Final - Practice exam for the Algorithms and data structure course at UBC

Computer Science (The University of British Columbia)

CPSC 221 2017W1: Final Exam

December 11, 2017

1 Who gets the marks? [1 marks]

Please enter your 4 or 5 digit CSID in this box:

2 Choices, Choices! [18 marks]

MC1 [1.5 marks]

Which of the following characterizes an Abstract Data Type?
- ○ Any class containing at least one pure virtual function.
- ○ A collection of class data members.
- ○ A description of the implementation of a data structure.
- ○ A description of the functionality of a data structure.
- ○ None of these options is correct.

MC2 [1.5 marks]

Consider an array based implementation of a stack, and suppose that it is initially empty. Upon n `push` operations the array will be resized in such a way that the running time per push is $O(1)$ per operation, on average. How many times is the array resized over the n `push`es, using this scheme?
- ○ $O(1)$
- ○ $O(\log n)$
- ○ $O(n)$
- ○ $O(n \log n)$
- ○ $O(n^2)$

MC3 [1.5 marks]

Suppose you do post-, pre-, in- and level-order traversals of an arbitrary Binary Search Tree with more than one node. Which of the following statements is (are) true about the traversals?
- ○ The root is output first in all the traversals.
- ○ The largest value in the tree is output last in exactly 2 of the traversals.
- ○ There exists a tree for which at least 2 of the traversals output the nodes in increasing order.
- ○ Exactly two of (a), (b) and (c) are true.
- ○ None of the above are true.

MC4 [1.5 marks]

Suppose a friend claims his/her AVL Tree `remove` function runs in time $O(\log(n^3))$. After thinking about it you have some concerns. Which of the following observations are valid?
- ○ The algorithm is inefficient because it is asymptotically slower than $O(\log(n))$.
- ○ This running time only reflects average running time, not worst case.
- ○ This running time neglects to account for rotations.
- ○ At least two of (a), (b) and (c) are valid concerns.
- ○ The running time is reasonable and you are not concerned.

MC5 [1.5 marks]

Suppose that you have a binary tree of height h containing n nodes. What is the running time of an efficient algorithm to determine if the tree is in order. (i.e at every node k, all the elements in the k's left subtree are smaller or equal to k's key and all the elements in the k's right subtree are larger or equal to k's key.)

○ $O(1)$
○ $O(h)$
○ $O(n)$
○ $O(n^2)$
○ None of the options is correct.

MC6 (2.5pts)

Suppose that, for every node v in a binary tree containing n nodes, you wish to compute the length of the longest path from v down to a leaf, storing that distance in the node. What is the running time of the best algorithm to do this?

○ $O(\log n)$
○ $O(n)$
○ $O(n \log n)$
○ $O(n^2)$
○ None of these answers is correct.

MC7 [1.5 marks]

Suppose we have run Dijkstra's algorithm on a graph to find a shortest path from vertex u to vertex v. Call that path P. a) If we increase the weight of every edge in the graph by a constant amount, is path P still a shortest path from u to v? b) If we divide the weight of every edge by 2, is path P still a shortest path from u to v?

○ a) Yes, b) Yes.
○ a) Yes, b) No.
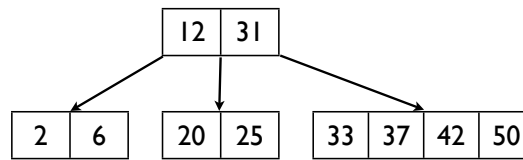○ a) No, b) Yes.
○ a) No, b) No.

MC8 [1.5 marks]

Which of the following expressions represents the *load factor* for a hash table of size t containing s keys?

○ $s + t$
○ $s * t$
○ s/t
○ t/s
○ None of these is the load factor.

MC9 [1.5 marks]

Consider the BTree below, and assume that the root node has been loaded into main memory. How many disk seeks are necessary in a search for key 37?

- ○ 0
- ○ 1
- ○ 4
- ○ 5
- ○ None of the above is correct.

