STUDENT NAME: _____

STUDENT ID: _____

McGill University
Faculty of Science
School of Computer Science

FINAL EXAMINATION

COMP-250: Introduction to Computer Science - Fall 2010

December 20, 2010
2:00-5:00

Examiner: Prof. Doina Precup

Associate Examiner: Prof. Michael Langer

Write your name at the top of this page. Answer directly on exam paper. Three blank pages are added at the end in case you need extra space. There are 11 questions worth 350 points in total. The value of each question is found in parentheses next to it. Please write your answer on the provided exam. Partial credit will be given for incomplete or partially correct answers.

SUGGESTIONS: READ ALL THE QUESTIONS BEFORE YOU START! THE NUMBER OF POINTS IS NOT ALWAYS PROPORTIONAL TO THE DIFFICULTY OF THE QUESTIONS. SPEND YOUR TIME WISELY!

GOOD LUCK!

1. [50 points] **Short questions**

   (a) True or false: is $f(n) = 50n + \log_2 n$ in $O(n)$? Justify your answer.

   (b) Does an algorithm with $O(n \log n)$ always run faster that an algorithms with $O(n^2)$? Justify your answer.

   (c) Give an example of a graph problem in which depth-first search would be preferable to breadth-first search.

   (d) Your friend claims that he know an interesting card trick. If you are thinking of any card number (A, 2, 3, ... 10, J, Q, K), he will *always* guess what card you are thinking about by asking you just 3 questions. Do you believe him? Justify your answer.

   (e) True or false: In Java, if a variable is declared static, its value can only be modified by methods of the class in which it is declared.

2. [20 points]**Big-Oh**

   For the following algorithms, state what $O()$ is and explain your answer:

   (a) **Algorithm** f1(n)
   $i \leftarrow 1$
   while $i < n$
   for $j = n$ **to** i **do**
   print(j)
   $i \leftarrow i + 2$

   (b) **Algorithm** f2(n)
   $i \leftarrow 1$
   while $i < n$
   print(i)
   $i \leftarrow i * 2$
   for $j = 1$ **to** i **do**
   print(j)

3. [40 points] **More Big-Oh**

   Consider the following pseudocode algorithm:

   Algorithm MyAlg(n, k)

   if ($n == 1$ **or** $n == 0$) **return** k
   if (n **mod** $2 == 0$) **return** MyAlg($n/2$, $k + 1$)
   if (n **mod** $2 == 1$) **return** MyAlg($n + 1$, $k + 1$)

   (a) Suppose we call this with initial value $k = 1$. What will the algorithm return?

   (b) Write a recurrence for the running time of the algorithm. Note that you will need two equations: one for even values of n (i.e. $n = 2i$) and one for odd values of n (i.e. $n = 2i - 1$)

   (c) Based on the recurrence relation, what is the best case of the algorithm? Estimate the running time ($O()$) in this case, and justify your answer

   (d) Suppose now that you get an input n which is NOT the best case. Expand the recurrence, assuming worst-case scenarios as much as possible. Based on this what is $O()$ of the algorithm? Give an answer that is as "tight" as possible.

4. [10 points] **Inheritance**

   Consider the following piece of Java code:

```
public class A {
        private char x;
        public A() { x='a'; }
        public void print() {System.out.println(x);}
}

public class B extends A {
        private char y;
        public B() { super(); y='b'; }
        public void print() {super.print(); System.out.println(y);}
}
```

   What is the outcome of the following code:

```
B z = new B();
z.print();
```

5. [50 points] **A queuing algorithm**

   You are a programmer at a company that offers web services. You are tasked to write a Java class, Server, which is supposed to maintain a set of queues, and put arriving requests into the queues. You have been provided with a Request class, which provides the following method:

   void process() //processes a request object

   You are also provided with a Queue class, which contains the following methods:

   Queue() //constructor that creates an empty queue
   void enqueue(Request r) //Put a request into the queue
   Request dequeue() // take the object from the front of the queue out
   int size() //returns the number of objects in the queue

   None of these methods throw any exceptions. If you call dequeue from an empty queue, the program will crash. The queue resize automatically, so it is never "full".

   Write on the next page **Java code** for the Server class, which contains the following methods:
   Server(int n) - creates n empty queues
   void place(Request r) - places the request in the next queue
   //For example, if you had 3 queues, the first request would go in the first queue, the second request in the second queue, the third request in the third queue, the fourth request in the first queue, the fifth request in the second queue etc.
   void process() - dequeues the first request from the *longest* queue and processes it

   Please make sure that you declare all needed variables and you write the code for all the methods.

