CSC B63 Winter 2023
Final Examination
Duration — 3 hours
Aids allowed: none

*Do **not** turn this page until you have received the signal to start.*

Please fill out the identification section above and read the instructions below.

Good Luck!

This exam is double-sided, and consists of 6 questions. *When you receive the signal to start, please make sure that your copy is complete.*

- Please, make sure to **NOT write anything in the QR code areas**.
- Please, use a **black** or **blue pen** or a **thick in diameter pencil** to answer **all questions** in this booklet.
- If you use any space for rough work, indicate clearly what you want marked.

Question 1. Balanced Trees [12 MARKS]

Below is my attempt at an $\mathcal{O}(\log n)$ (where n is the number of nodes in the tree) algorithm `insert(T, k, v)` that takes (a root node of) an AVL tree T, a key k, and a value v, and returns the result of inserting the key-value pair (k, v) into T. Assume the algorithm `rebalance(T)` is provided in the same way as we did in class/textbook, i.e., it performs the appropriate rotations at T.

```
insert(T,k,v):
  if T == nil:
    return new_node(key=k,value=v,height=0)
  if k < T.key:
    T.left := insert(T.left,k,v)
  elsif k > T.key:
    T.right := insert(T.right,k,v)
  if abs(height(T.left) - height(T.right)) > 1:
    T := rebalance(T)
  return T
```

```
height(T):
  if T == nil:
    return 0
  h_left := height(T.left)
  h_right := height(T.right)
  return 1 + max(h_left, h_right)
```

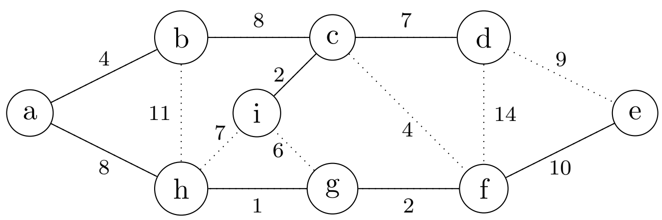
Is the algorithm correct? Does it achieve the $\mathcal{O}(\log n)$ time bound? If your answers are Yes, then explain why. If at least one of the answers is No, explain why not and how to fix the problem(s).

Question 1. (CONTINUED)

Question 2. Graph Algorithms I [15 MARKS]

In this question you are asked to develop (or recall from your A3!) the algorithm `get-paths` that takes a distance tree produced by Dijkstra's shortest paths algorithm and returns a list of the corresponding shortest paths.

Below are an example graph and the output distance tree.



Distance tree: $\{$(a,b,4), (a,h,8), (h,g,9), (g,f,11), (b,c,12), (c,i,14), (c,d,19), (f,e,21)$\}$

Part (a) [5 MARKS]

Describe a data structure that can be used to store the distance tree. Show how the example tree above would be stored.

Part (b) [10 MARKS]

Provide pseudo-code for `get-paths` that takes the data structure described above as input, and returns an array of linked lists of the form:

```
1: (b,a,4)
2: (c,b,8) -> (b,a,4)
3: (d,c,7) -> (c,b,8) -> (b,a,4)
4: (e,f,10) -> (f,g,2) -> (g,h,1) -> (h,a,8)
5: (f,g,2) -> (g,h,1) -> (h,a,8)
6: (g,h,1) -> (h,a,8)
7: (h,a,8)
8: (i,c,2) -> (c,b,8) -> (b,a,4)
```

Question 3. Graph Algorithms II [20 MARKS]

Recall the pseudo-code for Kruskal's algorithm and the invariants it maintains:

```
0. T := new container for edges
1. L := edges sorted in non-decreasing order by weight
2. for each vertex v:
3.   v.cluster := make-cluster(v)
4. for each (u, v) in L:
5.   if u.cluster != v.cluster:
6.     T.add((u,v))
7.     merge u.cluster and v.cluster
8. return T
```

1. each cluster is a tree
2. $T \subseteq T_{min}$ for some MST T_{min}

Your task is to complete the following arguments in the proof of correctness of Kruskal's algorithm.

