CS 311 Data Structures and Algorithms, Fall 2020

Final Exam

Due 2:15 pm Wednesday, December 9, 2020

Name: _____

The exam is worth 150 points. 7 pages (including this cover page), 11 problems.

Instructions

Answer all of the problems on the following pages. You will need to refer to "Information for the Final Exam", a PDF linked from the class webpage.

This exam is to be done individually. You may reference an class materials, books, your own notes, or anything on the web, but not other people.

Turn in a PDF on UAF Blackboard, under "Final Exam" for this class. The PDF should have your name on the first page. It should be turned in by 2:15 pm Wednesday, December 9, 2020.

1. [12 pts total] **Order.** In each part below, write the order of the given expression using asymptotic notation and in words. *Be sure to place each into one of the categories listed in class.*

1a. [6 pts] 5432.

Asymptotic Notation:

In Words:

1b. [6 pts] $6n \log n + 5n^2 - 800$.

Asymptotic Notation:

In Words:

2. [18 pts total] **Terminology.** In each part, explain the meaning of the boldface term.

2a. [6 pts] While a Hash Table is being used, a **collision** occurs.

2b. [6 pts] A certain function is **exception-neutral**.

2c. [6 pts] A certain sorting algorithm is **stable**.

3. [12 pts total] **Amortized Time.**

3a. [8 pts] **What is the difference** between *constant time* and *amortized constant time*?

3b. [4 pts] **Give an operation that**, if intelligently implemented, is amortized constant time, but not constant time.

4. [15 pts total] **Binary Trees.** Questions below refer to the Binary Tree pictured in "Information for the Final Exam".

4a. [3 pts] Give a **preorder** traversal of this tree.

4b. [3 pts] Give an **inorder** traversal of this tree.

4c. [3 pts] Give a **postorder** traversal of this tree.

4d. [6 pts] Is this tree **strongly balanced**? **Circle** your answer, and **explain** why.

YES NO

Explanation:

5. [9 pts total] Modern processors use *cache prefetching;* that is, when a memory address is loaded, nearby addresses are also loaded and stored in the processor for faster access. This has implications for the relative desirability of (smart) arrays vs. Linked Lists.

5a. [3 pts] **Which one** (array or Linked List) typically becomes more desirable when cache prefetching is done? *Circle one.*

Array Linked List

5b. [6 pts] **Explain why** the data structure in part a becomes more desirable.

6. [18 pts total] **Efficiency of Table Operations.** Complete the following. State the order of the three single-item Table ADT operations (retrieve, insert, delete) when a Table is implemented as indicated below. *Remember that you are being asked for* ***worst-case*** *performance!*

Implementation	**Retrieve**	**Insert**	**Delete**
(Smart) **Sorted** Array			
Binary Search Tree			
Red-Black Tree			
Hash Table			

7. [12 pts total] **Invisible Functions.**

7a. [6 pts] When we talk about a *move constructor* or *move assignment operator*, what do we mean by "move"?

7b. [6 pts] Why does your instructor refer to the member functions like the copy and move operations as "invisible functions"?

8. [15 pts total] **Best Practices.**

8a. [8 pts] In C++, destructors are allowed to throw exceptions. But it is a bad idea for them to do so. Why is it a bad idea?

8b. [7 pts] Your instructor says that the first goal, when writing code, should be to get it to compile, rather than getting it all written. Why is this?

9. [12 pts total] **Why are These Functions Defined?**

9a. [6 pts] Class template `std::vector` defines `operator<`. So we can do:

```
std::vector<int> x, y;
if (x < y) cout << "Hi!" << endl;
```

But there is no good reason to do that. Why does this `operator<` exist?

9b. [6 pts] The STL includes global functions `find`, which does Sequential Search, and `binary_search` & others, which do Binary Search. All of these work with `std::set`. But `set` has its own member function `find`. Why?

10. [12 pts total] **Why are These Functions NOT Defined?**

10a. [6 pts] Class templates `list` and `deque` have members `push_back` & `pop_back`, and also `push_front` & `pop_front`. But `vector` only has the *back* versions. Why does it not have the *front* versions?

10b. [6 pts] When we pop a Stack, we usually want to know what was popped. However, member function `pop` of `std::stack` returns nothing; `std::stack` has no member that removes & returns the top item Why not?

11. [15 pts] **Analyzing a Recursive Algorithm.** Consider the following function template `meow`, which takes a range of `int` values, specified using a pair of random-access iterators.

```
// Requirements on Types:
//     RAIter must be a random-access iterator type.
//     The value type of RAIter must be int.
template<typename RAIter>
int meow(RAIter first, RAIter last)
{
    size_t n = last - first;   // Size of range

    // BASE CASE
    if (n <= 1) return 7;

    // RECURSIVE CASE

    int val = first[1];        // Must exist, since n >= 2

    RAIter mid = first + n/2;  // Iterator to middle of range

    int a = meow(first, mid);  // Recursive calls
    int b = meow(mid, last);

    return a * b + val;        // Final value
}
```

Letting *n* represent the size of the given range, find the order of this function. ***Show your work****. Note that* `meow` *is recursive!*