MC10 [1.5 marks]

What is the best definition of a collision in a hash table?
- ○ Two entries are identical except for their keys.
- ○ Two entries with different data have the exact same key.
- ○ Two entries with different keys have the same exact hash value.
- ○ Two entries with the exact same key have different hash values.
- ○ None of the above is correct.

MC11 [1.5 marks]

Suppose we run depth first search on a connected, undirected graph with n vertices, each of which has degree 4. During the traversal, with v as the current vertex, we mark a previously unseen edge as a "thunderbird" edge when one endpoint is v, and the other endpoint is an earlier visited vertex. How many edges will be labelled "thunderbird" edges?
- ○ $n - 1$
- ○ $n + 1$
- ○ $2n$
- ○ $2n - 1$
- ○ The answer cannot be determined from the information given.

MC12 [1.5 marks]

Suppose we want to determine whether or not a simple undirected graph contains an edge, (u, v). What is the tightest worst case asymptotic running time of the best algorithm to do this if the graph is implemented using an adjacency matrix?

- ○ $O(1)$
- ○ $O(n)$
- ○ $O(m)$
- ○ $O(\min\{deg(v), deg(u)\})$
- ○ None of these accurately describes the running time.

3 Efficiency [12 marks]

Each item below is a description of a data structure, its implementation, and an operation on the structure. In each case, choose the appropriate running time from the list below. The variable n represents the number of items (keys, data, key/data pairs, or vertices) in the structure. Assume that graphs are simple, and that keys are comparable. In answering this question you should assume the best possible implementation given the constraints, and also assume that every array is sufficiently large to handle all items (unless otherwise stated). Place the letter of the appropriate asymptotic upper bound on the running time in each box below.

A $O(\log n)$

B $O(n)$

C $O(n \log n)$

D $O(n^2)$

E $O(n^2 \log n)$

MC13 [1.5 marks] [ ] Finding a representative element of a disjoint set, implemented using `upTrees` with smart union and no path compression.

MC14 [1.5 marks] [ ] Build a heap, given an unsorted array of keys.

MC15 [1.5 marks] [ ] Determine whether or not an undirected graph has a cycle.

MC16 [1.5 marks] [ ] Find a minimum spanning tree in a sparse graph.

MC17 [1.5 marks] [ ] Find a spanning tree in a connected, undirected graph.

MC18 [1.5 marks] [ ] Print the keys in a binary search tree in descending order.

MC19 [1.5 marks] [ ] Build a binary search tree (not AVL) with keys that are the numbers between 0 and n, in that order, by repeated insertions into the tree.

MC20 [1.5 marks] [ ] Remove the right subtree from the root of an AVL tree, and restore the height balance of the structure.

4 TreeRank [15 marks]

In this problem we are going to define a new characteristic of Binary Trees called *rank*, and then we will ask you to prove something about trees of rank r. A loose definition of *rank*: The rank of a binary tree T is the height of the largest perfect tree embedded in T.

More formally: the *rank* of a binary tree T is recursively defined as follows:

- if $T = \{\}$, then $rank(T) = 0$.

- otherwise, $T = \{root, T_L, T_R\}$, and

  - if $rank(T_L) = rank(T_R)$, then $rank(T) = rank(T_L) + 1$.
  - otherwise, $rank(T) = \max\{rank(T_L), rank(T_R)\}$.

(a) **[3 marks]** In the spaces below, draw trees of rank 1, 2, and 3, each containing as few nodes as possible.

_____   _____   _____

(b) **[2 marks]** In the spaces below, draw two significantly different trees of rank 3, each containing 10 nodes. Do not just make the two trees mirror images of one another.

_____   _____

(c) **[5 marks]** Let $N(r)$ denote the least number of nodes in a tree of rank r. Give a recurrence for $N(r)$. You must justify your answer using the definition of *rank*.