Part (a) [10 MARKS]

Suppose (1) and (2) are true before line 4 (beginning of iteration i).
To show: (1) and (2) are true after line 7 (end of iteration i).

Part (b) [10 MARKS]

Assuming the invariants hold at the end of each iteration, give an argument that the returned T is a MST.

Question 3. (continued)

Question 4. Disjoint Sets [20 MARKS]

Show the data structure that results from, and the answers returned by the `find-set` operations in, the following program.

```
for i = 1,2,3,...,16:  make-set(x_i)
for i = 1,3,5,...,15:  union(x_i, x_{i+1})
for i = 1,5,9,13:  union(x_i, x_{i+2})
union(x_1, x_5)
union(x_11, x_13)
union(x_1, x_10)
find-set(x_2)
find-set(x_9)
```

Part (a) [5 MARKS]

Use the linked-list representation with the weighted-union heuristic (i.e., the shorter list is merged into the longer list).

Part (b) [5 MARKS]

Use the forest implementation with union-by-rank and path compression.

Part (c) [10 MARKS]

Prove, by strong induction on the number of nodes, that every node in the forest implementation of disjoint sets has rank at most $\lfloor \lg n \rfloor$, where n is the number of nodes in the set.

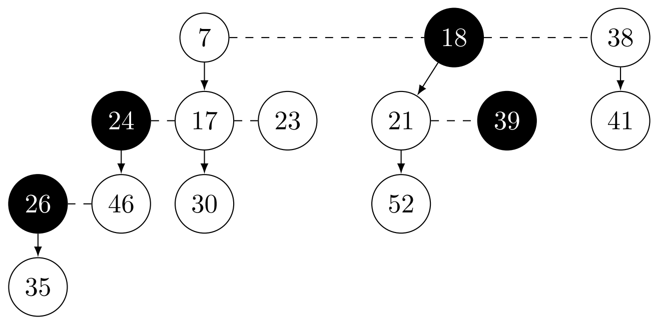
Question 5. Fibonacci Heaps [15 MARKS]

Part (a) [5 MARKS]

Starting with an empty Fibonacci Heap, show a sequence of operations that results in a heap with a root node being marked.

Part (b) [5 MARKS]

Show the result of running `extract-min` on the heap below:



Part (c) [5 MARKS]

Explain why runtime complexity (actual, not amortised) of the `consolidate` algorithm is $\mathcal{O}(r + d)$ where r is the number of root nodes and d is the maximum node degree in the heap.

Question 6. Hashing [22 MARKS]

Suppose we hash the following sequence of keys: $\{5, 28, 19, 15, 20, 33, 12, 17, 10\}$ into a hash table of size 9.

Part (a) [4 MARKS]

Show the resulting hash table if we use the hash function $h(k) = k \mod 9$ and resolve collisions by chaining.

0	
1	
2	
3	
4	
5	
6	
7	
8	

Part (b) [4 MARKS]

Show the resulting hash table if we use the same hash function as above, but resolve collisions by open addressing with linear probing.

0	
1	
2	
3	
4	
5	
6	
7	
8	

Part (c) [3 MARKS]

Suppose we have an open-address hash table that is 75% full and we perform a search for a key k that is not in the table. Give an upper bound on the expected number of probes. Explain your answer.

Part (d) [3 MARKS]

Suppose we have an open-address hash table of size m that contains n elements. If we insert a new element, how long do we expect the probe sequence to be? Explain your answer.

Part (e) [3 MARKS]

Complete the following definition:

Definition 1. *Let* $h : U \rightarrow \{i : 1 \leq i \leq m\}$ *be a hash function. The property of* simple uniform hashing *is defined as:*

Part (f) [5 MARKS]

If we hash n distinct keys into a hash table T of size m, assuming simple uniform hashing, what is the expected number of collisions? Explain your answer.