate the *factorial* of a given number (with coding comments clearly explaining the functionality in each).

[7 Marks]

d) Describe your understanding of the differences between P and NP problems by defining both. Provide a unique example for each. What is meant by the *P versus NP* problem?

[6 Marks]

[PTO]

Question 2

a) Describe how the *Quick Sort* algorithm operates and then demonstrate how the algorithm would work on the array below by working through *all* iterations/steps until the array is sorted:

[12, 98, 34, 67, 54, 23, 56, 45]

[7 Marks]

b) What is meant by the concept of *Generics* in Java programming? Show a simple example of how this concept could be used in the context of a binary tree Java implementation.

[6 Marks]

c) Provide algorithms for both a *Binary Search* and a *Sequential Search*, including details of any requirements that are imposed on their inputs. Demonstrate how the binary search algorithm would work for the following list of elements when the search item is 16.

[1, 4, 7, 8, 9, 13, 15, 16, 19]

[8 Marks]

d) Describe, in your own words, the differences between Java's *comparator* and *comparable* interfaces. Provide code snippets to aid your description of both.

[4 Marks]

[PTO]

Question 3

a) What does it mean to *traverse* a binary tree? Provide a description, and pseudocode, to differentiate between *Pre-Order Traversal*, *Post-Order Traversal*, and *In-Order Traversal* in the context of traversing a binary tree.

[6 Marks]

b) Describe what an AVL tree is and then, using the dataset below, illustrate how to perform the following operations:

(i) Inserting the data, *one item at a time in the order given* from the dataset below, into an AVL tree
Note: clearly *identify and describe <u>all</u> rotations and draw the state of the tree after <u>each</u> insertion*)

(ii) Searching for the value 16 within the resulting AVL tree.

The dataset is: [25, 15, 10, 18, 16, 26, 27].

[8 Marks]

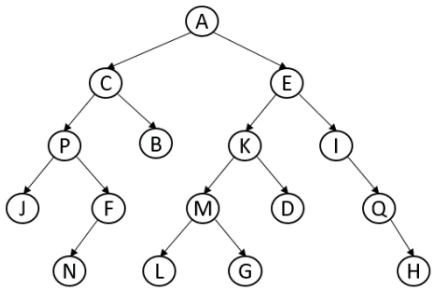
c) In the context of compression algorithms, describe what is meant by *Huffman Encoding*. Outline the steps (using both a text explanation and tree diagrams) for constructing a *Huffman Tree* with the following letters and frequencies.

A:5 B:8 C:1 D:3 E:2 F:1 G:5

[7 Marks]

d) From the tree shown, clearly identify each of the following:

(i) The leaves?
(ii) The internal nodes?
(iii) The parent of D?
(iv) The ancestors of F?
(v) The descendants of I?
(vi) The height of the tree?
(vii) The depth of K?
(viii) The subtree of C?



[4 Marks]

[PTO]

Question 4

a) A vaccination centre want you to develop a Java program for storing patient details as they enter the premises. The program should be able to sort them by name and write them out to a file every night when the centre is closed.

The owner is currently unsure about what the best data structure to use would be and has asked you to outline the pros and cons of using:

- An array
- A singly-linked list
- A doubly-linked list

Define each of these, with illustrations, and discuss why they might or might not be suitable in this case.

[9 Marks]

b) The following code snippets are from an implementation of the *CircularArrayQueue* class.

```
public void enqueue(Object n)
{
        if (isFull()) {
                JOptionPane.showMessageDialog(null, "Cannot enqueue object '"
                + (String) n + "'; queue is full.");
                return;
        }
        rear++;
        Q[realindex(rear)] = n;
}


public Object dequeue()
{
        if (isEmpty())
                return null;

        Object toReturn = Q[realindex(front)];
        Q[realindex(front)] = null;
        front++;
        return toReturn;
}
```

i. What benefit does the CircularArrayQueue have over a regular ArrayQueue? [1]
ii. Explain the full functionality of the two methods above. [4]
iii. What is the purpose of having a *realIndex* method? [2]

[7 Marks]

[PTO]

c) Define, in your own words, what is meant by an *Abstract Data Type*. Write code to define a Java **interface** corresponding to a Stack ADT and provide a short description for each of the five key Stack operations.

[5 Marks]

d) Describe the difference in operation between a *stack* and a *queue* using diagrams as appropriate. Outline an example use for each that can be found on a computing system.

[4 Marks]

[END]