CSCI 262 Midterm I Review

What to study – Overview

Basic C++ (mostly review material)
- functions
- function overloading
- value & reference parameters
- memory model (function call stack)
- classes & objects
- syntax (scope resolution op, dot op)
- member variables & functions
- constructors
- visibility
- basic I/O

ADTs/Data structures:
- Vectors
- Stacks
- Queues
- Sets
- Maps

Other Important Stuff:
- Libraries & interfaces
- Pointers & arrays
What NOT to study - what won’t be on this exam

- Dynamic allocation/deallocation will **not** be on this exam.
- Debugging will not be a topic on **any** exam.

Basic C++

- Functions
  - How to write them, how to use them
  - What does pass by **value** mean?
  - What does pass by **reference** mean?
- Function Overloading
  - Can two functions have the same name if they have different parameters?
- Memory model (function call stack)
- Classes and objects
  - Know how to write a class, how to implement class methods using the scope resolution operator (::)
  - Know what making class members and methods public/private means
- Basic I/O
  - Know how to use cout/cin
- Syntax
  - Know what the scope resolution operator is for (::)
  - Know what the dot operator is for
  - Know what the arrow operator is for (->)
- Member variables and functions
  - What’s it mean for a variable/function to be a member of a class?
  - What’s different about the syntax when implementing a member function vs. a normal function
- Constructors
  - Know how to write a constructor (i.e. what the syntax looks like.
  - What is a constructor for? When does it run?
- Visibility
  - Know what making class members and methods public/private means
Abstract Data Types/Data Structures

- Know how each of these work and what functions they offer you, but don’t memorize the functions word-for-word
  - (e.g. do remember that you can add elements to the back of a vector, but don’t memorize that there is an underscore in `vector.push_back()`)
  - Function prototypes and small comments telling you what each function does is provided for you on a separate page - see practice test on Piazza

- Vectors
  - Make sure you know the syntax differences between declaring an array of ints vs a vector or ints, for example

- Stacks
  - Does `pop()` return the value on the top, or just remove it?
  - Can you iterate over a stack using a range based for loop?

- Queues
  - Does `pop()` return the value at the front, or just remove it?
  - Can you iterate over a queue using a range based for loop?

Abstract Data Types/Data Structures (Continued)

- Sets
  - What happens if you add the same element to a set twice?
  - In a range based for loop, what order are elements of a set accessed in, assuming we’re using a standard set?
  - How do you check if an element is contained within a set?

- Maps
  - What happens when you use square brackets and the assignment operator to assign a value to a key that’s already in the map? (e.g. `my_map['a'] = 7`)
  - What happens when you use the methods `insert` or `emplace` with a key that’s already in the map? How is this different from using square brackets?
Abstract Data Types/Data Structures (Continued Again)

- Maps (Continued)
  - What happens when you try to access an element that isn’t in the map using square brackets? (e.g. `my_map[ 'a' ]++` when ‘a’ wasn’t a key in the map)
  - How can you iterate over a map using iterators and range based for loops?
    - Where does the pair class help?
    - Where can the keyword ‘auto’ help?
  - In a range based for loop, what order are elements of a map accessed in, assuming we’re using a standard map?

Other Important Stuff

- Pointers and Arrays
  - What is a pointer?
  - How are pointers and arrays similar?
  - What does the ampersand do? (&)
  - What happens when you add values to a pointer? (Understand pointer arithmetic)
  - Do arrays have a `.size()` function?
  - What does “dereference” mean? How do you dereference a pointer?
  - Can you use the dot operator on a pointer to an object to access the object’s members?
  - What the heck does this arrow thing do? ->

- What does "compile" mean? What happens when you compile?
- Understand how the modulus operator works
- Libraries & interfaces
  - Know that an "interface" is the user facing part of a library, and that a library is a collection of functions/classes made by someone and packaged together for re-use.