OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Semester 2 Examinations 2022-2023

Course Instance 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.	1
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	5
Discipline(s)	Computer Science

Requirements:

Release in Exam Venue	No[☑]	Yes [ ]
MCQ Answer sheet	No [☑]	Yes [ ]
Handout	No [☑]	Yes [ ]
Formulae & Tables*	No [☑]	Yes [ ]
Cambridge Tables 2nd Edition**	No [☑]	Yes [ ]
Graph Paper*** A4 Graph Paper 1mm 0.1cm Squared (Standard)	No [☑]	Yes [ ]
Other Materials	No [☑]	Yes [ ]
Graphic material in colour	No [☑]	Yes [ ]

End of requirements.

[PTO]

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 Abstract Data Types. I am not fully sure what they are since I missed a few classes. Could you explain it to me using simple examples that are easy to understand!? I also recently implemented a Queue using an array as the underlying data structure. Dequeuing seems to get really slow and inefficient as the size gets bigger. Do you know why this is and can you describe how I can fix this? Please include code snippets and diagrams to help me! Thanks.*"

Reply to your friend and explain what an Abstract Data Type is by providing a definition and simple examples as part of your explanation. You should also explain what issues are encountered when using a regular array for a Queue implementation. Make suggestions as to how the code should be updated to make it more efficient with some small changes. Include code snippets of the updates and diagrams for a comprehensive response.

[10 Marks]

b) Explain the differences between a Singly Linked List and a Doubly Linked List by providing definitions and diagrams of each. Provide all the code (with comments) for a Singly Linked List implementation of the following interface:

```
public interface Queue
{
      public void    enqueue(Object n);
      public Object  dequeue();
      public boolean isEmpty();
      public boolean isFull();
      public Object  front();
}
```

[10 Marks]

c) 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.

[5 Marks]

[PTO]

Question 2

a) Provide the full Java code implementation for both a *Sequential Search* and a *Binary Search*, including details of any requirements that are imposed on their inputs. Using a diagram, demonstrate how the binary search algorithm would work for the following list of elements when the search item is 7.

[1, 4, 6, 7, 8, 9, 13, 17, 25, 26, 29]

[8 Marks]

b) Describe what is meant by O-notation and outline how the algorithmic efficiency could be derived for the following algorithm which finds the largest integer stored in an array.

```
Algorithm arrayMax(A, n)
   currentMax = A[0]
   for i = 1 to n – 1 do
      if A[i] > currentMax then
         currentMax = A[i]
      {increment counter i}
   return currentMax
```

Clearly identify all primitive operations and include all workings from the derivation of the expression. Comment on the final O-notation that you have derived.

[6 Marks]

c) Describe how the *Shell Sort* algorithm operates (using pseudocode and/or diagrams) and then demonstrate how the algorithm would work on the array below by working through *all* iterations/steps until the array is sorted:

[15, 58, 31, 61, 55, 29, 56, 47, 8, 42, 80, 53, 92]

[7 Marks]

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

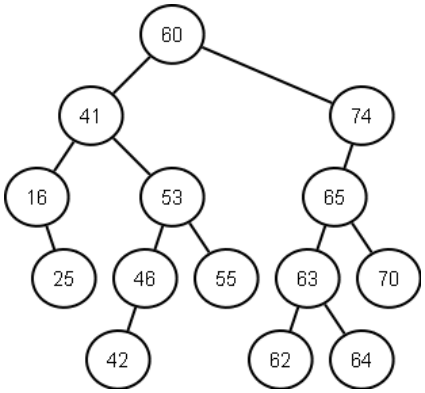
[4 Marks]

[PTO]

Question 3

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

(i) The leaves
(ii) The internal nodes
(iii) The parent of 25
(iv) The ancestors of 55
(v) The descendants of 65
(vi) The height of the tree
(vii) The depth of 53
(viii) The subtree of 41



[4 Marks]

b) Describe how an AVL operates. An addition or deletion from a AVL tree can cause an imbalance. Outline four different scenarios for additions to the tree, in which specific rotations would be required.
With these scenarios in mind, complete the following tasks:

(i) Insert 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) Outline the steps that take place when searching for the value 16 within the resulting AVL tree.

The dataset is: [22, 13, 9, 18, 15, 26, 27].

[8 Marks]

c) What does it mean to *traverse* a binary tree? Provide a brief 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]

d) Outline the steps involved in the *Radix Sort* algorithm.
Illustrate how the algorithm would work for the following unsorted array:
You should include all the intermediate steps involved.

[122, 392, 215, 018, 518, 007, 512, 127, 221, 296]

[7 Marks]

[PTO]

Question 4

a) 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. List the codes that are derived from the tree.

A:3 B:7 C:2 D:6 E:5 F:1 G:8

Resulting codes are *prefix-free.* Explain what this means with the use of a simple example.

[8 Marks]

b) Describe what is meant by the *Fibonacci Sequence* and discuss how the recursive calculation of the sequence can be greatly improved when a *dynamic programming* approach is applied. Provide code snippets or diagrams to support your discussion. Outline the complexity, in terms of O-notation, for both the implementation with and without the dynamic programming approach. Discuss how these were derived.

[7 Marks]

c) A society in the university want to create a simple program, to use on a small embedded device, for recording the names of attendees at an event. They cannot decide which underlying data structure to use from the following:

(i) Array
(ii) Singly Linked List
(iii) Doubly Linked List

Write a message to them outlining the pros and cons of each in the context of their application.

[6 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]