Boğaziçi University, Dept. of Computer Engineering

CMPE 250, DATA STRUCTURES AND ALGORITHMS

Fall 2010, Midterm 2

Name: _____

Student ID: _____

Signature: _____

- Please print your name and student ID number and write your signature to indicate that you accept the University honour code.
- During this examination, you may not use any notes or books.
- Read each question carefully and **WRITE CLEARLY**. Unreadable answers will not get any credit.
- There are **5** questions. Point values are given in parentheses.
- You have **120 minutes** to do all the problems.

Q	1	2	3	4	5	Total
Score						
Max	20	20	20	20	20	100

1. What is .. (Give short answers. Long answers do not get any credit.)

   (a) a data structure ?
   A mathematical object that represents the organization of data, stored on a storage medium such as RAM or disk.

   (b) an algorithm ?
   A method for solving a computational problem expressed as a **finite sequence** of steps. Each step can be specified by a **list of well-defined instructions**. The instructions describe a computation that, for a given initial state and admissible input, proceeds through a well-defined series of successive states, eventually terminating in a final ending state.

   (c) a hash function ? A many to one function that maps a key to a storage index.

   (d) double hashing ? The use of two hash functions to resolve collisions

   (e) rehashing ? Resizing a hash table and remapping the elements

   (f) a binary heap ? A complete binary tree with the convention that all parent and child nodes satisfy a certain relation (often an ordering such as $>$)

   (g) d-ary heap ? A complete tree, where each node can have up to d children with the convention that all parent and child nodes satisfy a certain relation (often an ordering such as $>$)

   (h) an example application where a heap is useful ? Task manager in an operating system where tasks are ordered according to their priority

   (i) a graph ? A tuple $G = (V, E)$ where V is a countable set and $E \subset V \times V$

   (j) a sparse graph ? A graph $G = (V, E)$ where $|E| \ll |V|^2$

   (k) a hypergraph? A tuple $G = (V, E)$ where V is a countable set and $E \subset V \times V \times V \times \ldots$

(l) a good data structure for the representation of a graph when the number of edges $|E| = O(|V| \log |V|)$, where $|V|$ is the number of vertices and space is an issue? Adjacency list

(m) a spanning tree?

(n) a minimum spanning tree?

(o) a Greedy algorithm?

(p) Prim's algorithm?

(q) a Breadth first search ?

(r) a Depth first search ?

(s) what is a template in C++?

(t) what is a copy constructor?

(20 points)

2. Given the input $\{4371, 1323, 6173, 4199, 4344, 9679, 1989\}$ and a hash function $h(x) = x \mod 10$, show the resulting?

   (a) Separate Chaining hash table

   (b) Hash table with linear probing

   (c) Hash table with quadratic probing

   (d) Hash table with second hash function $h_2(x) = 7 - (x \mod 7)$

(20 points)

3. Give an algorithm to find and print all nodes less than some given value X in a min-heap. First, explain your idea in a few sentences.

   - Your algorithm must be $O(K)$ where K is the number of elements less than X.
   - You should not modify the heap

(20 points)

4. Write down an algorithm to topologically sort a graph represented by an adjacency list, modified such that the algorithm prints out a cycle, if it is found. First, explain your idea in a few sentences. (Don't use depth first search, we want just a modification of the basic topological sort.) *(20 points)* **Solution:**
   If no vertex has indegree 0, we can find a cycle by tracing backwards through vertices with positive indegree; since every vertex on the trace back has a positive indegree, we eventually reach a vertex twice, and the cycle has been found.

5. A undirected graph is *k-colorable* if each vertex can be given one of the k colors, and no edge connects identically colored vertices. Give an efficient algorithm (linear in the number of edges) to test a graph for 2-colorability. The graph is represented as an adjacency list. First, explain your idea in a few sentences. Specify any additional data structures needed. *(20 points)*
   Solution:
   1) Use a depth-first search, marking colors when a new vertex is visited, starting at the root, and returning false if a color clash is detected along a backedge.

   2) Use Breadth first search, inserting unvisited nodes with the opposite color of their parent. If a child node is already visited, check its color and return false if parent and child have the same color. Return true if no such node is found.