UNIVERSITY OF OTAGO EXAMINATIONS 2015

COMPUTER SCIENCE

Paper COSC242

ALGORITHMS & DATA STRUCTURES

Semester 2

(TIME ALLOWED: THREE HOURS)

This examination comprises 5 pages.

Candidates should answer questions as follows:

Candidates must answer **all** questions.
Each question is worth various marks and submarks are shown thus: (5)
The total number of marks available for this examination is 100.

The following material is provided:

Nil.

Use of calculators:

No calculators are permitted.

Candidates are permitted copies of:

Nil.

TURN OVER

1. **Complexity classes**

(a) Suppose you need to sort a nearly sorted input array of length n. Which of the following algorithms would you use: Quicksort or Insertion Sort? (1)

(b) Suppose you need to sort an input array of length n and memory is very tight. Which would you use: Insertion Sort or Merge Sort? (1)

(c) What is the worst case time complexity of Binary Search on a sorted input array of length n? (1)

(d) What is the worst case time complexity of Insertion Sort on an input array of length n? (1)

(e) Suppose you need to sort an input array of length $n = 10000$ and the keys in the array are small integers ranging in value from 0 to 1000. Which of the following algorithms is likely to be faster: Quicksort or Counting Sort? (1)

(f) What is the worst case time complexity of searching a perfect hash table? (1)

(g) What is the worst case time complexity of searching a Binary Search Tree with n nodes? (1)

(h) What is the worst case time complexity of searching a Red-Black Tree with n real nodes? (1)

(i) How many distinct subsets does a set of size n have? (1)

(j) How many permutations may be generated from a set of n items? (1)

2. **Recurrences, Big-O, Induction**

Use the iteration method to solve the recurrence equations

$f(1) = 3$
$f(n) = f(n-1) + 4.$

Use induction to prove that your solution is correct. (10)

3. **Sorting**

(a) Describe two ways to improve the performance of the basic Quicksort algorithm. (Stick to improvements discussed in class — don't invent your own.) (4)

(b) Show how Radix Sort would sort the keys 13, 12, 23, 15, 18, 28, 45, 62. (Show each stage of the sorting process, not just the sorted output.) (3)

(c) What does it mean to say that a sorting algorithm is stable? Give an example of a stable sorting algorithm. (3)

4. **Hash Tables**

(a) Given a table of size 7, a hash function h(k), and input keys $72, 37, 35, 51, 42, 64, 71$ (in that order), draw the hash table that results from:

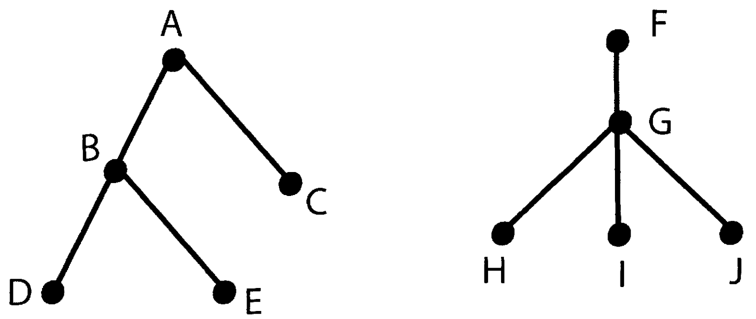
(i) Open addressing with double hashing. Use $h(k) = k\%7$ as the primary hash function, and $g(k) = 1 + (k\%6)$ as the secondary hash function. (5)

(ii) Chaining, with universal hash function $h_{(10,10)}(k) = ((10k + 10)\%101)\%7$. (5)

(b) Suppose you were using a perfect hashing scheme to create a hash table from the keys above. Would $h_{(10,10)}$ be acceptable as the primary hash function? Give your reasoning. (5)

5. **Trees**

(a) The diagram below shows a forest (a list of rooted ordered trees). Demonstrate your understanding of the correspondence between forests and binary trees by giving the binary tree that corresponds to this forest.



(3)

(b) Draw the final binary search tree T that results from successively inserting the keys $75, 64, 53, 42, 31$ into an initially empty tree. (1)

(c) Write down the keys of T in the order in which they would be visited during a postorder traversal. (1)

