University of Toronto
Faculty of Arts and Science
December 2013 Examinations

CSC263H1F/CSC265H1F

No Aids Allowed

There are eleven questions worth a total of 140 marks.

Check that your exam book has 16 pages (including this cover page).

PLEASE COMPLETE THIS SECTION:

Name _____

(Please underline your family name.)

Student Number _____

FOR USE IN MARKING:

1. _____/ 18
2. _____/9
3. _____/8
4. _____/10
5. _____/12
6. _____/10
7. _____/20
8. _____/8
9. _____/20
10. _____/15
11. _____/10

Total:_____ /140

1. **Hash Tables** (18 marks) In this question, we consider hash tables with chaining, where m is the size of the hash table (i.e., the size of the array) and n is the number of items that will be stored in the hash table.

   (a) (2 marks) What is the issue with the *direct access table* data structure that the *hash table data* structure solves?

   (b) (2 marks) If we want to store n items in a hash table, what is a good choice for m?

   (c) (2 marks) What is the purpose of chaining?

   (d) (2 marks) What property should a good hash function have?

   (d) (6 marks) Assume the SUHA (the Simple Uniform Hashing Assumption) holds. Fill in the following table for the running-time of hash table perations where k is a key and x is a (pointer to) an item inside the hash table. Give asymptotic tight bounds (use Θ). Your answers should *not* contain any parameters other than n and m.

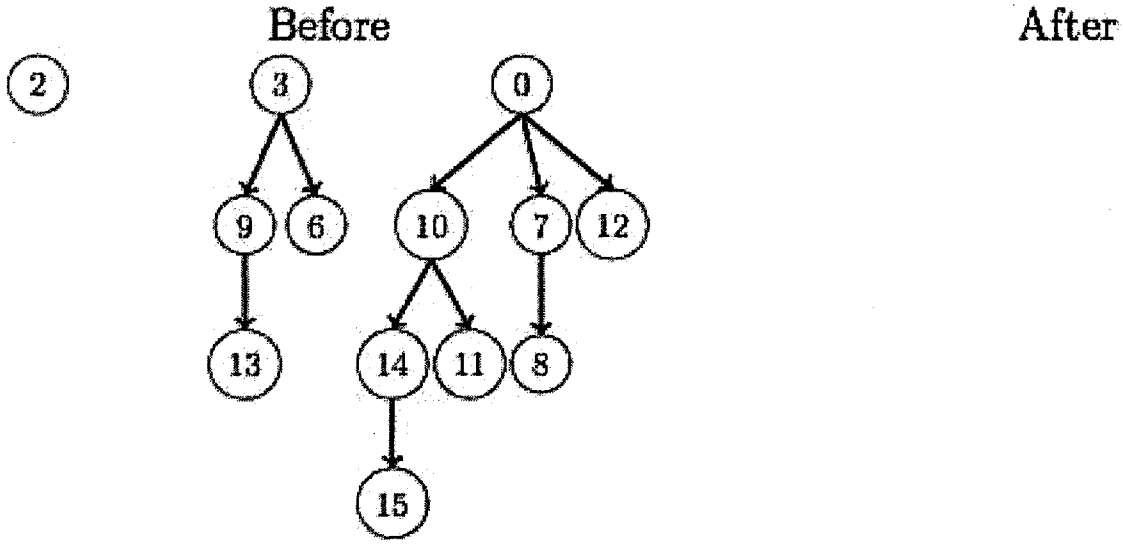
Complexity	Search(k)	Insert(k)	Delete(x)
Worst Case Runtime			
Expected Runtime			

   (e) (4 marks) Consider a hash table T of size 1000 that uses chaining. Suppose that T is initially empty. Assume that SUHA holds. What is the *maximum* number of keys that can be inserted into T such that after these insertions, the *expected number* of key comparisons to search for a key k that is **not** in T is at most 15. Do *not* justify your answer.

