Midterm Review

Liam Strand and Lucas Maley

October 24th, 2022

Potential question formats:

Word Bank

Students could be given a word bank and a set of statements about C++, programming, and computer science concepts with various words replaced by blanks, and would need to select an item from the word bank to fill each blank space.

Short Response

Students may be asked to briefly explain a concept from class in 1–2 sentences or provide a small piece of information, such as a terminal command. They also could be given a section of code and asked to explain what it is doing or why it is producing a certain output.

Programming

For some problems, students will be given a starter code file and asked to modify it to implement a certain functionality or behavior. Some problems might only require students to add to an already fleshed-out program, while others could ask them to write more of the code—however, in all cases they will be given at least a starter file with the necessary boilerplate code.

For programming problems, students will be given a demo program that produces the intended output. They will be responsible for testing their code themselves and creating their own input files to do so (any additional files they make won't be submitted).

Debugging

Similar to the programming problems, students could be given a relatively "complete" program that has several errors in it and be asked to identify and fix these errors. Again, they will likely have access to a demo and can make input files to test their fixed program against it.

Potential exam topics:

General C++ Concepts/Syntax

Students will need to be familiar all of the basic programming tools they have learned since the start of class. They should be comfortable using (and potentially explaining):

- Variables (and their types)
- Basic operations (math operators, `+=` and `++`, etc.)
- Conditionals (and boolean expressions/operators)
- Loops (`for` and `while`)
- Arrays (more focus on regular arrays than 2-D)
- Functions: Students will need to understand how functions are defined and called, and may be asked to write functions for programming problems. It will be important for them to understand the concept of scope and how it may affect programs that use functions.
- File I/O: Students should understand how to read in data from a file, including how to set up file reading with all of the usual steps (opening/checking/reading/closing). They should be comfortable reading from a file in a loop with `eof()` and storing information in a data structure (such as an array).
- Students should also have a general understanding of how we can use command-line arguments to get information from the user, as this is often paired with file reading.

Pointers

Students should be comfortable using and explaining pointers. In particular, they will need to understand the syntax of declaring pointers with `*` and what it means when we apply the `*` and `&` operators to them after they've already been declared.

- Students may be presented with code that uses pointers, and asked to trace the operations being performed in order to explain what the code is doing and/or figure out what it will output.
- It will be especially important that students understand how pointers can be used in conjunction with functions and how this affects scope. They may be presented with (or asked to write) functions that dereference and modify pointers that have been passed as parameters, and will need to understand how this can affect values elsewhere!
- Students should be comfortable working with pointers to arrays, and may need to work with/interpret code that uses pointer arithmetic to navigate arrays.

Dynamic Memory

This topic is not likely to feature heavily on the exam, but students may be asked about the syntax of `new` and `delete` (don't forget `[]` when necessary!), what it means to use them, and why we would ever need to use dynamic memory.