STUDENT NAME: _____

STUDENT ID: _____

McGill University
Faculty of Science
School of Computer Science

FINAL EXAMINATION

COMP-250: Introduction to Computer Science - Winter 2006

April 25, 2006
2:00-5:00

Examiner: Prof. Doina Precup

Associate Examiner: Prof. Mathieu Blanchette

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 10 questions worth 100 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!

CLOSED BOOK.
CALCULATORS + DICTIONARIES NOT ALLOWED.
TRANSLATION DICTIONARIES ARE ALLOWED.

1. [30 points, 3 points each] **Short questions**

   Indicate whether the following statements are true or false. Give a two-line justification for each. Credit will be given only if the justification is correct.

   (a) $2^{\log_{10} n}$ is $O(n)$

   (b) All sorting algorithms have a worst-case running time of $O(n \log n)$.

   (c) Suppose that some algorithm A has running time $O(n \log n)$ and algorithm B has running time $O(n^2)$. Then A is always preferred to B.

   (d) Problems in the class P are also part of the class NP.

   (e) Suppose I have a hash table with 50 buckets. Then I cannot store more than 50 items in this hash table.

(f) $n \cos n$ is not $O(n)$.

(g) Quicksort's worst case input is when the array is already sorted.

(h) An algorithm that does not contain recursive calls always terminates.

(i) Removing an desired element from a linked list takes $O(n)$ running time, where n is the number of elements inn the list.

(j) Queues are used to in web servers instead of stacks because they ensure that all customers are treated equally.

2. [10 points] **A simple algorithm**

   You are given an array of n positive integers. The numbers do not appear in any particular order. Write an algorithm that will move all the numbers that are multiples of 13 (if any) to the front of the array. You algorithm should have running time $O(n)$.

   For example, your initial array might be: 2 13 4 26 5 1 2 After the rearrangement, the array should have 13 and 26 in the first two positions. The order does not matter.

3. [10 points] **Merging in triplets**

Recall that the merge sort algorithm proceeds by recursively splitting an array of n numbers, sorting each part recursively, and then merging the resulting array. The pseudocode of the algorithm is as follows:

Algorithm mergeSort(a,p,q)
Input: An array a and two indices p and q between which we want to do the sorting.
Output: The array a will be sorted between p and q as a side effect
if $p < q$ **then**
  int $m \leftarrow \frac{\lfloor p+q \rfloor}{2}$ //this is the middle of the part of interest
  mergeSort(a,p,m)
  mergeSort(a,$m+1$,q)
  merge(a,p,m,q)

Algorithm merge(a,p,m,q)
Input: An array a, in which we assume that the halves from $p \ldots m$ and $m+1 \ldots q$ are each sorted
Output: The array should be sorted between p and q
Array tmp // this is an array of size $q-p+1$ (same as a) which will hold the temporary result
int $i \leftarrow p$ //these are the two indices in the two halves of the array
int $j \leftarrow m+1$
int $k \leftarrow 1$ //this is the index we will use in the tmp array
while ($i \leq m$ **or** $j \leq q$) **do**
  if ($j = q+1$ **or** $a[i] \leq a[j]$ **then**
    $tmp[k] \leftarrow a[i]$
    $i \leftarrow i+1$
  else
    $tmp[k] \leftarrow a[j]$
    $j \leftarrow j+1$
  $k \leftarrow k+1$
for $k = p$ **to** q **do**
  $a[k] \leftarrow tmp[k]$

Now suppose that you want to cut the array in 3 parts instead of 2. Write the pseudocode for the mergeSort and merge functions in this case. Does this make a difference in terms of the big-oh of the algorithm? Justify your answer.

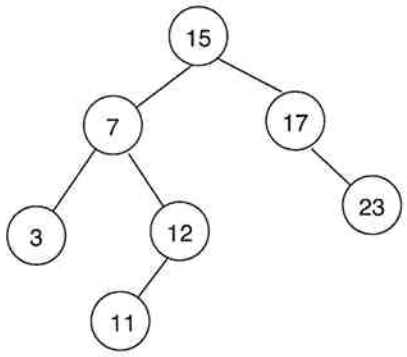
4. [10 points] **Tree node counting**

   Suppose you have a binary tree. Write an algorithm in pseudocode that counts the number of nodes with exactly one child. State the big-oh of your algorithm as a function of the number of nodes in the tree, N.