2. **Disjoint Sets** (9 marks) Fill in the following table for the *worst-case* running time of the operations of the disjoint sets data structure and the amortized running-time of an operation in a sequence of m operations where n of them are MakeSet operations. Provide good asymptotic *upper bounds* (use O). Your answers should *not* contain any parameters other than n and m. The heuristics are the two that were discussed in class: *path compression* and *union by weight.*

Implementation	FindSet(x)	Union(x,x')	Amortized
Linked List union by weight			
Forest no heuristics			
Forest both heuristics			

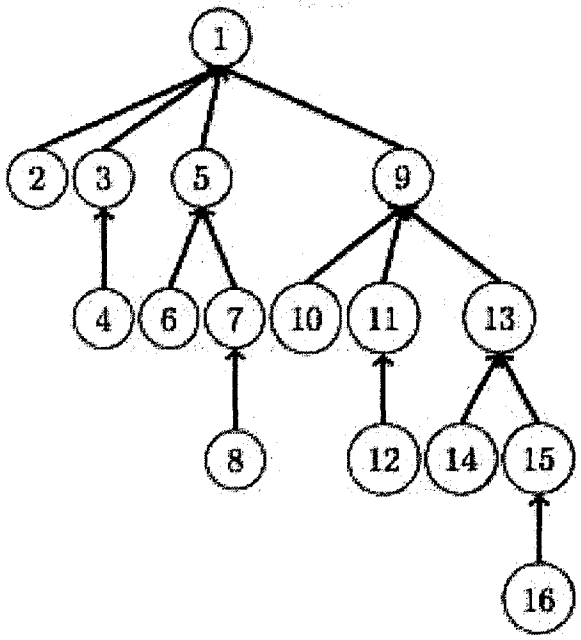
3. **Binomial Heaps** (8 marks) Execute the ExtractMin operation on the following *binomial heap* and show the resulting binomial heap. Do not show the intermediate steps.



4. **Disjoint Sets** (10 marks) Consider the forst implementation of the disjoint set abstract data type with both heuristics (Weighted Union (by size)) and Path Compression. Perform the indicated operation on the given forest and draw the resulting forest. Do *not* show the intermediate steps. Note: each part is independent from the other.

   (a) (5 marks) Perform a Find(15) operation:

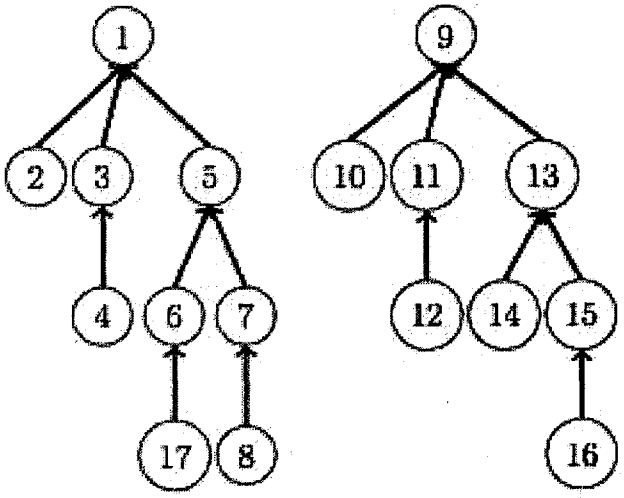
Before



After

(b) (5 marks) Perform a Union(Find(15),Find(6)) operation:

Before



After

5. **Graph Data Structures** (12 marks) In this question G denotes a graph, n is the number of vertices in G, and m is the number of edges in G. Answer the following questions by giving good asymptotic *upper-bounds* (use the O notation). Your answers must be in terms of n and m only. Do *not* justify your answers.

   (a) (2 marks) How much space do we need to store G using an *adjacency list*?

   (b) (2 marks) How much space do we need to store G using an *adjacency matrix*?

   (c) (4 marks) Fill in the following table for the worst-case running time of the BFS and DFS algorithms when the graph is given by: an adjacency list, an adjacency matrix.