(d) **[5 marks]** We solved the recurrence and found a closed form solution for $N(r)$ to be $N(r) = 2^r - 1$, $r \geq 0$. Prove that our solution to your recurrence from part (c) is correct by finishing this inductive proof:
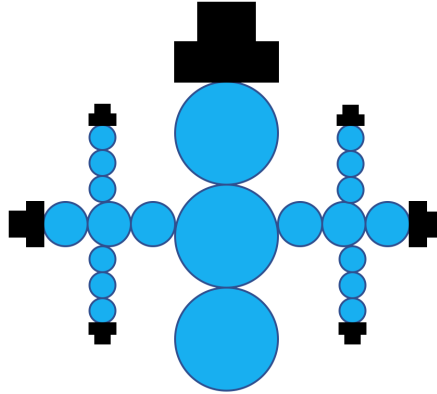
Consider an arbitrary non-negative integer r.

- If $r = 0$ then the closed form gives $N(r) = $ _____, which is also the base case of the recurrence.
- otherwise, if $r > 0$ then by an inductive hypothesis that says,

_____

we have $N($_____$) = $ _____,

so that $N(r) = $ _____ $ = $ _____, which was what we wanted to prove.

5 Snow People [15 marks]

A recursive snow person (RSP) is defined as follows: a RSP of size 1 is a stack of 3 snowballs with a hat on its head. A RSP of size n is a stack of 3 snowballs with a hat on its head, and two arms, each of which is a RSP of size $n - 1$.

(a) **[2 marks]** The RSP above has size 3, but it aspires to be size 4! Add all the tiny circles and hats to our drawing to turn it into RSP of size 4.

(b) **[4 marks]** Let $B(n)$ denote the number of snowballs in a RSP of size n, for $n \geq 1$. Fill in the blanks below to complete the recursive definition for $B(n)$.

$$B(n) = \begin{cases} \boxed{} & \text{when } n = 1 \\ \boxed{} B(\boxed{}) + \boxed{} & \text{when } n > 1 \end{cases}$$

(c) **[5 marks]** Complete the table below. It will help you devise a closed form expression for $B(n)$.

n	B(n)	B(n)/3	B(n)/3 + 1	A nice expression for B(n)
1				
2				
3				
4				
5				

(d) **[4 marks]** Write the closed form for the expression here: $B(n) = \boxed{}$

(e) Oops, we've run out of space for a proof.

6   Quists [15 marks]

In each of the problem segments below, we have given you "before and after" models of linked lists. Your task is to transform the "before" into the "after" using simple pointer manipulations on the list nodes. Refer to the elements of the list nodes using the `listNode` class below. Your solutions should follow these guidelines:

- You may declare `listNode` pointer variables to use in navigating the lists. When you are finished with them, just set them to `NULL`.

- You must never refer to the `data` member of the `listNode` class.

- You may write loops to simplify your solutions, but your answers don't need to be general... they just need to work on the given lists. (Don't worry about even/odd length, or empty lists, for example.)

- Any variables listed in the picture can be used in your solution.

- If a node is removed from a list, be sure to free its memory!

```
class listNode {
   public:
      int data;
      listNode * next;
      listNode(int e): data(e), next(NULL) {}
};
```
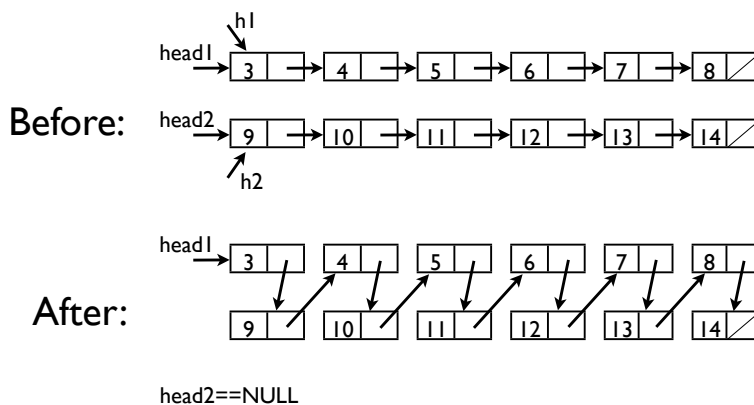
(a) **[5 marks]**
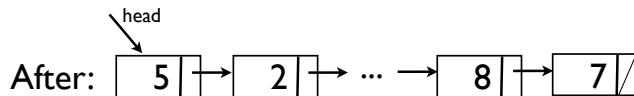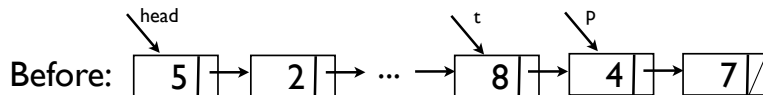
(b) **[5 marks]** Try to do this one using only one additional `listNode *` variable, and without referring to either of the head pointers.

