STUDENT NAME: _____
STUDENT NUMBER: _____

FACULTY OF SCIENCE
FINAL EXAMINATION

COMPUTER SCIENCE COMP 250

Introduction to Computer Science

Examiner: Prakash Panangaden — Friday, 29 April 2005

Associate Examiner: Prof. Mathieu Blanchette — 2:00 - 5:00

Instructions:

This exam has 9 questions. Please answer all questions. You may use any books or notes or dictionaries that you have. You have three hours in all. Writing Java code instead of an algorithm will be penalized. By this I mean writing Java code for input and output; you should just say what is input and what is output. Making unwarranted assumptions about the representations of abstract data types will be *severely* penalized. **Answer the questions in the spaces provided on the exam paper and turn in the exam paper.** No calculators, laptops, cell phones, handheld computers or Blackberries are allowed.

This exam has 9 pages (not including the cover page).

Question 1

You are given an array A of integers of length N. You are given an array P of integers of length n. Assume that $n < N$. We want to see if there is any subsequence in the array A that looks like P. Write an algorithm to do this; it should return the starting position of the subsequence in P or say that P does not appear anywhere. For example if

$$A \text{ is } [1423537413639025] \text{ and } P \text{ is } [741]$$

your algorithm should return 6. Remember that the numbering of the arrays goes from 0 to $N-1$ or $n-1$ respectively. If P had been [343] your algorithm should say that there is no occurrence of P in A.

Question 2

Is 2^{3^n} $O(3^{2^n})$ or is 3^{2^n} $O(2^{3^n})$? Prove your answer. You can use any results from class without rederiving them.

Question 3

You are given a sequence of integers s, an integer x_0 and a two-argument numerical function f (for example $+$ and $*$ are such functions). Write an algorithm to compute the effect of applying f repeatedly as follows. First we apply f to the first two elements of the sequence obtaining a result, call this result x_1. Then we apply f to x_1 and the third element of the sequence to obtain a new result, call this result x_2. We proceed in this way until we are done with all the elements in the sequence to obtain the result r and then we compute $f(r, x_0)$. If the sequence has only one element, say y, we compute $f(y, x_0)$ as our answer and if the sequence is empty we return x_0 as the answer. This is called *reduction* of a sequence by a binary function. For example

```
reduce((1 2 5 3 4),0,+) = 15
reduce((3 6 2 2),1,*) = 72
reduce((2 3 2),1,exp) = 512
```

Think of reduce as follows: if the sequence is written as $(x_1 x_2 \ldots x_n)$ we are computing $f(x_1, f(x_2, \ldots (f(x_n, x_0)) \ldots))$. For example

```
reduce((1 2 5 3 4),0,+) = 1 + (2 + (5 + (3 + (4 + 0))))
```

Write a simple algorithm for reduce **using sequence operations**.

Question 4

Suppose that you are interested in representing big integers, much bigger than the maximum integer provided by languages like C or $C++$ or Java. For example I may want to represent

$$1267650600228229401496703205376$$

which is 2^{100} or even much larger numbers. Describe a data structure that allows me to represent numbers as large as the *total* memory of the computer allows. By this I mean that the representation should use space proportional to the number of decimal digits in my number. Give the data structure informally with a picture if necessary. You do **not** have to give me Java class definitions.

Question 5

We can sort a sequence of numbers by building a binary search tree and inserting all the numbers. What type of traversal of the resulting tree will produce the sorted sequence? What is the worst-case running time of this algorithm? When does the worst case happen? The last question is the most important.

Question 6

You have a set of expression trees defined with the following types of nodes. The leaves are characters like a, b, c and the internal (binary) nodes are labelled with a $+$ or a $*$. Two trees are *equivalent* if

- they have just one leaf and the label is the same
- one tree is obtained from the other by flipping some of the subtrees.

The idea is that if we were to evaluate two equivalent expression trees we should get the same answer because of the commutativity of $+$ and $*$. Write an algorithm to test whether two trees are equivalent. Assume that you have the following primitive operations on trees: Left and Right, root, and isempty. You can do equality tests like

```
if root(t) = root(t') then ...
```

Question 7

Suppose that we are given a line segment in the plane. We are given the end points of the line segment. We are also given a rectangle in the plane with the sides of the rectangle parallel to the axes. The rectangle is specified by giving the coordinates of the corners. I want to know if

- the line segment is completely inside the rectangle, this includes the case where the end points of the line segment are on the boundary of the rectangle
- the line segment is completely outside the rectangle, this includes the case where one of the corners of the rectangle lies on the line segment
- the line segment crosses the rectangle and is partly inside and partly outside.

This cannot be done in a numerically unstable manner using linear equations, it has to be done without any divisions ever being performed. Write an algorithm for solving this problem.

Question 8

Given a binary search tree and two nodes in the tree we want to find the *nearest common ancestor*. This is the unique node that is an ancestor of both nodes and closest to both nodes. Give an $O(h)$ [h is the height of the tree] algorithm for this problem.

Question 9

The final question consists of 4 multiple-choice questions. Circle the right answer. You do not have to write any reasons. Each correct answer gets you 5 points, each incorrect answer gets you -1 point, an unanswered question gets you 0.

1. (a) $n + \log_2 n$ is $O(\log_2 n)$.
   (b) $n + \log_2 n$ is $\Theta(n)$.
   (c) $n + \log_2 n$ is $O(n \log_2 n)$.
   (d) $n + \log_2 n$ is $\Theta(n \log_2 n)$.

2. It is impossible to have an algorithm to solve the halting problem because
   (a) the problem is not well formulated;
   (b) the present technology is still inadequate, but with future technology it is expected to be solved;
   (c) the world wide web never halts;
   (d) it leads to a logical contrdiction.

3. At an intersection of two roads with a 4-way stop sign, the convention is that the person on the right has the right of way if two cars arrive at the same time.
   (a) This scheme is fair and safe.
   (b) This scheme can deadlock if 4 cars arrive simultaneously.
   (c) This scheme is fair but unsafe because two cars can go simultaneously.
   (d) This scheme is unfair, but it can be fixed by allowing the person with the more expensive car to proceed first.

4. The growth rate of the Catalan numbers is
   (a) $O(n)$
   (b) $O(\log n)$
   (c) $O(4^n)$
   (d) $\Theta(n!)$

   The formula for the Catalan numbers is $Cat(n) = \frac{(2n)!}{(n+1)!n!}$.

Have a great summer!