   Algorithm CountOneChild (TreeNode n)
   Input: A TreeNode n
   Output: The number of nodes, in the subtree rooted at n, that have exactly one child

5. [5 points] **Binary search trees**

   Consider the following binary search tree:

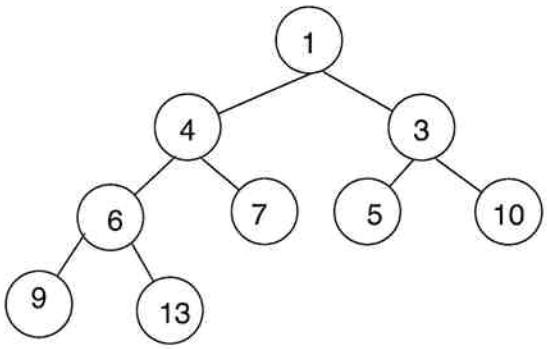


   (a) [2 points] Draw the binary search tree after an element with key 14 has been inserted

   (b) [3 points] Draw the binary search tree after remove(7) has been executed. Start from the original tree, not the tree you got in (a)

6. [5 points] **Heaps**

Consider the following heap:

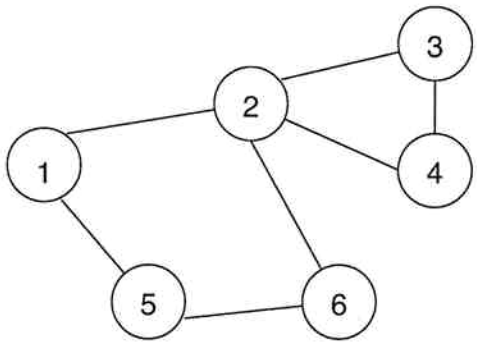


(a) [2 points] Draw the heap after the element with key 2 has been inserted.

(b) [3 points] Draw the heap after removeMin() has been executed. Start from the original heap, not the one you got in (a)

7. [10 points] **Search in graphs**

   Consider the graph in the figure below.



   (a) [4 points] Suppose that you are using breadth-first search to find a path from vertex 1 to vertex 4. Show the state of the queue after each node expansion, assuming nodes are enqueued in increasing order. What path do you find?

   (b) [4 points] Suppose that you are using depth-first search for the same task. Show the state of the stack after each node expansion, assuming that successors of the same node are pushed in increasing order. What path do you find?

   (c) [2 points] Based on this example, what is the main advantage of depth-first search? What is the main advantage of breadth-first search?

8. [10 points] **Which data structure to pick?**

   You have been hired by a large computer company as a summer intern. They have a huge database of customers, containing the customer's user name, their age group (under 20, 20-29, 30-39, 40-49, 50-59, more than 60), and the songs they have downloaded so far, along with the song's genre (one of 20 categories like classic rock, jazz etc). The company believes that customers in the same age group have similar preferences. They want to set up a new feature which will propose to new customers songs that they are likely to buy. The feature should display 5 songs at a time, and it should do this without perceivable delays. It is desirable to propose different songs every time (to the extent possible).

   You are given initially a large list containing all transactions, in the format:
   user-name, age-group, song, genre

   You know that using this list directly will be very slow, so you consider building a different data structure(s) based on this data. You are allowed to summarize the data you have in any way you want. Describe, in at most 7 sentences, what data structure(s) you would build, why you are making this choice, and how you would design the desired feature. No pseudocode is necessary.

9. [5 points] **Proof by induction**

   Recall that a binary tree is called *proper* if every node has either 2 or 0 descendents. Prove by induction (on the height of the tree) that in a proper binary tree, the number of leaves is always equal to the number of internal nodes + 1.

10. [5 points] **A linear-time sorting algorithm**

    Suppose you are given an array of length n, containing integer elements. You know that each element is equal to 1, 2, 3, 4 or 5: $a[i] \in \{1, 2, 3, 4, 5\}, \forall i = 1 \ldots n$. Give a linear-time ($O(n)$) algorithm for sorting the array.

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