Before:

After:
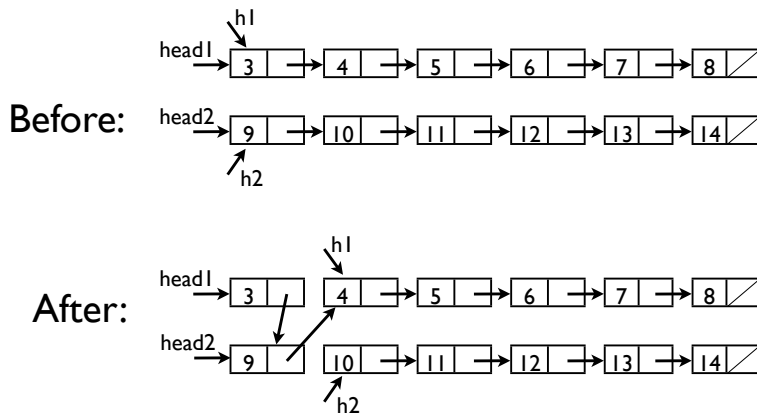
(c) **[5 marks]** Note that trying to change all these pointers explicitly will burn too much exam time. Use a loop!

Before:

After:

head2==NULL

7   Kth [20 marks]

In this problem we will investigate a structure commonly used to maintain the kth largest value in a dynamically changing data set. Assume that the data is a streaming sequence of integers, and that our structure will be designed to report the kth largest value of all the data seen in the stream, so far. A partial definition of class `kthStream` is found below. Note, in particular, functions `getkth()` and `insert(int p)` which are used to report and remove the kth largest value, and to insert new data into the structure, respectively.

```
class kthStream {
public:
   kthStream(int k);
   int getkth();
   void insert(int p);
private:
   int k;
   minPQ sun;
   maxPQ rain;
};
```

The private members of the `kthStream` class are an integer value of k initialzed in the constructor, and two priority queues, one of which (`minPQ`) facilitates function `removeMin`, and the other one of which (`maxPQ`) facilitates function `removeMax`. The priority queue classes also provide public functions `insert(int p)`, `size()` - which tells you the number of keys in the priority queue, and `top()` - which tells you the value of the root without removing it from the priority queue. `size()` and `top()` run in constant time.

As a simple example, in the stream of numbers 3, 18, 24, 8, 30, 7, 12, the third largest value is 18, since only 30 and 24 are larger.

(a) **[3 marks]** One of the priority queues in the `kthStream` class should be used to store all the values in the stream *greater* than the kth value. Which one is it: `sun` or `rain`?

(b) **[6 marks]** Complete the code below to implement the `kthStream insert(int p)` function.

```
void kthStream::insert(int p) {

        if (p > _____.top()) {

                _____.insert(p);

            if (_____.size()  > _____)

                        _____.insert( _____.remove_____);

        } else _____.insert(p);
    }
```

(c) **[5 marks]** Next we will write the `getkth()` function. It should return the kth largest value among all the numbers seen so far. If there have not yet been k numbers, simply return -1. Note that `getkth()` should also remove the value from the structure. Complete the code below.

```
int kthStream::getkth() const {




}
```

(d) **[2 marks]** What data structure(s) would you use to implement the priority queues?

