CS 112 - Exam 1 Review Suggestions - Fall 2022

last modified: 2022-09-14

Exam 1 BONUS Opportunity

- You can receive (a maximum) ***5 POINTS BONUS*** on Exam 1 if you do the following:
  - Make a **hand-written** Exam 1 study sheet.
  - Submit a photo or scan of it saved as a .pdf, .png, .jpg, .gif, or .tiff to Canvas **by 9:00 am on Friday, September 23** such that I can read at least some significant CS 112 Exam 1 material on it.
  - Please let me know if you have any questions about this, and I hope it helps you in reviewing course concepts more effectively before Exam 1.
  - You are **encouraged** to have this **at hand** as you are taking Exam 1.

Exam 1 Set-up

- You will take Exam 1 in Canvas while you are in BSS 317 on Friday, September 23.
  - You are **required** to be in BSS 317 while you are taking Exam 1.
    - If you attempt to take Exam 1 from anywhere else, you will receive a grade of **0** on the exam.
    - The instructor will take roll at the beginning of lab.
  - It will be set up with a **time limit** of **1 hour 50 minutes**, and will be available from **9:00 - 10:55 am** on Friday, September 23.
  - It will be set up such that **you may only attempt Exam 1 once**.
  - Let the instructor know of any technical difficulties, so they can make provisions as needed!
  - Exam 1 will be set up such that **you will be shown one question at a time**,
    - BUT there will be a list of question-links on the right-hand-side of the Canvas screen, and you can go back and forth between questions during that **one** exam attempt.
  - You are expected to work **individually** on the exam -- it is not acceptable during the exam to discuss anything on the exam with anyone else.
  - You may look up information from your Exam 1 study sheet, on-line, or from the course textbook during the exam, but note that **if you take too long looking up material, you may have trouble completing the exam during the time period**.
  - I expect there will be some multiple-choice questions, and the rest will be short- to medium-answer questions.
      - You will be reading and writing C++ expressions, statements, and fragments, including C++ function definitions.

    - You will be answering questions about concepts as well

- A link to a packet of references and additional instructions - intended for use with Exam 1 - will be linked from the Exam 1 Instructions.
    - So, you can have it open in another browser window while you are taking Exam 1.
    - This is intended both for reference and for use directly in some exam questions.
- Your studying should include careful study of posted examples and notes thus far.
- You are responsible for material covered in class sessions, lab exercises, and homeworks through and including the Week 3 Lab Exercise (Friday, September 9) and Homework 3 (due 11:59 pm on Friday, September 16).
    - This review handout is intended to be a quick overview of especially important material.
    - The Savitch text is very comprehensive; references below to chapters in the text are there just to point out where in the text they are. You will NOT be responsible for all information in those chapters, just the parts we've covered in lectures, labs, and assignments.

      With that in mind:

        - Chapters 1-5,
        - Chapter 6 for file input/output,
        - Chapter 7 for arrays,
        - Chapter 8, Section 8.2, for the C++ `string` class, and
        - Chapter 12, Section 12.1, for separate compilation,

      ...can be useful for additional background reading.
    - TIP: It is **perfectly fine** to retake/read over the short-answer questions in Canvas from Homeworks 1, 2, and 3 as you are studying for Exam 1!

      These are set up for unlimited retakes, and only keep the highest score, so you will not hurt your grade by doing so!
    - although we started discussing writing classes in C++ during Week 4, that will **not** be on Exam 1. (That *will* be fair game on Exam 2 and beyond!)
- Remember that C++ **is case sensitive** - for example, `String` is not the same data type as `string`. You are expected to use the correct case in your answers.
- You are also expected to follow CS 112 course style guidelines in your answers (**including indentation**).
    - You should use the **Formatting Practice Question** linked from the course Canvas site's Home page to practice writing formatted C++ statements BEFORE Exam 1!
    - If you find you are having trouble with this, make sure to come by student hours or ASK ME before Exam 1, so you won't lose points for poorly-indented answers.

Separate compilation for C++ functions

- (Chapter 12, Section 12.1 of the course Savitch text might be useful if you would like additional reading on this topic.)
- When you are writing a non-`main` function in its own `.cpp` and `.h` files to allow for separate compilation:
  - What should go in that function's `.h` file?
    - What is considered good style/CS 112 class style to use for the name in its `#ifndef` and `#define`?
  - What should go in that function's `.cpp` file?
  - You should be able to make use of the class templates for a non-`main` function's `.cpp` and `.h` files to write either or both of these files for a given function.
  - You should be able to write the `#include` that needs to be used in each `.cpp` file that includes a function that will call a given function that is implemented in its own `.cpp` and `.h` files.
- For a function to use something from a C++ standard library, how do you `#include` that standard library in that function's `.cpp` file (What needs to surround its name)?
  - (Note that, if a function header uses something from a standard library, you would need such a `#include` in that function's `.h` file as well.

