EXAMINATION PAPER

Examination Session:	Year:	Exam Code:
May/June	2017	COMP1081-WE01

Title: Algorithms and Data Structures

Time Allowed:	2 hours	
Additional Material provided:	none	
Materials Permitted:	none	
Calculators Permitted:	yes	Models Permitted: Casio fx-83 GTPLUS or Casio fx-85 GTPLUS
Visiting Students may use dictionaries:	yes	

Instructions to Candidates:	Answer FOUR questions (TWO from EACH section).

Revision:	

Algorithms and Data Structures (COMP1081 WE01)

Section A Algorithms and Data Structures (Dr Robert Powell and Dr Tom Friedetzky)

Question 1

(a) Define the data structure **stack** and the associated operations **push**, **pop**, **top** and **isEmpty**. **[6 Marks]**

(b) Given an initially empty stack, what is output as the following sequence of operations is executed?

push(5), push(2), push(3), pop, push(9), isEmpty, pop, pop, top, push(4).

Moreover, what does the stack contain after the sequence of operations above has been executed? **[6 Marks]**

(c) Briefly explain how an array can be used to implement a stack. State one advantage and one disadvantage of this implementation. **[5 Marks]**

(d) Let S be a non-empty stack, and Q be an empty queue. Describe a sequence of operations that will reverse the stack S.

In your procedure only the data structures S and Q are available, and only the operations push, pop and isEmpty (on the stack) and enqueue, dequeue and isEmpty (on the queue) can be used. **[8 Marks]**

continued

Question 2

(a) When considering the relations on real numbers it is well known that $a \geq b$ implies either $a > b$ or $a = b$. Is the same true for functions, that is, is it true that for $f, g : \mathbb{N} \to \mathbb{R}^+$ we have that $f = \Omega(g)$ implies that either $f = \omega(g)$ or $f = \Theta(g)$? If you think the answer is "yes" then argue your case, and if you think the answer is "no" then provide a counterexample. In either case, carefully explain your reasoning. **[10 Marks]**

(b) Consider a variant of MergeSort that splits the input not into two but three equal-sized parts.

i. For the two-way MergeSort we said that we may assume that, without loss of generality, the input size is a power of two. Argue that we may now assume that without loss of generality the input size is a power of three. **[3 Marks]**

ii. Explain how you would implement the three-way MergeSort function and the corresponding three-way Merge function, assuming the input size is a power of three. You do not need to provide code. **[4 Marks]**

iii. State the recurrence that expresses the running time of three-way MergeSort (think carefully about the running time of your three-way Merge function). **[3 Marks]**

iv. Solve the recurrence from (iii). You may use any method. **[5 Marks]**

If you wish to use the Master Theorem, here is its statement. Suppose a recurrence is of the form $T(n) = aT(n/b) + f(n)$ for $a \geq 1$ and $b > 1$.

- **If** $f(n) = O(n^{\log_b(a)-\epsilon})$ for some constant $\epsilon > 0$ **then** $T(n) = \Theta(n^{\log_b(a)})$.
- **If** $f(n) = \Theta(n^{\log_b(a)})$ **then** $T(n) = \Theta(n^{\log_b(a)} \log n)$.
- **If** $f(n) = \Omega(n^{\log_b(a)+\epsilon})$ for some constant $\epsilon > 0$ **and** if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all n large enough **then** $T(n) = \Theta(f(n))$.

End of Section A

continued

Section B Algorithms and Data Structures (Dr Robert Powell and Dr Tom Friedetzky)