(e) **[2 marks]** What is the running time of `kthStream`'s `insert` function, assuming data of size n, and using your data structure from part (d)?

(f) **[2 marks]** What is the running time of `getkth`, assuming data of size n, and using your data structure from part (d)?

8 Biparts [20 marks]

A graph is said to be "bipartite" if the vertices in the graph can be labelled by two different labels (i.e. colors) so that no adjacent vertices are labelled the same.

(a) **[2 marks]** Draw a bipartite graph with 5 vertices and 5 edges. Please use gold (G) and blue (B) as your labels. (We've drawn the vertices for you, below. You just add the edges and labels.)

①
② ⑤
③ ④
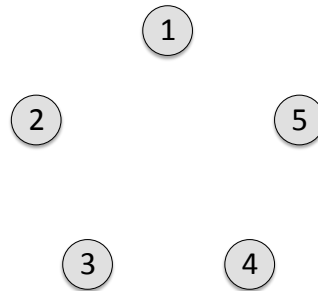
(b) **[2 marks]** Redraw your graph from part (a) so that the blue nodes are in the box on the left, and the gold nodes are in the box on the right. Make sure you use exactly the same edges you used in part (a). Next, emphasize the edges in one cycle in your graph, by drawing the edges on the cycle with a double line.

blue   gold
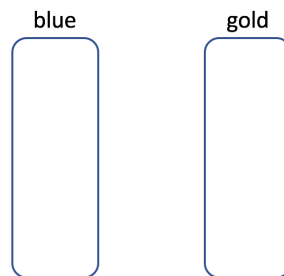
(c) **[2 marks]** Draw a graph with 5 vertices and 6 edges that is NOT bi-partite.

①
② ⑤
③ ④
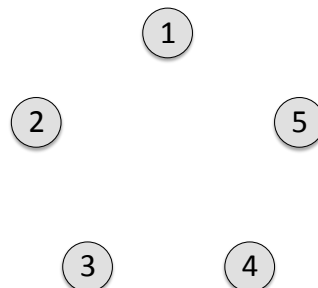
(d) **[4 marks]** Think of an efficient algorithm to determine whether or not a given graph is bipartite. Your algorithm is a slight modification of which of the following? (circle one)

DFS   BFS   Prim's MST   Kruskal's MST   Dijkstra's Shortest Path

(e) **[4 marks]** How will your algorithm know if the input graph is NOT bipartite? Your answer should be a simple succinct sentence.

(f) **[2 marks]** What is the running time of your algorithm from parts (d) and (e) if the graph is implemented using an adjacency list? (As always, assume the graph has n vertices and m edges.)

(g) **[4 marks]** Suppose we have a bipartite graph with an even number of vertices, half of which are labelled blue, and half gold. Such a graph can have no more than _____ edges. (Fill in the blank with an expression in terms of n, the number of vertices in the graph. Be as exact as you can–we are not looking for a big-O response.)

9 Lost [20 marks]

In this problem you will use Kruskal's algorithm for Minimum Spanning Trees to generate a random 3×3 maze. Every cell in the grid represents a node in a graph, and every wall in the grid corresponds to an edge between its adjacent cells. Each wall is labelled with a randomly generated priority, and those priorities correspond to the cost of removing a wall. Your algorithm should attempt to remove walls from the maze in order from least to greatest priority, and it should only remove a wall if there is no path between its adjacent cells.

(a) **[4 marks]** Use the middle table below to model a priority queue. Insert all of the relevant data.

(b) **[4 marks]** Execute Kruskal's algorithm to completion to create a random maze. When executing the algorithm, make it clear when you employ `removeMin` by drawing a line through every key you remove from the priority queue. Only `removeMin` as many times as you need, and no more.

(c) **[2 marks]** Use the unlabeled grid to illustrate your random maze after execution of the algorithm.

Draw the maze here!

(d) **[4 marks]** How many walls will be removed in constructing an $n \times n$ maze?