    For example, if a function has a `string` parameter, it needs to `#include <string>` in both its `.h` file and its `.cpp` file.)
- For a function *a* to use another function *b* you have written that is in its own `.cpp` and `.h` files, how do you `#include` function *b* in function *a*'s `.cpp` file? (What file do you need to include, what needs to surround this file's name?)
- Note: for CS 112, you are expected to use the standard namespace -- that is, you will put:

  ```
  using namespace std;
  ```

  ...after your `#includes`.
- Given the functions involved in a program and the name of the file containing its `main` function, you should be able to write the `g++` command to compile/link/load that program, resulting in an executable program.
  - You should be able to write the command to then run that resulting executable program.

C++ Basics

- (Chapter 2 of the course Savitch text might be useful if you would like additional reading on these topics.)
- Be able to read, write, and answer questions about C++ variable declarations; should be familiar with at least the data types `int`, `double`, `bool`, `char`, and `string`.
- Be able to read, write, and answer questions about C++ assignment statements.

- Be able to read, write, and answer questions about C++ arithmetic operators and expressions using those operators.
- Be able to read code using, write code using, and answer questions about C++ stream input and output.
  - What operator do you use for stream input? If you are reading into a `string` variable, what function can you use for this (for reading in everything up until a newline is entered)?
  - What operator do you use for stream output?
  - What stream is used for input from the keyboard?
  - What stream is used for output to standard output? (usually a console or the screen, depending upon your environment)
- Be able to write a C++ block.
- Be able to read code using, write code using, and answer questions about C++ `if` statements.
  - if an `if` part or `else` part is to contain more than one statement, what must you put around those statements?
- Be able to read code using, write code using, and answer questions about C++ loops.
  - if a loop is to contain more than one statement, what must you put around those statements?
- Be able to read, write, and answer questions about the C++ `bool` type, as well as about C++ `bool` and relational operations.

More Flow of Control

- (Chapter 3 of the course Savitch text might be useful if you would like additional reading on these topics.)
- Be able to write more-complex `bool` expressions.
  - Be able to write C++ expressions of type `bool` that include `&&` and `||` and `!` (or, if you prefer, `and` and `or` and `not`)
  - Be able to give the opposite of a given `bool` expression.
- Be able to read, write, and answer questions about C++ `for` loops.
- Be able to read, write, and answer questions about nested statements, including loops nested within loops, if statements nested within if statements, loops nested within if statements, if statements nested within loops, etc.

Procedural Abstraction and Functions that Return a Value

- (Chapter 4 of the course Savitch text might be useful if you would like additional reading on these topics.)
- Be able to read, write, and answer questions about C++ named constants; understand the restriction on named constants.

  - What is CS 112 course style for how named constants are to be named?
- Be able to read, write, and answer questions about non-`main` functions that return a value.
  - Should be able to read, write, and answer questions about function headers and function bodies.
  - Should be able to write a correct call to a function given its function header.
- This is not in Savitch, but was discussed in class: you should be able to write signature and purpose statement comments meeting the required CS 112 style for C++ functions, as well as tests for them in their opening comment blocks, along with running those tests appropriately in a testing-style `main` function.
  - See the notes about the "design recipe" test-first approach in the Week 1 Lecture 2 posted notes from class,

    the posted basic Design Recipe summary,

    the C++ "graphic design recipe helper",

    and the class examples of non-`main` functions; these functions should all have these parts.
- When a `return` statement is executed within a function, what happens next?
- Understand the difference between a function returning a value and a function printing a value to the screen.
- Understand the different between a function having a parameter and a function reading in a value from a user or a file.
- What is meant by a local variable? What is the scope of a local variable? What is the scope of a parameter variable?

File input/output

- (Chapter 6 of the course Savitch text might be useful if you would like additional reading on this topic.)
- What `#include` directive is needed to use the library for C++ file input/output used in this course and discussed in the course text?
- How do you declare an input file stream using this library?
  - How do you open such an input file stream and attach it to a file? How can you see if this succeeded?
  - How do read from this now-opened input file stream?
  - How do you close this input file stream when you are done?
- How do you declare an output file stream using this library?
  - How do you open such an output file stream and attach it to a file? How can you see if this succeeded?
  - How do you write to this now-opened output file stream?

  - – How do you close this output file stream when you are done?

- When you open an output stream and attach it to a file, what happens to any data already in that file?

Arrays

- (Chapter 7 of the course Savitch text might be useful if you would like additional reading on this topic.)
- Be able to read, write, and answer questions about (static) arrays.
- Should be able to declare a (static) array; should be able to access an individual element of an array.
  - What is the special syntax for initializing an array when it is declared?
- Should be able to loop through an array, doing something to/with each element in that array.
  - What are the preferred loops for doing this?

    (Note that you can always use a classic `for` loop for this; a "foreach" style `for` loop can also work very well *if* the loop's purpose is NOT to fill or change the array being "walked" through!)