THE UNIVERSITY OF MELBOURNE

School of Computing and Information Systems

COMP20003 Algorithms and Data Structures

End of Semester 2 2017

Reading Time: 15 min

Writing Time: 180 min

This paper has 5 pages including this page and any Appendices.

Student Number: _____

Authorised Materials
Calculators, Drawing Instruments, Books, and (Lecture) Notes: Not permitted
Dictionaries: Paper based language translation dictionaries are permitted provided they are not annotated in any way.

Instructions to Invigilators:

- The examination paper is to remain in the examination room.
- Students are to be provided with 1 script book.
- Provide extra script books on request.
- Students should write their ID number on the examination paper and on all script books.
- Students may not remove any part of the examination paper from the room and must return any used and unused script books with their examination paper.

Instructions to Students:

- This total marks for this paper is 60.
- Ensure your student number is written on the examination paper, on all script books and answer sheets during writing time
- There are 5 questions. You should attempt all questions.
- Answer all questions on the right-hand lined pages of the script book
- You may use the reverse side of the page to make notes or prepare draft answers. The reverse sides will not be looked at or marked unless you clearly indicate that you wish this to be your final answer.
- You may use pencil.
- Mobile phones, tablets, laptops, and other electronic devices, wallets and purses must be placed beneath your desk.
- All electronic devices (including mobile phones and phone alarms) must be switched off and remain under your desk until you leave the examination venue. No items may be taken to the toilet.

The paper is to be lodged with the Baillieu Library.

Programming

1. (18 points) This question is about the C programming language.

   Given an array of *positive* integers of size 5, the goal is to efficiently find the frequency of each integer in the array.

   For example, given the following array

```
int data[5] = {1, 1, 4, 2, 2, 2};
```

   the frequency array should contain

```
2, 3, 0, 1
```

   as 1 appears twice, 2 appears three times, and 4 once.

   Before you write the code, take some time to think of the most efficient solution possible.

   (a) (5 points) Write the function

```
void find_range_and_min(int* data, int* min, int* range)
```

   Given an array finds the range and the minimum value. In the example above the range is 3 and the minimum value is 1. The range is defined as the difference between the *maximum* number and the *minimum* number. Assume that the data array is given as a pointer, the pointer to min will save the minimum value, and the pointer to range will save the range.

   (b) (5 points) Write the function

```
int* count_freq(int *data, int range, int min)
```

   that given an array returns the frequency counts. You have to create a new array of size `range`, and map each integer of data to the correct position in the frequency array using the `min` value. Return the frequency counts.

   (c) (5 points) Write the main function, use a static array called data, initialize it with the example data array shown above and print the frequency counts as shown above using your implemented functions. Last but not least, free your memory if needed. You do not need to write the `include` statements.

   You should write the function in your script book. Include brief comments in the code to explain your reasoning and help with readability. 3 marks of your code will be based on best practice use of C (1 mark), readability and comments (2 marks).

Computational Complexity

2. (6 points) This question is about computational complexity

   For the remaining parts of this question you are required only to give an answer. You are not required to explain or justify your answers in this section. Each correct answer gives 1 point.

   (a) (1 point) What is the time complexity of inserting one item into a sorted array of n items? Give the closest upper bound in big-O notation.

   (b) (1 point) What is the time complexity of obtaining the maximum element in a max-heap of n items and fixing the *heap condition*? Give the closest upper bound in big-O notation.

   (c) (1 point) What is the time complexity for building a heap of n items if you insert all items in an unsorted array and then perform $n/2$ *downheap* operations? Give the closest upper bound in big-O notation.

   (d) (1 point) What is the time complexity of searching for one item when collision resolution is achieved with chaining in a hashtable containing n items. Give the closest upper bound in big-O notation.

   (e) (1 point) What is the *extra* space complexity of sorting an array of n items using mergesort? Give the closest upper bound in big-O notation.

   (f) (1 point) What is the *extra* space complexity of sorting an array of n items using distribution counting, where the range of values of the items is r? Give the closest upper bound in big-O notation.

Advanced Data Structures

3. (10 points) This question is about the practical usage of several data structures.

   (a) (3 points) You are given a graph G and you want to implement a traversal which is non-recursive. Explain which data-structures you need to implement a *Breadth First Traversal*, a *Depth First Traversal* and *Dijkstra*. Justify your answer to get full marks.

   (b) (2 points) Why is it better to implement a Priority Queue with a Heap that uses an Array, than a Heap that uses a Linked List? Justify your answer in terms of the big-O complexity of *downheap* and *upheap* operations.