(e) **[2 marks]** What ADT would you use to assure that your algorithm creates no cycles?

(f) **[2 marks]** Name the 2 classic functions implemented in the ADT from part (e).

(g) **[2 marks]** Complete the following sentence:

The functions in part (f) have running times that are very nearly $O(\rule{1cm}{0.4pt})$.

10 MST [15 marks]

Suppose darkened edges in the diagram below are all part of some minimum spanning tree of the graph.
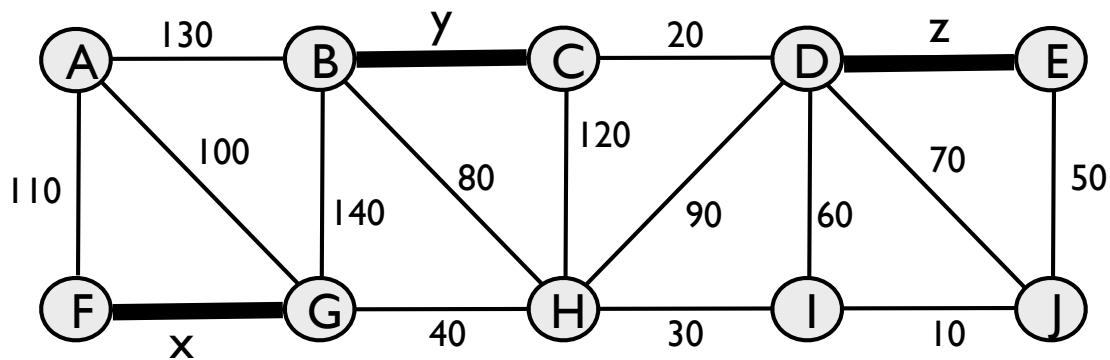
(a) **[6 marks]** List the weights of the other edges in the MST in order of increasing weight.

10 _____ _____ _____ _____ _____

(b) **[3 marks]** Circle the possible values of x.

5   15   25   35   45   55   65   75   85   95   105   115   125   135   145

(c) **[3 marks]** Circle the possible values of y.

5   15   25   35   45   55   65   75   85   95   105   115   125   135   145

(d) **[3 marks]** Circle the possible values of z.

5   15   25   35   45   55   65   75   85   95   105   115   125   135   145

The following questions are for stress relief only and will **NOT** be graded. Enjoy!

11   Santa's Big Squeeze [0 marks]

Assuming a spherical Santa with a diameter of 2 metres, to what height must he deform (volume-preserving) to enter a home through a chimney with a rectangular opening of 0.5m $\times$ 0.25m?

12   Rocket Reindeer [0 marks]

Assume a frictionless system and a reindeer with a mass of 200kg which flies by rocket propulsion. At what velocity must a pile of "rocket fuel" with a mass of 0.8kg be expelled from the rear end of the reindeer in order to propel the reindeer forward at a velocity of 60km/h?

13   NSAnta [0 marks]

Santa has installed a surveillance network which monitors 350 million households in North America. Periodically each household is polled and 8 bytes of data are returned to Santa's spy server, indicating naughty or nice activity for the identified household. If Santa's ISP limits his incoming bandwidth to only 10 Mbps, what is the maximum household polling frequency that will not oversaturate his connection?

14   Elf Exploitation [0 marks]

After centuries of oppression, Santa's elves have demanded that Santa increase their minimum wage from 5 cookies and a 300mL glass of milk per day to 6 cookies and 400mL of milk per day. Suppose Santa purchases milk and cookies wholesale at $0.15 per cookie, and $0.80 per litre of milk, and currently employs 3000 elves in his workshop. How many elves must be laid off for Santa's toymaking operation to incur only a 10% cost increase as a result of the elves' pay raise?

This page intentionally left (almost) blank.

If you write answers here, you must CLEARLY indicate on this page what question they belong with AND on the problem's page that you have answers here.

This page intentionally left (almost) blank.

If you write answers here, you must CLEARLY indicate on this page what question they belong with AND on the problem's page that you have answers here.