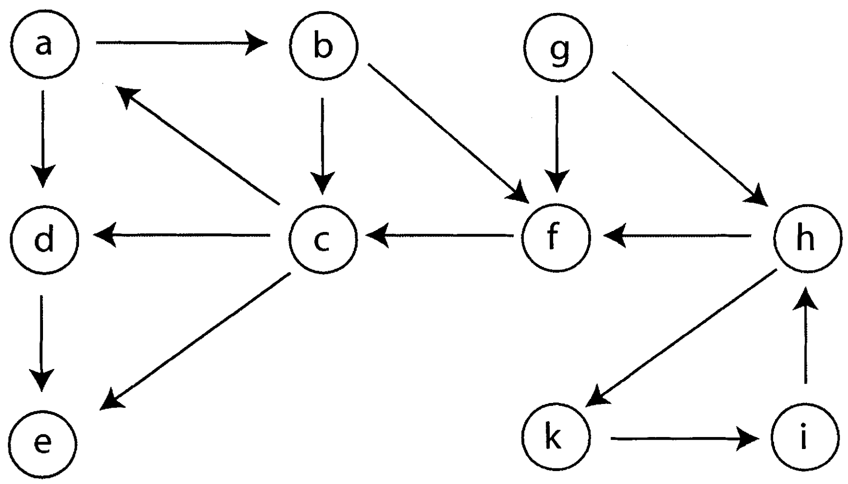
(d) Show all the red-black trees that result after successively inserting the keys $75, 64, 53, 42, 31$ into an initially empty red-black tree. State which cases apply. (6)

(e) Show the results of successively deleting 64 and 42 from the red-black tree of (d), giving details of your reasoning. (7)

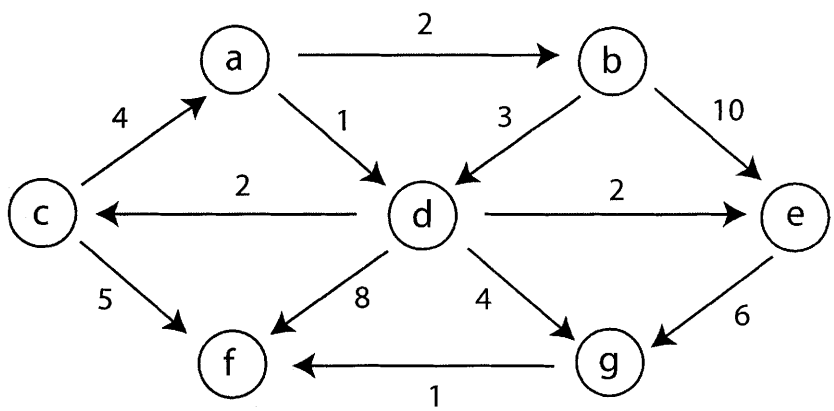
(f) By a 2-3-4 tree we mean a B-tree of minimum degree t = 2. Show the results of successively inserting the keys $6, 2, 4, 8, 7, 9, 10, 11, 12$ into an initially empty 2-3-4 tree. You should at least draw the trees just before some node must split and just after the node has split. (7)

6. **Graphs**

(a) Copy the following directed graph into your answer book. Starting at a, and considering adjacency lists to be alphabetically ordered, show how depth-first search would allocate time stamps to vertices, and label the edges with T, F, B, or C according to whether each is a tree edge, forward edge, back edge, or cross edge. (10)



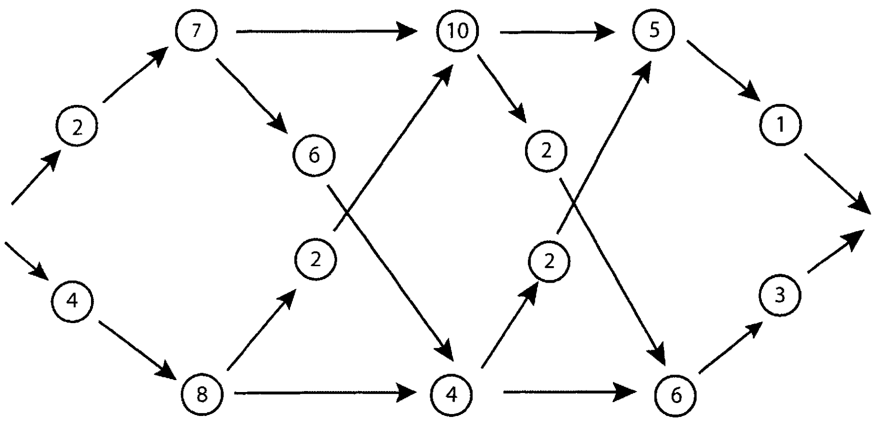
(b) Copy the following weighted directed graph into your answer book. Show how Dijkstra's algorithm would find the shortest paths from source a. Show clearly how the priority values change, and show the order in which vertices are extracted from the priority queue. Give a table showing vertices and their parents from which the shortest path to any vertex can be computed. (10)



TURN OVER

7. **Dynamic programming**

Consider the assembly line scheduling problem below. Give a dynamic programming solution. Show any bottom-up tables used in your solution and any calculations you perform. Explain what the entries in your tables mean. (5)



8. **P and NP**

In a few well-chosen sentences, explain what the classes P and NP are, and what it means to say that a problem is NP-complete. Give one example of an NP-complete problem. (5)

END