CSci1113 Spring 2018
Study Guide: Midterm Exam 2

Midterm 2 will be held in the usual classroom for the first 50 minutes on the week of April 10 (week 12). Thus, the night class takes it on 4/11 (Tuesday). This study guide has three parts:

- General comments.
- A detailed list of topics.
- A few example study questions.

I. General Comments

Midterm 2 will be Tuesday, April 10 in the usual room at the usual time and room (6:30pm in Tate 101).
Here are some additional comments:

- The exam will be designed for 50 minutes. So please make sure to arrive on time.
- The exam will be open book and notes. You may also use electronic devices, to store information, but you cannot use one to process information (i.e. a compiler).
- Although the exam is open book, there will not be enough exam time to consult it extensively. So you should know the material well enough that you do not need to look up many items.
- The exam will cover weeks 5 to 8. Specifically, this is material on functions (Chapters 4 and 5), file I/O (Chapter 6), arrays (Chapter 7), strings (Ch. 8.1 and 8.2) and recursion (Chapter 14). There will not be questions that focus exclusively on the material in the earlier Chapters 1 -- 3. However, since the material in those chapters was fundamental, it will often appear implicitly in problems. For example, a problem asking you to read information from a file (Ch. 6) will often involve correctly using a loop (Ch. 3).
- Make sure you understand the homework and lab problems, since exam questions will often be on the general topics in those homeworks and labs, and sometimes be quite similar to tasks you needed to do in solving those problems.
- Expect 6 to 8 questions (some of which are multipart) based on different topics, and the ability to drop one question.

II. Topics

You are responsible for all the material presented up to week 8, up to and including recursion in Chapter 14. This includes the reading assignments, lectures, lab exercises, and homework exercises. Test questions may be drawn from any of these, but are more likely to involve concepts or C++ constructs that have appeared often in the lectures, labs, and homework problems.

The following recent material will *not* be on Midterm 2 (but might be on the final exam):

- Material on class basics (Chapter 10).
- Material on friend functions and overloaded operators (Chapter 11).
- Material on pointers and dynamic arrays (Chapter 9).

The following list gives more detail on topics that *might* be on the exam. This list is not comprehensive, but includes the most likely topics.

1. Function basics
2. Function return types, parameters, and arguments
3. Variable scope
4. Call-by-value and call-by-reference
5. Opening and closing files, input and output streams
6. Reading from and writing to files
7. Array basics: declaration, initialization, element access, etc.
8. Common array operations
9. Arrays and functions
10. One-dimensional and multidimensional arrays
11. Common array errors (e.g. index out of range)
12. String types
13. C++-style string basics: declaration, initialization, element access, etc.
14. getline
15. Common string operations
16. Recursion basics
17. Identifying base cases and recursive reduction steps
18. Writing recursive function in C++

As usual, exam problems may ask you to do the following types of tasks:

1. Answer short questions about any of the topics above.
2. Trace the execution of a code segment.
3. Locate syntax, runtime and logic errors in given C++ code.
4. Modify given C++ code to solve a problem.
5. Write C++ code (ranging from one-line answers on short questions, to an entire function, and/or short program on more involved questions).
6. Solve a given problem and write the solution as a C++ function or program.

Most problems in Midterm 2 will involve writing at least some code.

III. Sample Problems

Here are a few sample study problems. These problems illustrate some (but not all) of the types of problems that may be on the midterm exam. We will not post solutions; however, you are welcome to discuss these problems with others, write and run code solutions on a computer, or go to office hours and discuss your answers with the TAs.

A. Write a function that takes nine doubles as arguments: two vectors (ux, uy, uz) and (vx, vy, vz) , computes the cross product of those vectors, and stores the result in pass-by-reference parameters (wx, wy, wz). (If you do not remember the formula for the cross product, look it up.) Here each component is of type double, and the components are not grouped as arrays (so each component is a separate parameter).

B. Redo Problem A except assume that the parameters are three arrays u, v, and w, each of which are three-element arrays.

C. Write a function that takes in a C++-style string s (from #include <string>), as well as an integer n, and then returns true if any substring of length n in s repeats itself, and false otherwise. For example, if s is "toe-to-toe" and n is 3, then the function would return true since "toe" occurs twice within s. However, if n were 4, the function would return false. As a second example, if s is "singing" and n is 2, then the function would return true since, e.g., "in" appears more than once (as does "ng"). In this problem you may use any of the C++ string member functions.

D. Suppose you have two arrays of ints, both of length n. Write a *recursive* C++ function that will return true if the arrays are the same (i.e., contain identical values in the identical order), and false otherwise. Specifically, (i) write down the base case(s); (ii) write down the recursive reduction step(s), and (iii) write a C++ function implementing your solution.

E. Suppose you have a 2D int array c of colors, where c[i][0] is the red component of the color with index i, c[i][1] is the green component, and c[i][2] is the blue component. Write a C++ code that takes in the array c, its size n, and a search color with components r,g,b (these search color components are all separate int variables, i.e., they are not in array); the function should return true if the search color appears anywhere in the array, and false otherwise.

F. Write a function that takes in an array strArray of C++-style strings (from #include <string>), as well as a character ch, and outputs to the console the indexes of any strings in strArray in which the character ch appears more than once.