UNIVERSITY OF TORONTO
Faculty of Arts and Sciences

AUGUST EXAMINATIONS 2004

CSC 263 H1 Y

Duration — 3 hours

No Aids Allowed

Name ______________________________ Student No. ______________________________

Answer ALL questions on test paper. Total pages: 20 (including the cover and scratch pages).

Total Points: 104

Question 1 (17 points): ____________

Question 2 (20 points): ____________

Question 3 (12 points): ____________

Question 4 (22 points): ____________

Question 5 (13 points): ____________

Question 6 (20 points): ____________

(1) (a) (13 points) Each row of the following table represents a data structure for the dictionary ADT. Fill in each square with the worst-case complexity (in Θ-notation) of the given operation in the given data structure. Assume n is the number of elements in the dictionary. For hashing, assume m is the number of hash table entries and that collisions are resolved by chaining in a doubly-linked list. For trees, assume each node has a pointer to its children and its parent. Recall that the semantics of DELETE are that you are given a pointer to the element that is to be deleted. A couple of the squares have been filled in as examples:

DATA STRUCTURE	SEARCH	INSERT	DELETE
Unsorted Singly Linked List		$\Theta(1)$, WC	
Binary Search Tree			
Red-Black Tree			
Direct Addressing			
Closed-Address Hashing			$\Theta(1)$, WC

(b) (4 points) Give the following running times. Give exact answers unless otherwise indicated.

* The average case running time of quicksort over the space of permutations of $(1, \ldots, n)$ where each permutation is equally likely (use Θ-notation):

* The average case running time of SEARCH in a closed-address hash table assuming simple, uniform hashing. Let n be the number of elements in the table and m the number of slots:

cont'd...

* The amortized complexity of `INCREMENT` in a binary counter (in terms of the number of bits that are changed).

* The worst-case running time of `EXTRACT-MIN` in a heap with n elements (in Θ-notation):

cont'd...

(2) (a) (5 points) Draw all valid heaps (in tree form) for elements with the following priorities: 3,8,10,11,14.

cont'd...

(b) (5 points) Let S be a sample space (probability space) whose elements are the heaps from part (a), where each heap is equally likely. What is the average case running time of `INSERT(9)` over the space S? Measure the running time in terms of the number of "swaps" that are performed.

(c) (5 points) Let $n = 2^k - 1$ for some $k \geq 1$. Let $T(n)$ be the number of different heaps for n elements with distinct priorities. Write a recurrence relation for $T(n)$. Make sure to specify the value of $T(1)$.

cont'd...

(d) (5 points) Solve the recurrence relation from part (c).

cont'd...

(3) When using hash tables, it is often a good idea to make sure that the size of the table is a prime number. This question is about implementing dynamic arrays (which are important for the implementation of hash tables) where we ensure that the array's size is always a prime number. To keep things simple, there will be only one operation on the array: `APPEND`. The array will start out with size 2 (a prime number). Whenever we call `APPEND(x)` and there is space in the array, x gets inserted at the leftmost open slot (just like in lecture). If there is not space in the array, we do the following (let A be the current full array and let m be its size):

```
m' := FindPrime(m);
Allocate new array A' of size m';
Copy the elements of A to the first m slots of A';
Insert x into slot m+1 of A';
m := m';
A := A';
```

The procedure `FindPrime(m)` returns a prime number m' such that $2m \leq m' \leq 4m$ and its running time is $p(m)$.

(a) (2 points) We will measure the cost of `APPEND` in terms of the number of elements it writes (not reads), plus the amount of time that `FindPrime` takes. A normal `APPEND` where the array does not grow therefore costs 1. How much does `APPEND` cost when the array does grow?

cont'd...

(b) (8 points) We are interested in the worst-case sequence complexity of n APPENDs ($WCSC(n)$). Assume that $p(m) = km$ for some constant k. Using the accounting method, how much should you charge for each APPEND and what will be the credit invariant?

(c) (2 points) Given (b), compute $WCSC(n)$.

cont'd...

(d) *(Extra extra credit–seriously, don't do this part unless you are done with everything else and you have nothing better to do tonight):* Show that, for any natural number $x \geq 2$, there is a prime number between x and $2x$.

cont'd...

(4) A real estate agency maintains information about a set of houses for sale. For each house it maintains a record (p, s), where p is the price and s is the floor area of the house. Each record of this type is called a *listing*. The listings must be kept in a data structure D that supports efficient implementation of the following operations:

- `INSERT(D,x)`: Insert a new listing x into D.
- `DELETE(D,x)`: Delete a listing x from D.
- `MAXAREA(D,p)`: Return a listing from D whose price is at most p and whose area is maximal subject to this price limit. If there is no listing whose price is at most p, then return `NIL`.

In this question, you must augment Red-Black Trees so that `INSERT` and `DELETE` run in time $O(\log n)$ and so that `MAXAREA` runs a quickly as possible.

(a) (4 points) What part of a listing will serve as the key value? What extra information will you store at each node?

cont'd...