Durham University

EXAMINATION PAPER

Examination Session:	Year:	Exam Code:
May/June	2016	COMP1081-WE01

Title:

ALGORITHMS AND DATA STRUCTURES

Time Allowed:	2 hours	
Additional Material provided:	None	
Materials Permitted:	None	
Calculators Permitted:	No	Models Permitted:
Visiting Students may use dictionaries: Yes		

Instructions to Candidates:	Answer FOUR questions. (TWO from Section A, TWO from Section B) PLEASE ANSWER EACH SECTION IN A SEPARATE ANSWER BOOKLET

Revision:

Section A Dr. Stefan Dantchev and Dr. Tom Friedetzky

Question 1

(a) Explain what a queue data structure is. (When doing so, think of it as a black-box, i.e. avoid implementation-related details.) **[4 Marks]**

(b) Given a queue and an element, which is distinct from everything that the queue contains, give an algorithm that

i. counts the elements in the queue and does not change the queue; **[2 Marks]**

ii. puts the end element right at the beginning but does not change the order of the others; **[2 Marks]**

iii. as above in (ii), but you are only given two extra variables, each of which can hold a single element from the queue (or the extra distinct element) and nothing else. **[4 Marks]**

(c) Explain what a stack data structure is. (Again, think of it as a black-box, and avoid implementation-related details.) **[4 Marks]**

(d) Explain how you can simulate a queue by two stacks. **[4 Marks]**

(e) Explain how you can simulate two separate stacks by a single queue. You may assume that you have special constants (elements which are different from anything that normally the stacks contain) and a few extra variables, each of which can hold a single element, including one of the special ones. **[5 Marks]**

continued

Question 2

(a) Binary search trees:

i. Explain the BST property. **[2 Marks]**

ii. Explain how to insert a node into a BST. **[2 Marks]**

iii. Explain how to remove a node from a BST. **[3 Marks]**

iv. Draw a BST after the following sequence of operations:

insert 50 25 40 30 100 200 150 70 27
delete 200
insert 10 5 15
delete 25

[3 Marks]

You do not have to provide code for ii. and iii., nor do you have to go into detail as to how to fix the pointers (you may simply say something like "*make the new node a right child of this node*").

(b) Explain briefly how *recursive binary search* works. What conditions on the input do you need to have for it to work, and why? How can you eliminate the recursion? Justify your answers. **[5 Marks]**

(c) Heaps:

i. Explain the *heap* property. **[2 Marks]**

ii. Explain how a heap can be represented in an array. **[2 Marks]**

iii. Draw the heap corresponding to the array

$$A = [22, 13, 10, 8, 7, 6, 2, 4, 3, 5].$$

[2 Marks]

iv. Explain how the heapify function works. **[4 Marks]**

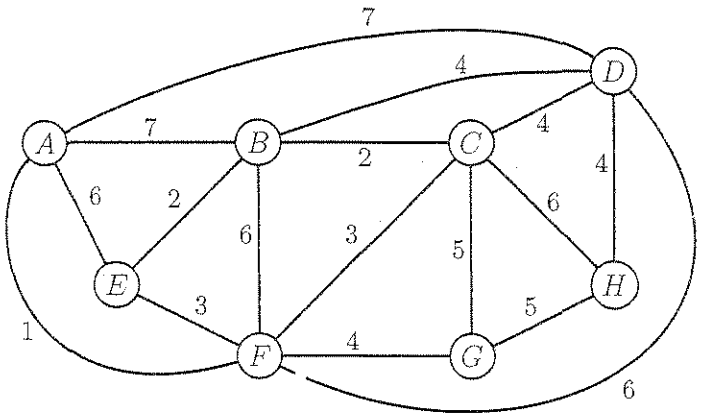
Section B Dr. Tom Friedetzky and Dr. Matthew Johnson

Question 3

(a) From a depth-first search of a directed graph G, a depth-first tree T is obtained that contains the edges traversed during the search. Describe how the edges of G can be classified with respect to T as tree, forward, back or cross edges. **[4 Marks]**

(b) Suppose that G is a directed graph that contains at least one directed cycle and that the edges of G have been classified with respect to a depth-first tree. Is it possible that every edge of G is either a tree edge or a cross edge? Explain your answer. **[5 Marks]**

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



this question is continued on the next page

(d) Suppose we have integer values between 1 and 1000 in a binary search tree and search for 363, using the standard lookup method. Which of the following **cannot** be the sequence of keys examined? Explain your answer.