   (c) (5 points) Dijkstra's algorithm uses a priority queue to choose the next vertex to look at. Assume the priority queue is implemented using a *hash table*, with *chaining* for collision resolution, and using the *best known distance* as the hashkey of each vertex `hashkey(v) = dist[v]`. The maximum distance to any vertex in the graph is known as `max_dist`, hence the hash table size is `m = max_dist + 1`:

i. (2 points) Write in pseudocode how to initialize the array of integers `dist`. Remember that `dist` should represent the initial distances from the source vertex s to any other vertex $v \in V$. Note as well that `dist` represent upper-bounds on the shortest path distances.

ii. (2 points) Give the least upper bound big-O complexity of `deletemin(P)` operation. Explain your answer *briefly*.

iii. (1 point) If the maximum distance to a vertex $v \in V$ is unknown, and we fix the size of the hashtable to be constant `m = 101`, how should we define the `hashkey` function?

Graphs

4. (16 points) This question is about graphs and graph algorithms.

   (a) (3 points) A new NASA mission to send a robot to Mars wants to precompute all the shortest paths on Earth in order to minimise the battery drain. The mission engineers represent the points-of-interest to visit and the roads as a graph $G(V, E)$. Which Algorithm should they use? Justify your answer and give the worst case behavior in big-O notation.

   (b) (4 points) Which of the following are true and which are false? *Explain your answers for full marks*

      i. (1 point) One Depth-first traversal is sufficient to visit all the vertexes in a fully connected acyclic graph.

      ii. (1 point) Dijkstra will always find the shortest path over weighted DAGs, when weights are reals that range from 0 to 1.

      iii. (1 point) Using a Binary Heap as a priority queue, the worst case big-O complexity of Prim's algorithm is $O(E + (V \log V))$.

      iv. (1 point) You can prove that a problem A is *NP-Hard* if every problem in *NP* can be reduced in exponential time to A.

   (c) (5 points) Give an example of a graph with *more than one connected component*, and design an algorithm to compute the number of connected components.

   (d) (4 points) Give an example of a graph such that running Dijkstra and Prim's algorithm would yield different solutions. Explain why these algorithms return different solutions

Searching and Sorting

5. (10 points) This question is about searching, sorting and balanced trees.

   (a) (3 points) Given a list of sorted integers, mergesort and quicksort can have the same worst-case performance. Give the Big-O performance of both algorithms and explain how quicksort should chose pivots.

(b) (3 points) You are given records with the following keys, to be stored in a hash table

1 2 4 8 16 32 64 128 256 512

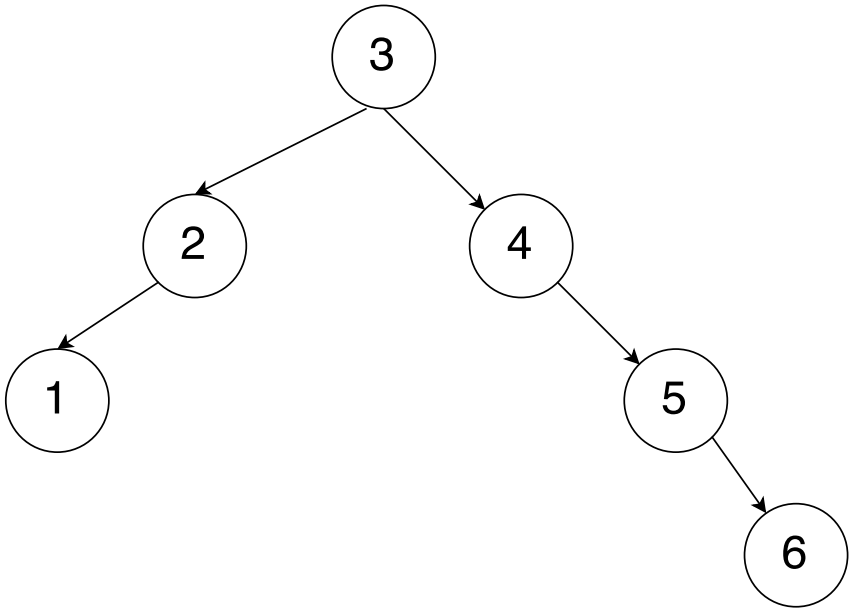
The hash table contains a maximum of 10 items, and the hash function `hash(key)` returns $key\%10$. Linear probing is used for collision resolution.

i. (1.5 points) Show the hash table after insertion of these records

ii. (1.5 points) Show the hash table if double probing is used to insert the keys, using `hash2(key)`=$key\%2 + 1$

(c) (4 points) The rotation operation is used in AVL trees, red-black trees, and splay trees to adjust the balance of a binary tree.

In AVL tree shown below, you are to perform a single rotation to balance the tree.



Explain which rotation is needed and justify your choice. Draw the resulting balanced tree.