UNIVERSITY OF TORONTO
Faculty of Arts and Science
DECEMBER Fall 2015 Final Examination
CSC 263H1F
Duration - 3 hours
No Aids Allowed

PLEASE COMPLETE THE SECTION BELOW.

First (Given) Name:

Last (Family) Name:

Student Number:

Exam Instructions and Course Policy

- Check that your exam has 21 pages (including this cover page and 5 blank pages at the end of this book). Bring any discrepancy to the attention of an invigilator.
- There are ten questions worth a total of 112 points.
- For rough work, use the backs of the pages or the last 5 pages; these will not be marked.
- The exam will be graded for correctness, completeness, clarity and conciseness.
- Course policy reminder: a mark of at least 40% on the final exam is necessary to pass the course.
- Unless stated otherwise, you can use standard data structures or algorithms discussed in class or in tutorial and you do not need to describe their implementation. Simply state which algorithm or data structure you are using. You may also state its complexity without proof. If you modify a data structure or an algorithm from class, you must describe the modification and its effects.
- DFS stands for depth-first search, BFS stands for depth-first search, MST stands for minimum spanning tree.

Good Luck!

This page is for marker's use only. Do NOT use it for answering or for scratch paper.

Question 1	/14
Question 2	/6
Question 3	/10
Question 4	/10
Question 5	/12
Question 6	/10
Question 7	/16
Question 8	/12
Question 9	/12
Question 10	/10
Total	/112

1. (14 points, 2 points each) Circle the correct answer; no justification is necessary.

   (a) True or False: Let X and Y be two random variables. Then $E[X+Y] = E[X] + E[Y]$.

   (b) True of False: The worst case complexity is $O(\log n)$ for performing a search in an AVL tree.

   (c) True or False: The worst case complexity is $O(\log n)$ for search in a binary search tree.

   (d) True or False: The operations of delete and insert applied to a binary search tree are commutative. That is, given a binary search tree T, deleting x and then delete y leaves the same tree as deleting y and then x.

   (e) Suppose we have n keys and they are searched for with equal probability. They are stored in a binary search tree that minimizes the expected number of nodes examined to find a key. The *worst-case* number of nodes examined to find a key in that tree is:

      i. $\Theta(1)$
      ii. $\Theta(n)$
      iii. $\Theta(\log n)$
      iv. None of the above.

   (f) Suppose we insert n keys into a binary search tree. The first inserted key is 0, the second one is 1, and from then on, each inserted key is the average of the previous two. Thus the sequence of inserted keys begins as follows: $0, 1, \frac{1}{2}, \frac{3}{4}, \frac{5}{8}, \ldots$. The height of the resulting tree is:

      i. $\Theta(1)$
      ii. $\Theta(\log n)$
      iii. $\Theta(n)$
      iv. $\Theta(n \log n)$
      v. None of the above.

   (g) Let G be a connected, undirected graph with n nodes and m edges. Suppose its edges have distinct weights $1, 2, \ldots, m$. Then the cost of a minimum cost spanning tree of G must be exactly:

      i. $n - 1$
      ii. n
      iii. $n(n-1)/2$
      iv. $n(n+1)/2$
      v. None of the above.

2. (6 points, 2 points each) Consider the Randomized Quicksort algorithm where we pick the pivot element at random at each iteration. Answer each question below. No justification is necessary.

   (a) What is the worst-case complexity of Randomized Quicksort?

   (b) What is the best-case complexity of Randomized Quicksort?

   (c) What is the average-case complexity of Randomized Quicksort?

3. (10 points) Consider an algorithm that determines whether two n-bit numbers $x = x_{n-1}, \ldots, x_0$ and $y = y_{n-1}, \ldots, y_0$ are equal, by comparing them one bit at a time, from the most significant to the least significant bit, until the bits differ or until all bits have been compared. Suppose that x and y are each equally likely to be any n-bit binary number, and they are chosen independently.

   (a) Define your probability space.

   (b) Define all necessary random variables.

   (c) What is the expected number of bit comparisons for $n = 3$? Explain your answer.

   (d) What is the expected number of bit comparisons for arbitrary n? (Simplification is not necessary.) Explain your answer.

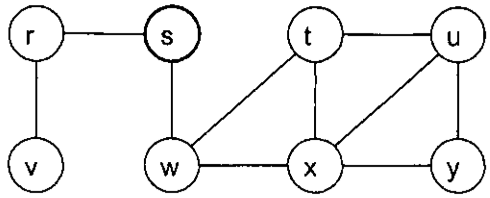
4. (10 marks) Suppose we implement min-heap as an array. To perform an INSERT operation, if the array is not full we proceed as explained in class. If the array is full, we first move the elements of the heap from the current (full) array into a new array of twice the size and then we perform the insertion in the new (half-full) array. To perform an EXTRACTMIN operation, if the array is more than a quarter full we proceed as explained in class. If the array is only a quarter full, we first move the elements of the heap from the current (quarter-full) array into a new array of half the size, and then we perform the EXTRACTMIN in the new (half-full) array.

   Suppose that the heap is initially empty and we perform an arbitrary sequence of n INSERT and EXTRACTMIN operations.

   (a) What is the *worst-case* cost for an individual operation in such a sequence? Explain your answer.

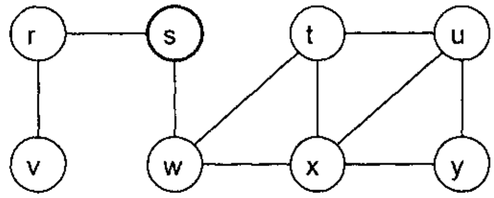
(b) Use the accounting method to determine the *amortized* cost per operation in a sequence of n operations. State the cost of each operation, and the credit invariant, and explain why the credit invariant can be maintained.

