THE HONG KONG UNIVERSITY OF SCIENCE & TECHNOLOGY
Department of Computer Science

COMP 171: Data Structures and Algorithms
Spring 1999

Midterm 1 Examination (Cream)

Synnöve Kekkonen-Moneta & Derick Wood

Date: March 17, 1999. Time: 19:30–21:30

Place: LT A, B and D (allocated by tutorial section)

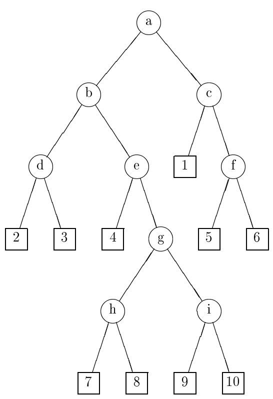
This is a closed book examination.
No calculators.
Each question tells you how many marks and how many parts it has.
This booklet has 9 pages including the cover page, please check it.

Student name: ____________________ **Student ID:** ____________________

Problem	Your mark	Maximum	Problem	Your mark	Maximum
1		16	5		14
2		14	6		15
3		14	7		15
4		12			
Subtotal		56	Subtotal		44

Your total marks: __________

1. (16 marks) (4 parts) **Trees and traversals**
Answer the following four questions based on the tree:



(a) What is the height of the given tree?

(b) Assuming that "visit a node" means "output the node's label" give the output from the inorder traversal of the tree that visits both internal and external nodes.

(c) Assuming that "visit a node" means "output the node's label" give the output from the reverse inorder traversal of the tree that visits only internal nodes.

(d) Assuming that "visit a node" means "output the node's label" give the output from the bottom-up level-order traversal of the tree that visits only external nodes.

2. (14 marks) (4 parts) **Priority tree representation of priority queue**
Consider max-priority trees of size 5 and assume that "one data move" means "rewriting a node's content (or label)".

(a) What is the maximum possible number of data moves for `deletemax()`?

(b) Give an example of a max-priority tree of size 5 that requires the maximum possible number of data moves for `deletemax()`. Draw the priority tree before and after `deletemax()`.

(c) What is the minimum possible number of data moves for `deletemax()`?

(d) Give an example of a max-priority tree of size 5 that requires the minimum possible number of data moves for `deletemax()`. Draw the priority tree before and after `deletemax()`.

3. (14 marks) (2 parts) **Heap representation of priority queues.**

(a) You are given the following max heap H

7	7	6	5	1	3	2

i. Give H after one `deletemax()`.

ii. Give H after another `deletemax()`.

(b) You are given the following max heap H

7	7	6	5	1	3	2

i. Give H after `insert(11)`.

ii. Give H after another `insert(11)`.

4. (12 marks) (4 parts) **Mathematics for sorting and searching.**
For each of the given binary relations over $A = \{a, b, c\}$, indicate by writing Yes or No in the appropriate cell whether the relation is:

- reflexive (R)
- symmetric (S)
- antisymmetric (AS)
- transitive (T)
- a partial order (PO)
- a total order (TO)

Number	Relation	R	S	AS	T	PO	TO
(a)	$R = \{\}$						
(b)	$R = \{(a,a),(b,b),(c,c)\}$						
(c)	$R = \{(a,a),(b,b),(c,c),(a,c),(c,b)\}$						
(d)	$R = \{(a,a),(b,b),(c,c),(a,c),(a,b),(c,b)\}$						

5. (14 marks) (4 parts) **Elementary sorting methods.**

(a) Sort the following array using insertion sort and give a snapshot of the array after each pass of the insertion sort algorithm. (In other words, after each execution of the outer loop of the algorithm.)

13	12	10	8	1	9	11

(b) How many key comparisons did you use in part a?

(c) Merge the following two sorted sequences A and B using the merging algorithm to produce one sorted sequence C.
The sequence A is

2	7	9	13

The sequence B is

5	8	11

What is the sequence C?

(d) How many key comparisons did you use in part c?

6. (15 marks) (2 parts) **Performance analysis.**
You have carried out a performance analysis of two algorithms A and B. You have discovered that:

- The time function f_A for algorithm A is $f_A(n) = 2^n$, for all $n \geq 1$, in the best, average/expected, and worst cases.
- The time function f_B for algorithm B is $f_B(n) = 2n + 10^6$, for all $n \geq 1$, in the best, average/expected, and worst cases.
- For 95 percent of the time, the input size is $n = 15$ ($2^{15} = 32768$).

(a) Which algorithm would you choose based on analytical considerations? Briefly justify your answer.

(b) Which algorithm would you choose based on practical considerations? Briefly justify your answer.

7. (15 marks) (2 parts) **Performance analysis.**
The definition of big oh is: *A function f is big oh of a function g if there are a constant c and an integer n_0 such that, for all $n \geq n_0$, $f(n) \leq cg(n)$. We write f is $O(g)$ in this case.*

(a) You are given the two functions:

$$
\begin{aligned}
f(n) &= 20n + 15 \\
g(n) &= n^3.
\end{aligned}
$$

Is f $O(g)$?

- If your answer is "yes", then give the values for n_0 and c such that, for all $n \geq n_0$, $f(n) \leq cg(n)$.
- If your answer is "no", then explain briefly in your own words why not. In other words, you do not need to formally prove the result.

(b) You are given the two functions:

$$
\begin{aligned}
f(n) &= \begin{cases} n, & \text{if } n \text{ is even,} \\ n^4, & \text{if } n \text{ is odd;} \end{cases} \\
g(n) &= n^3.
\end{aligned}
$$

Is f $O(g)$?

- If your answer is "yes", then give the values for n_0 and c such that, for all $n \geq n_0$, $f(n) \leq cg(n)$.
- If your answer is "no", then explain briefly in your own words why not. In other words, you do not need to formally prove the result.