Algorithm	Adjacency List	Adjacency Matrix
BFS (Breadth-First Search)		
DFS (Depth-First Search)		

   (d) (4 marks) What is the worst-case running time of Kruskal's algorithm when it is executed on a connected graph? (Recall that Kruskal's algorithm is the greedy algorithm for minimum spanning tree.) Assume that this graph is given by its weighted edge list and the algorithm uses efficient data structures, as we did in class.

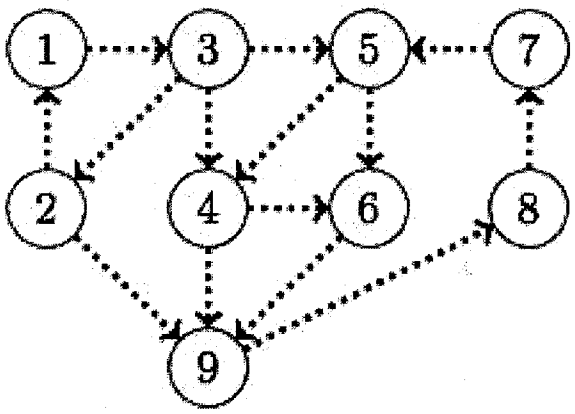
6. **Graphs** (10 marks)

   (a) (5 marks) Give an efficient algorithm that determines whether a connected undirected graph $G = (V, E)$ with weight function w has more than one minimum spanning tree.

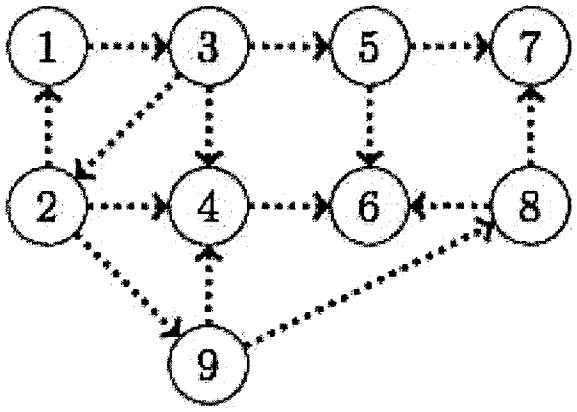
   (b) (5 marks) Prove that, if all edges in E have different weights, then G has only one minimum spanning tree.

7. **Graphs** (20 marks)

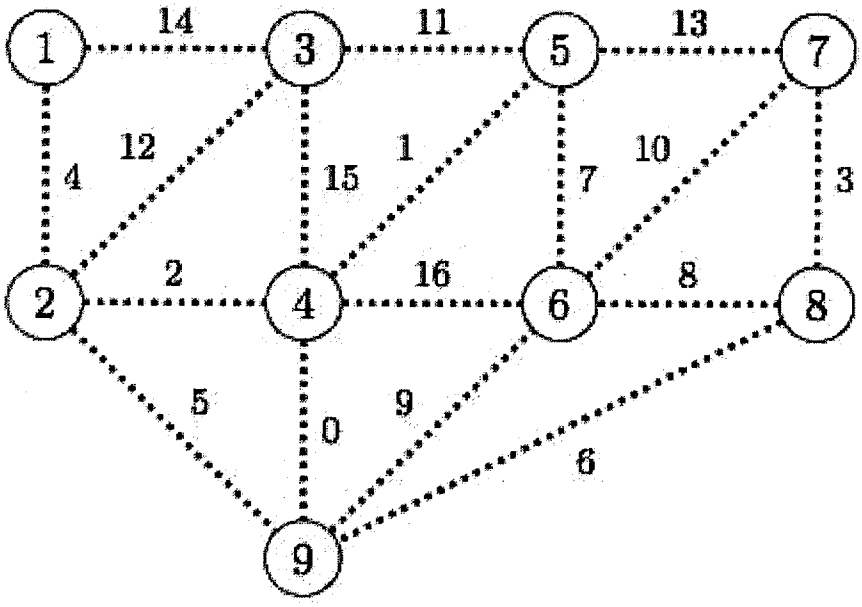
   (a) (5 marks) **Breadth First Search** Execute the BFS algorithm on the following directed graph starting from vertex 1. Assume that the vertices are listed in increasing order in the adjaceny list of the graphs. For example, for vertex 3, the list of vertices are 2, 4, 5 (in this order) in the graph's adjacency list. Draw the resulting BFS tree. What is the value of $d[8]$ given to vertex 8 by this BFS?



