CS1260

UNIVERSITY OF WARWICK

First Year Examinations: Summer 2015

CS126 Design of Information Structures

Time Allowed: 1 hour 30 minutes.

Answer **SIX** questions, **ALL** questions in Section A and **TWO** questions from Section B.

Read carefully the instructions on the answer book and make sure the particulars required are entered on each answer book.

Calculators are **not** allowed.

Section A Answer **ALL** questions

1. (a) State *formally* what is meant by each of:

$f(n)$ is $O(g(n))$ [3 marks]
$f(n)$ is $\Omega(g(n))$ [2 marks]
$f(n)$ is $\theta(g(n))$ [2 marks]

(b) Determine whether each of the following statements is true or false, justifying your answers:

i. 2 is $O(1)$
ii. $n^2 \log(n)$ is $O(n^3)$
iii. n^2 is $\Omega(n \log(n))$ [6 marks]

2. (a) What is an abstract data type (ADT)? [2 marks]

(b) What is a data type? [2 marks]

(c) Define the Stack ADT. [4 marks]

(d) Describe one way in which a Stack may be used by a language *compiler* and illustrate your description with a short concrete example. [5 marks]

3. (a) Using pseudocode, describe the algorithm *bubble sort*. [6 marks]

(b) Illustrate your answer for part (a) by tracing the successive steps of bubble sort applied to the list (2,0,7,1,9,3) [4 marks]

(c) What is the time complexity of bubble sort? Justify your answer. [3 marks]

4. (a) When writing a program in Java, the programmer chooses to design the software by using a particular Abstract Data Type. When coding the program, the programmer discovers that a class exists within the Java Collection Framework (JCF) which appears to implement that ADT.

State *three* reasons why the programmer may choose to use the JCF class rather than implement the ADT directly. [3 marks]

(b) What are the differences between an *interface* and an *abstract class* in Java? [4 marks]

(c) How would a programmer use a Java *interface* in order to assist them coding an ADT? [6 marks]

Section B Answer **TWO** questions

5. (a) Describe, using pseudocode or otherwise, the sorting algorithm *quicksort*. [5 marks]

(b) Illustrate your answer to (a) by showing the intermediate lists produced while applying quicksort to the following values:

2 9 1 7 8 4 0 [3 marks]

(c) What is the worst-case time complexity for quicksort? Justify your answer. [4 marks]

(d) What is the average-case time complexity for quicksort? Justify your answer. [4 marks]

(e) Define the concept of *stability* as it applies to a sorting algorithm. Is quicksort stable? Explain your answer. [4 marks]

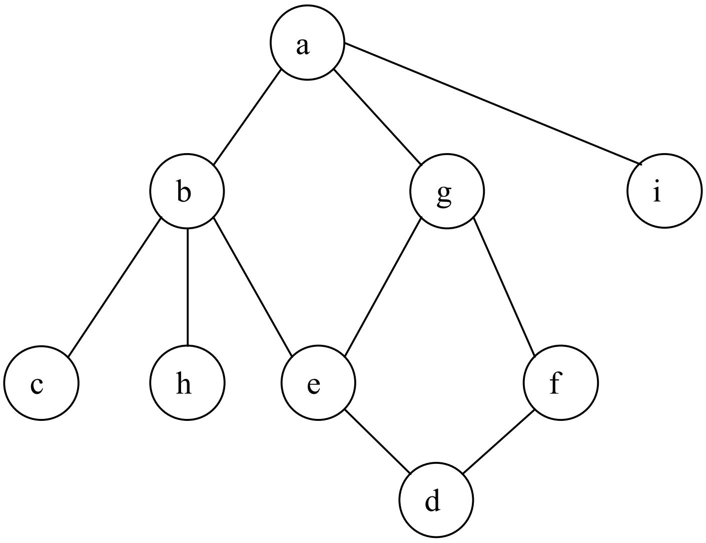
(f) Define the concept of *adaptivity* as it applies to a sorting algorithm. Is quicksort adaptive? Explain your answer. [4 marks]

6. (a) Describe the adjacency list and adjacency matrix structures for representing unweighted directed graphs. [8 marks]

(b) Give one example of an application for which an adjacency list representation would be appropriate, and one for which an adjacency matrix would be suitable. Justify your answers. [4 marks]

(c) Describe using pseudo-code the depth-first and breadth-first search methods which can be applied to graphs. [10 marks]

(d) Using the following graph state the order in which the nodes will be traversed for depth-first search (starting at "a"). [1 mark]

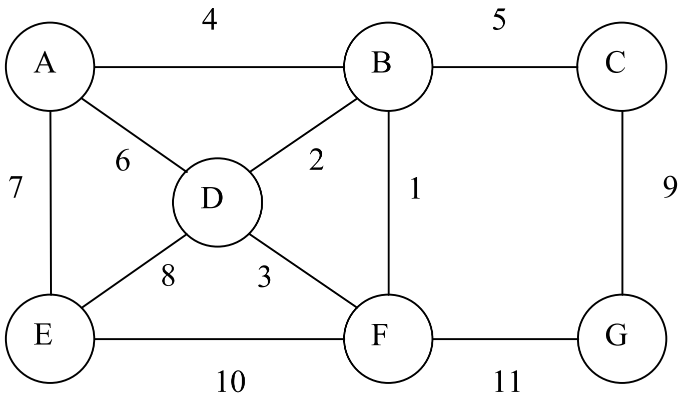


(e) Using the same graph as (d) state the order in which the nodes will be traversed for breadth-first search (starting at "a"). [1 mark]

7. (a) Define a *minimum spanning tree* M for a weighted graph G. [4 marks]

(b) Describe, using pseudo-code, Kruskal's Algorithm for calculating the minimum spanning tree for a weighted graph. State clearly any assumptions you make. [7 marks]

(c) For the following graph, find a minimum spanning tree using Kruskal's Algorithm. Show your working clearly. [8 marks]



(d) For a graph with n nodes and m edges, what is the worst case time complexity for Kruskal's algorithm? Justify your answer. [5 marks]

8. (a) In the context of the object-oriented programming language Java, explain the meaning of each of the following terms, using example code to illustrate your explanation. Note that you will not be penalised for minor syntactical errors in your examples.

(i)	Comparator	[4 marks]
(ii)	Iterator	[5 marks]
(iii)	Casting	[4 marks]
(iv)	Generics	[4 marks]

(b) Using the Queue ADT as an example, explain the role of an Interface in implementing an ADT using Java. [7 marks]