[5 Marks]

i. 2 252 401 398 330 363

ii. 399 387 219 266 382 381 278 363

iii. 3 923 220 911 244 898 258 362 363

iv. 4 924 278 347 621 299 392 358 363

v. 5 925 202 910 245 363

(e) Let S be an **unsorted** array of $2n$ integers. Give an algorithm that partitions the numbers into n pairs, with the property that the partition minimises the maximum sum of any pair.

For example, say we are given the numbers $[1, 3, 5, 9]$. The possible partitions are $((1, 3), (5, 9))$, $((1, 5), (3, 9))$, and $((1, 9), (3, 5))$. The pair sums for these partitions are $(4, 14)$, $(6, 12)$, and $(10, 8)$. Thus the third partition has 10 as its maximum sum, which is the smallest maximum sum over the three partitions.

Your algorithm must run in $O(n \log n)$ worst-case time. You do **not** need to formally prove the correctness of your algorithm, a brief justification is sufficient. **[6 Marks]**

Question 4

(a) For each of the following problems, give an algorithm that finds the desired numbers within the given amount of time. You do not need to provide pseudo-code, and your answers do not need to be very detailed: something such as "sort the numbers using algorithm A and return the 5th smallest as x and the 17th largest as y" is acceptable. A brief justification of the correctness of your solution is sufficient; no formal proofs are required.

i. Let S be an **unsorted** array of n distinct integers. Give an algorithm that finds the pair $x, y \in S$ that maximises $|x - y|$. Your algorithm must run in $O(n)$ worst-case time. **[3 Marks]**

ii. Let S be a **sorted** array of n distinct integers. Give an algorithm that finds the pair $x, y \in S$ that maximises $|x - y|$. Your algorithm must run in $O(1)$ worst-case time. **[3 Marks]**

iii. Let S be an **unsorted** array of n distinct integers. Give an algorithm that finds a pair $x, y \in S$ that minimises $|x - y|$, for $x \neq y$. Your algorithm must run in $O(n \log n)$ worst-case time. **[3 Marks]**

iv. Let S be a **sorted** array of n distinct integers. Give an algorithm that finds a pair $x, y \in S$ that minimises $|x - y|$, for $x \neq y$. Your algorithm must run in $O(n)$ worst-case time. **[3 Marks]**

(b) Is $2^{2n} = O(2^n)$? Justify your answer. **[2 Marks]**

(c) Suppose a depth-first search is run on the directed graph on vertex set $V = \{1, \ldots, 8\}$ given by the adjacency matrix below, with vertex 1 as the source. Classify each edge of the graph as a tree, forward, back or cross edge. **[6 Marks]**

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

this question is continued on the next page

(d) Let G be an undirected graph with weights on the edges. Let T_1 and T_2 be two distinct minimum spanning trees of G. Show that there is an edge e_1 in T_1 but not T_2 and an edge e_2 in T_2 but not T_1 such that $\{T_1 - e_1\} \cup e_2$ is also an MST. **[5 Marks]**

Question 5

(a) A breadth-first search on an undirected graph uses two arrays: a distance array and a predecessor array. When the search is complete, what do the values in these arrays represent? **[2 Marks]**

(b) We want to modify breadth-first search so that when the algorithm terminates, each vertex in the breadth-first tree is labelled either *odd* or *even*, and

- the source vertex s is labelled *odd*, and
- the label of each vertex differs from that of all its neighbours in the breadth-first tree (that is, if a vertex is labelled *odd* all its neighbours in the tree are labelled *even* and vice versa).

The label of s can be set at the start of the search. Describe how the label of every other vertex can be set when it is first discovered by the search. **[3 Marks]**

(c) Suppose that the modified breadth-first search is run on a connected undirected graph G. Prove or disprove the following: G contains a cycle of odd length (that is, containing an odd number of edges) if and only if there are vertices u and v that are adjacent in G and have the same label (that is, are both labelled odd or both labelled even). **[8 Marks]**

(d) Briefly explain how you might design a deterministic QuickSort with worst-case running time $O(n \log n)$. **[5 Marks]**

this question is continued on the next page

(e) Recall the Master Theorem:

- **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** $T(n) = \Theta(f(n))$

For each of the following recurrences, give an expression for the runtime $T(n)$ if the recurrence can be solved with the Master Theorem. Otherwise, indicate that the Master Theorem does not apply.

i. $T(n) = 3T(n/2) + n^2$

ii. $T(n) = 4T(n/2) + n^2$

iii. $T(n) = T(n/2) + 2^n$

iv. $T(n) = 2^n T(n/2) + n^n$

v. $T(n) = 16T(n/4) + n$

vi. $T(n) = 2T(n/2) + n \log n$

vii. $T(n) = 2T(n/2) + n/\log n$

[7 Marks]

END OF PAPER