(b) (8 marks) **Depth First Search** Execute the CFS algorithm on the following directed graph starting from vertex 1. Assume again that the vertices are listed in increasing order in the adjacency list of the graph. Draw the resulting DFS forest. What are the *discovery* and *finish* times of vertex 8 in this DFS? Draw the *forward*, *back*, and *cross* edges (if any) with dotted lines and label them as b, f, c accordingly.



(c) (7 marks) **Kruskal's Minimum Spanning Tree Algorithm** The following undirected graph has 17 edges, and the edge weights are the consecutive integers $0, 1, 2, 3, \ldots, 16$. 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.*



8. **Probabilistic Analysis** (8 marks) 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) (3 marks) What is the expected number of bit comparisons for $n = 3$? No explanation is required.

   (b) (5 marks) What is the expected number of bit comparisons for arbitrary n? Simplification is not necessary, and no explanation is required.

9. **Lower Bounds** (20 marks) Conside the following problem P: Given a Boolean array $A[1..n]$, return an $i \in \{1, \ldots, n-1\}$ such that $A[i] \neq A[i+1]$, if such an i exists, or return 0 if there is no such i. Let $P_=$ be the restricted version of problem P with the precondition $A[1] = A[n]$ and let $P_\neq$ be the restricted version of problem P with the precondition $A[1] \neq A[n]$.

   (a) (5 marks) Give a description of a comparison tree that solves $P_=$ with worst-case depth $O(n)$.

   (b) (5 marks) Prove that the worst-case depth of *any* comparison tree that solves problem $P_=$ is $\Omega(n)$.

(c) (5 marks) Give a description of a comparison tree that solves $P_{\neq}$ with worst-case depth $O(\log n)$.

(d) (5 marks) Prove that the worst-case depth of *any* comparison tree that solves problem $P_{\neq}$ is $\Omega(\log n)$.

10. **Range Search** (15 marks) We want to perform the following operations on a set of integers, S:

    - $Insert(k)$: inserts an integer k into S,
    - $Delete(x)$: removes the item (pointed to by) x from S,
    - $Search(k)$: returns *true* if and only if S contains k,
    - $IsEmptyRange(S, a, b)$: returns *true* if and only if S contains no integer in the range $[a, b]$, i.e., there is no k in S such that $a \leq k \leq b$ (where a and b are integers such that $a \leq b$).

    The worst-case running time of each operation must be $O(\log n)$ where n is the number of items in S.

    (a) A data structure that we studied in this course supports *Insert*, *Delete*, *Search* and can also be used to implement $IsEmptyRange$ (with the desired $O(\log n)$ running time) *without any modifications to the data structures.* Which one?

    (b) Give the pseudo-code of an algorithm that implements operation $IsEmptyRange(S, a, b)$ with this data structure. Your algorithm must be simple and short (its pseudocode should have at most sevel lines.)

    (c) Briefly explain why the worst-case running time of your algorithm for $IsEmptyRange(S, a, b)$ is $O(\log n)$.

11. **Amortized Analysis** (10 marks) Using the accounting method, prove that the amortized cost of BINOMIAL-HEAP-INSERT is $O(1)$ and amortized cost of BINOMIAL-HEAP-EXTRACT-MIN is $O(\log n)$, where n is the number of elements in the binomial heap. Let \$1 denote the maximum of: (1) the time to create a binomial heap with one element; (2) the time to compare the sizes of two binomial trees, the value of their roots, and link them together; and (3) the time to perform enough pointer operations to move a tree from one linked list to another.