Code for the Server class goes here

6. [20 points] **Binary trees**

   Prove by induction that a binary tree with n nodes has $n - 1$ edges. If you want a bigger challenge, prove the same problem but for a general tree.

7. [50 points] **Binary search trees**

   Recall that in a binary search tree, at every node, all elements to the left of the node have a smaller key, and all elements to the right of a node have a larger key.

   (a) [30 points] Write in pseudocode an algorithm which, given a binary search tree, will print all the elements bigger than a specified value, min. Make your algorithm as efficient as possible.

   Algorithm PrintElements(BinarySearchTreeNode node, int min)
   Input: A binary search tree node, representing the root of a BST
   Output: Does not return anything, but prints all node values that are bigger than min

   (b) [10 points] What is the best possible running time for your algorithm, as a function of the number of nodes in the tree?

   (c) [10 points] What is the worst possible running time for your algorithm, as a function of the number of nodes in the tree?

8. [30 points] **Secret Santa**

   Suppose that you and your friends decided to exchange secret gifts this year. The usual procedure for this is to put the names of all the people in the group into a bag, then draw names at random (if you draw your own name, you put the note back). We will call "Santa assignment problem" the problem of finding an assignment of Santas so that everyone receives exactly one gift. We will use graphs to model this problem. Suppose that n friends are in the group. We will build a graph with n vertices, and put a directed edge (i, j) if person i is willing to give a gift to person j.

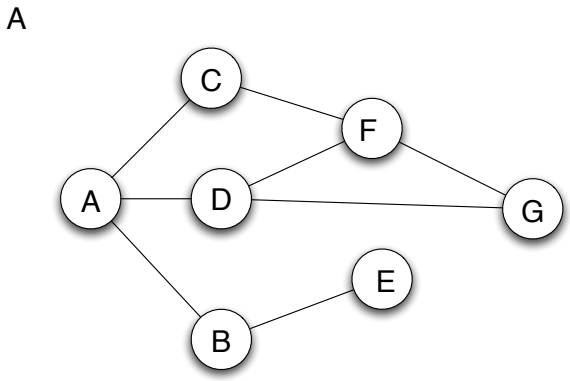
   (a) Formulate "Santa assignment" as a problem in this graph

   (b) Suppose that any person is willing to give a gift to any other person. Does this problem always have a solution? If yes, give a solution example. If no, explain why not.

   (c) Suppose that some of the people are not willing to give a gift to everyone else, but only to a strict subset of the people in the group. In this case, is there always a Santa assignment? Justify your answer.

9. [20 points] **Search in graphs**

Consider the graph in the figure below.



(a) Suppose that you are using breadth-first search to find a path from vertex A to vertex G. Show the state of the queue after each node expansion, assuming nodes are enqueued in alphabetical order. What path do you find?

(b) Suppose that you are using depth-first search for the same task. Show the node expansion, assuming that successors of the same node are investigated in alphabetical order. What path do you find?

10. [20 points] **Binary Search Trees**

    Consider the binary search tree in the figure below.

    (a) Draw the tree resulting from the insertion of value 5

    (b) Draw the tree resulting from the removal of value 2 from the initial tree (not the tree after 5 was inserted)

11. [40 points] **Using data structures**

Suppose that you are hired by a big music streaming company to build a recommendation engine. The company has about 10 million songs. Each song has a title, artist, year, album and genre (there are 50 genres in total). There are roughly 2 million users, and each user has a unique identifier, gender, age group, zip code (to encode geographic location) and country. The company wants you to build "user profiles" which can be used to recommend songs to users. The idea is that if we want to recommend a song to user u, we have to find similar users, and choose songs to which the similar users listened.

(a) Your boss suggests that you should maintain the user profile at the level of the songs: for each user, for each song in the data base, record how many times the user has listened to this particular song. Propose a data structure that you could use to maintain this information.

(b) You secretly think that it would be better to maintain user profiles by genre rather than song (for each user, how many times they played songs from this particular genre. How would you record this information, and what data structure would you use? How could you use such a profile to find songs that you can recommend?

(c) A different group in the company is developing a method that compares users. For users u and v, u.distance(v) will return a number which tells how similar u and v are. Suppose that this method only uses the user's personal information. Describe a way of efficiently organizing the data, so that for a new user, you can quickly find similar users.

(d) You think that a better way of comparing users is by looking at their profile information. In this case, the profiles change all the time. Can you still organize the data efficiently so that for a user u, you can quickly find similar users? Justify your answer

Page left intentionally blank in case you need extra space

Page left intentionally blank in case you need extra space

Page left intentionally blank in case you need extra space