5. BFS/DFS problem (12 points)



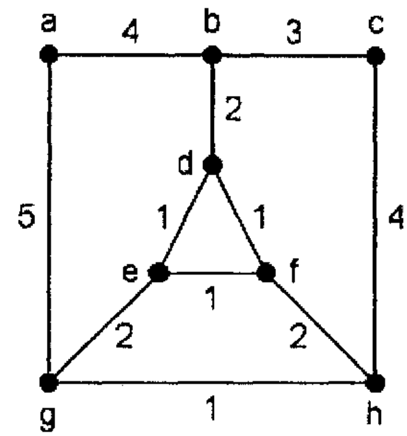
Assume that the graph above is given as an array of adjacency lists where the vertices are listed in alphabetical order, and for each vertex, the vertices in its adjacency list are listed in alphabetical order. For example, the adjacency list for vertex t contains the vertices u, w, x, in this order.

(a) Execute the BFS algorithm on the following undirected graph starting from vertex s. Draw the resulting BFS tree. What is the distance label $d[u]$ of vertex u in this BFS?



(b) Now execute the DFS algorithm on the same graph starting from vertex s. Draw the resulting DFS forest. What are the discovery and finish times of vertex u in this DFS? Draw the back, forward and cross edges (if any) with dotted lines and label them as back, forward, cross accordingly.

6. MST problem (10 points)



Execute Kruskal's MST algorithm on this graph and list the MST edges in the order they are included in the MST by the algorithm. Let the sorted order of the edges be:

$$(d, e), (g, h), (e, f), (d, f), (b, d), (e, g), (f, h), (b, c), (a, b), (c, h), (a, g).$$

7. (16 points) Consider the following ADT that consists of a subset $S \subseteq \{1, \ldots, n\}$ and supports the following operations:

- DELETE(S, i): Delete integer $i \in \{1, \ldots, n\}$ from the subset S. If $i \notin S$, this operation has no effect.
- PRED(S, i): Return the predecessor in S of $i \in \{1, \ldots, n\}$, i.e. $\max\{j \in S \mid j < i\}$. If i has no predecessor in S (i.e. if $i \leq \min(S)$), then return 0.

Initially, $S = \{1, \ldots, n\}$.

In this question, you will explain how to use a disjoint set data structure to implement this ADT so that the amortized cost of each operation is in $O(\log^* n)$.

(a) (1 point) Let $n = 12$ and $S = \{3, 5, 6, 8, 12\} \subseteq \{1, \ldots, 12\}$.

What is the set $\{i \in \{1, \ldots, 12\} \mid \text{PRED}(S, i) = 0\}$?

What is the set $\{i \in \{1, \ldots, 12\} \mid \text{PRED}(S, i) = 5\}$?

What is the set $\{i \in \{1, \ldots, 12\} \mid \text{PRED}(S, i) = 8\}$?

What is the set $\{i \in \{1, \ldots, 12\} \mid \text{PRED}(S, i) = 12\}$?

(b) (4 points) Explain how the disjoint set abstract data type can be used to represent a subset $S \subseteq \{1, \ldots, n\}$. Include a diagram of your data structure for the subset $S = \{3, 5, 6, 8, 12\} \subseteq \{1, \ldots, 12\}$. (Hint: Look at part (a).)

(c) (1 point) Which disjoint data set structure should you use?

(d) (3 points) Explain how to implement DELETE(S, i).

(e) (3 points) Explain how to implement PRED(S, i).

(f) (2 points) Briefly explain why the amortized cost of each operation is in $O(\log^* n)$.

(g) (2 points) Explain how to initialize your data structure. How much time does it take?

8. (12 marks) Recall that a *simple cycle* in an undirected graph $G = (V, E)$ is a sequence of distinct vertices $v_0, v_1, \ldots, v_{k-1}$, where $k \geq 3$ and $\{v_0, v_1\} \in E$, $\{v_1, v_2\} \in E$, $\ldots$, $\{v_{k-1}, v_0\} \in E$. An undirected graph G is *acyclic* (i.e., it contains no simple cycle) if and only if G is a *forest* (i.e., G consists of one or more tree(s)).

   Explain how to use a graph algorithm discussed in class to determine, in $O(|V|)$ worst case time, whether a given undirected graph $G = (V, E)$ contains a cycle. State which algorithm from class you will use, and then how to use it to solve the cycle problem. Briefly explain why your solution runs in $O(|V|)$ time.

9. (12 points, 3 each) Explain how to augment an AVL tree representing a set of at least two integers so that, given a pointer p to any node in the tree, it is possible to determine in $O(1)$ time, the integer c in the set that is closest, but not equal, to the integer stored in the node to which p points.

   (a) What additional field(s) do you add to each node?

   (b) Explain how to modify INSERT and explain why it still takes $O(\log n)$ time, where n is the number of integers in the tree when the operation is performed.

(c) Explain how to modify DELETE and explain why it still takes $O(\log n)$ time, where n is the number of integers in the tree when the operation is performed.

(d) Explain how to to determine, in $O(1)$ time, the integer c in the set that is closest, but not equal, to the integer stored in the node to which a given pointer p points.

10. Universal Hashing (10 points) Let $U = \{1, \ldots, m\}$ and let $R = \{0, \ldots, m-1\}$.

(a) (5 points) Give the definition of a universal family of hash functions from U to R.

(b) (5 points) Let $f : U \to R$ denote the function $f(x) = x - 1$. Is $\{f\}$ a universal family of hash functions from U to R? Justify your answer.

Scratch Paper

Scratch Paper

Scratch Paper

Scratch Paper

Scratch Paper