Question 3

(a) Suppose we have integer values between 1 and 100 in a binary search tree and search for 28, using the standard lookup method. Which of the following **cannot** be the sequence of keys examined? Explain your answer. **[5 Marks]**

i. $17, 56, 24, 30, 28$

ii. $93, 70, 62, 73, 28$

iii. $56, 34, 20, 31, 28$

iv. $13, 91, 20, 43, 28$

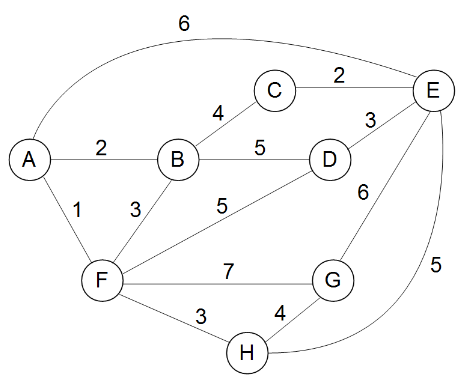
v. $10, 20, 30, 29, 28$

(b) Provide a formal definition of, and an explanation of the meaning behind, the asymptotic class $o()$ (little-oh). **[5 Marks]**

(c) Explain why (formally speaking) writing $f(n) = \mathcal{O}(g(n))$ is nonsense. **[2 Marks]**

(d) Define a minimum spanning tree (MST) of a connected undirected graph with weights on the edges. **[2 Marks]**

(e) Find a minimum spanning tree of the graph below using Kruskal's algorithm. State the edges of the tree in the order they are added to the MST. **[5 Marks]**



(f) Let G be a connected undirected graph with each edge having a unique weight. Show that there is only one minimum spanning tree of G. **[6 Marks]**

continued

Question 4

(a) Argue that since sorting n elements takes $\Omega(n \log n)$ time in the worst case in the comparison model, any comparison-based algorithm for constructing a binary search tree from an arbitrary list of n elements takes $\Omega(n \log n)$ time in the worst case. **[6 Marks]**

(b) What are the minimum and maximum possible numbers of elements in a heap of height h (recall, the height is the maximum number of edges on any path from the root to any leaf)? **[6 Marks]**

(c) A hash function is the composition of two functions. State and briefly describe these two functions. **[4 Marks]**

(d) Draw the hash table that results from applying the hash function

$$h(k) = (2k + 3) \text{ mod } 11$$

to the keys 1,6,4,17,8,7,12. Assume the hash table has 11 buckets and that collisions are handled using Quadratic Probing. **[5 Marks]**

(e) Explain the purpose of tombstones with regard to deleting elements from a hash table. You should also describe how their use impacts on look-ups and insertions. **[4 Marks]**

continued

Question 5

(a) Professor Quick thinks he has discovered a remarkable property of binary search trees. Suppose that the search for key k in a binary search tree ends up in a leaf. Consider three sets: A, the keys to the left of the search path; B, the keys on the search path; and C, the keys to the right of the search path. Professor Quick claims that any three keys $a \in A$, $b \in B$, and $c \in C$ must satisfy $a \leq b \leq c$. Is the professor's claim true? If you think the answer is yes then provide a proof, otherwise give a small counterexample.

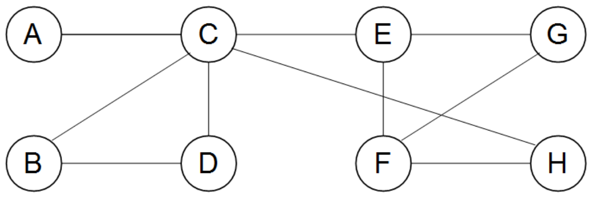
[5 Marks]

(b) In a binary search tree, is the operation of deletion commutative in the sense that deleting x and then y from a binary search tree always leaves the same tree as deleting y and then x? Argue why it is or give a counterexample.

[8 Marks]

(c) The following questions all concern breadth-first search (BFS).

i. Given the undirected graph below, perform a breadth-first search using node A as the source. You should calculate the distance array and the predecessor array, and draw the breadth-first tree.



[6 Marks]

ii. What do the values in the distance array and predecessor array represent?

[2 Marks]

iii. Assume it takes constant time to make changes to elements in the distance and predecessor arrays, and to enqueue and dequeue vertices that have been discovered and are waiting to be processed. Prove the upper bound on the running time for BFS is $\mathcal{O}(V + E)$.

[4 Marks]

END OF PAPER