Student ID: ____________________

UNIVERSITY OF OTAGO EXAMINATIONS 2013

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.

Other instructions:

Please write your Student ID number at the top of this page.

At the end of the exam, hand in your completed paper attached to your answer book.

1. **Complexity classes**

(a) How many comparisons will Mergesort perform on an input array of length n? (1)

(b) What is the worst case time complexity of Quicksort on an input array of length n assuming we use our unmodified Partition algorithm? (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 searching a linked list of length n? (1)

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

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

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

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

(a) Show by using the iteration method that the recurrence equations

$T(1) = 1$

$T(n) = 2T(n/2) + n$

define the function given by $T(n) = n \log n + n$.

You do **not** need to prove that your solution is correct. You may assume values of n are powers of 2. (5)

(b) Using the definition of big-O and induction, prove that $n^2 = O(2^n)$. (6)

(c) Use proof by contradiction to show that $n^3 \neq O(n^2)$. (2)

3. **Sorting**

(a) Suppose you had to implement a sorting algorithm under conditions where memory is tight. Which of the following sorting algorithms could you use? Insertion Sort? Mergesort? Quicksort? (2)

(b) Now suppose you discover that your sorting algorithm would often be given nearly-sorted input. Which algorithm would you implement and why? (2)

(c) Name one algorithm that can sort integers in $O(n)$ time. What requirements must be satisfied in order for this algorithm to be used and to work efficiently? (3)

(d) Use Bucket Sort to sort the following keys: $0.92, 0.55, 0.86, 0.13, 0.52, 0.88, 0.25$. (3)

(e) What is the difference between a stable sort and an unstable sort? Which of the following sorts is stable? Insertion Sort? Quicksort? Counting Sort? (3)

4. **Hash Tables**

(a) Given a table of size 7, a hash function h(k), and input keys $86, 96, 93, 34, 11, 26, 73$ (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 $h_{(10,5)}(k) = ((10k + 5)\%101)\%7$. (5)

(iii) Cuckoo hashing, with $h(k) = k\%7$ as the primary hash function, and $h_{(10,5)}(k) = ((10k + 5)\%101)\%7$ as the secondary hash function. (5)

(b) Suppose you were using a perfect hashing scheme to create a hash table from the keys above. Would $h_{(10,5)}$ be acceptable as the primary hash function? Give your reasoning. (2)

5. **Trees**

(a) Draw the final binary search tree T that results from successively inserting the keys $86, 96, 93, 34, 11, 26, 73$ into an initially empty tree. (1)

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

(c) Show all the red-black trees that result after successively inserting the keys $86, 96, 93, 34, 11, 26, 73$ into an initially empty red-black tree. State which cases apply. (6)

(d) Show all the red-black trees that result from the deletion of 96. State which cases apply. (4)

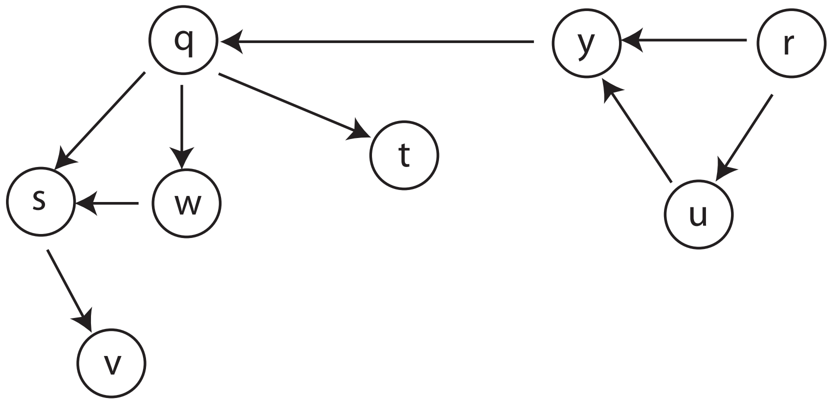
(e) By a 2-3-4 tree we mean a B-tree of minimum degree t = 2. Show the results of successively inserting the keys $1, 2, 3, 4, 5, 6, 7, 8, 9$ into an initially empty 2-3-4 tree. You should at least draw the trees just before some node must split and just after the node has split. (5)

(f) Show what happens when you delete key 5. (3)

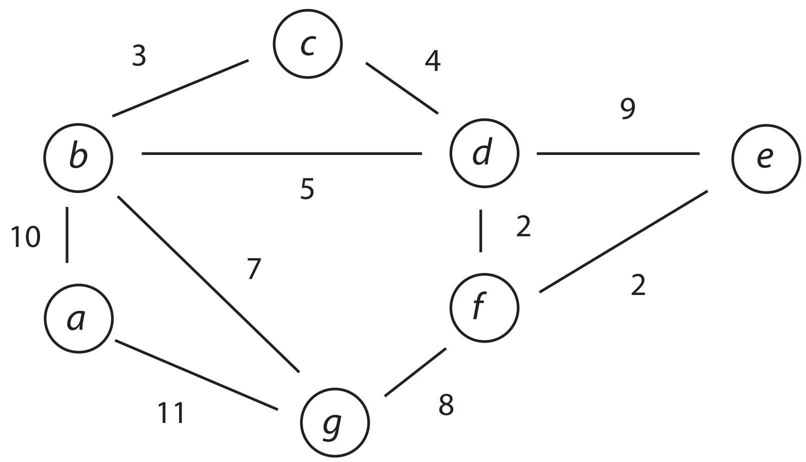
6. **Graphs**

(a) Give one reason why you might prefer to use an adjacency matrix representation of a graph, and one reason why you might decide to use an adjacency list representation. (4)

(b) Copy the following directed acyclic graph into your answer book. Starting at q, and considering adjacency lists to be alphabetically ordered, show how depth-first search would allocate time stamps to vertices, and then topologically sort the data. (8)

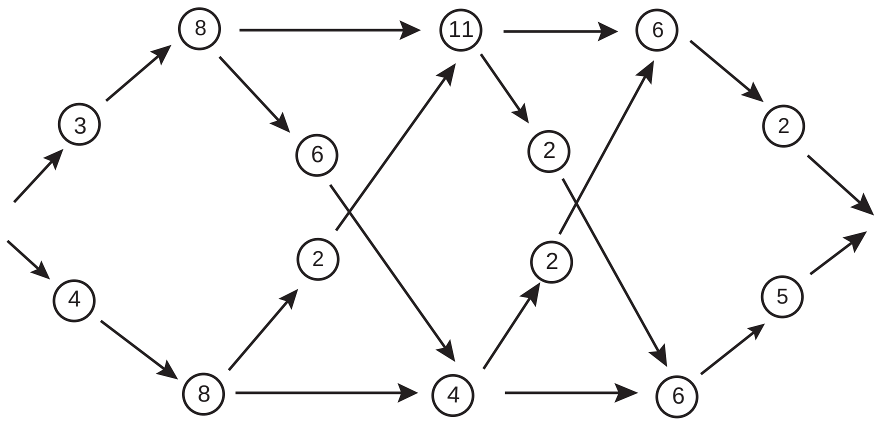


(c) Copy the following weighted undirected graph into your answer book. Show how Prim's algorithm would find the minimal spanning tree with root *a*. Show clearly how the priority values change, and show the order in which vertices are extracted from the priority queue. Give a table showing vertices and their parents from which the tree can be computed. (8)



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 to Aunt Maud what the classes P and NP are, and what it means to say that a problem is NP-complete. Give her one example of an NP-complete problem. (5)