CSCI 200 - Fall 2023
Foundational Programming Concepts & Design

Final Project

This assignment is due by Thursday, December 07, 2023, 11:59 PM.
NO LATE SUBMISSIONS WILL BE ACCEPTED

Jump To: **Rubric Submission**

· **Proposal** · **Code** · **Paper** ·

Each week this semester, you have been doing labs and homework assignments that emphasize specific aspects of the C++ programming language. For the last few weeks of the course, you will make use of these language skills in the context of a larger, more realistic project. The goals of this project are:

- To engage in a project which is tailored to your particular interests.
- To have additional experience using the programming constructs covered within the scope of this course.
- To take responsibility for designing and producing a large program, thereby gaining knowledge and understanding of the entire software development process.

The final project is open-ended. You may choose to write a program that plays a game, reads large data files and does a complex calculation with the data, or anything in between. Some detailed requirements are given below, so please read this document carefully.

Requirements - Project Proposal
(due Wednesday, October 11, 2023 11:59 PM)

On Wednesday, October 11, 2023 11:59 PM, a short description of your project is due. You need to submit your description (as a PDF) to Canvas. This document needs to include the following sections WITH the section titles listed below.

- **TITLE:** Include your name, your CSCI 200 section, and a project title.
- **PROBLEM DESCRIPTION:** You will write a one-paragraph description of your project. This paragraph will give the reader a general idea about the program requirements and what problem you are trying to solve. For example: *I will be creating a program that calculates the optimum pair of gears that should be selected on a bicycle, given a degree of incline and current velocity. Users select the type of bike, specify their speed, pain threshold, and degree of incline. The program then informs the user of the front and rear gears that should be selected. But wait, there's more! We then animate this on the screen to illustrate the bike as it climbs or descends a hill. Whee!*
- **DATA DESCRIPTION:** The data for your program consists of three requirements:
  - The UML class diagrams for the classes you will be creating for your program. *Be sure to include comments to describe what your data members and member functions will do.* This section is NOT in paragraph form; instead, this section is pseudocode made up of the UML class diagrams. This section does NOT include actual C++ code, it is only pseudocode.
  - How lists (either an array, vector, linked list, stack, and/or queue -- one-dimensional or multi-dimensional) will be incorporated into your program. What sets of data will need to be stored? How will they be stored - which structure(s) will you use?
  - How files will be used for your data. Will you read sets of data from an external file? Will you write the results of computations to a file? Both? You must manually perform the File I/O via an `ifstream` or `ofstream`. This task cannot be handed off to a third party library (such as SFML loading an image).
- **PROCEDURAL DESCRIPTION:** Include a brief description in pseudocode of how your main program will operate. This section should also NOT be in paragraph form. If you plan to use SFML, be sure to mention that here. This section does NOT include actual C++ code, it is only pseudocode.
- **SPECIAL NEEDS/CONCERNS:** Your Project Proposal should mention any special needs or concerns that the instructor should know about.
  - Will you need extra help on a particularly difficult idea that you will have to conquer in order to make this project work? If you're addressing a specific problem for a non-CS major, you may need to get advice from someone within that department.
  - Will you need to make use of any third party libraries? If yes, what is the library and what task is it accomplishing?

The document you submit will specifically answer the following questions:

1. What class will you create? What data members and member functions will it have?

2. What data structure (array, linked-list, stack, queue) will you use within your project? How will it be used?
3. Where and how will a data file be used?

You will not need to, and should not, do any coding to write this Project Proposal. The purpose of this Project Proposal is to get you to think about the initial design of your final project. (Note: we understand that the initial design you submit on Wednesday, October 11, 2023 11:59 PM is likely to change as you complete your project for the Thursday, December 07, 2023, 11:59 PM due date.)

Your program needs to be more than a database query / filter. If you are working with data, then some data analysis must be performed. It is not enough to read in a large set of data and write out a smaller ordered set of the same data. The data must be transformed, manipulated, etc. There needs to be non-trivial complexity to your program.

NOTE: You have a bit of time to decide upon your topic and a big picture of your design, but then only over about three weeks for implementation. Because of the tight time period for implementation, we strongly encourage you to have most of your design plans done earlier than Wednesday, October 11, 2023 11:59 PM (to give you extra time for implementation). **Many previous students have said "gosh, wish I had gotten started on the final project earlier."**

Your instructor will give you feedback on your Project Proposal after you submit it (e.g., too complex or too simple).

Requirements - Project Code
(due Thursday, December 07, 2023, 11:59 PM)

Your program must use at least one original class, written specifically for this project. You are free to use other classes we have developed during the semester, such as the `LinkedList` class, or classes described in zyBooks, but you must also write and use one original class.

- Your custom class must encapsulate some important data/functionality of your program. This functionality needs to be more complex than the setters/getters for the corresponding data.
- Your custom class must have some data/functions that are private.
- Your custom class must have a well-defined public interface.
- Your custom class must be defined and declared in separate files.
- Your custom class must use const, functions, data types, templates, etc. as appropriate.
- Your custom class must have functionality beyond getters and setters. The class needs to do more computations than simply store data.

Your program must use at least one list. This list needs to be either a list within your class OR a list of objects of your class type.

Your program must make use of File I/O. Data may either be read from or written to a file, or both. Please place your data file in the same directory relative to your `main.cpp` file.

Your program must use functions where appropriate.

Your program must use constants where appropriate.

Your project must make use of ample commenting. There should be enough documentation to allow another programmer to easily make modifications or enhancements.

Your program must adhere to our **CSCI 200 style guidelines**.

Your program must adhere to our **CSCI 200 best practice guidelines**.

Requirements - Project Paper
(due Thursday, December 07, 2023, 11:59 PM)

Create a text file called `final.txt` which contains the following sections WITH the section titles listed below. This file is submitted with your code (see Submission instructions below).

- **TITLE:** Include your name, your CSCI 200 section, and a project title.
- **PROBLEM DESCRIPTION:** This is the one-paragraph description of your project (from the Problem Proposal) with any necessary changes or updates.
- **PROGRAM DOCUMENTATION:** This section includes a brief description of how to run your program (i.e., what the user should type and any other information a user might need to know - such as if it needs to be built with SFML) and also a brief description that might be used by another programmer to modify/extend your program. For example, there may be some features that you would have included in your program if you had more time. You could include a list of those features, with any thoughts you had about how they should be implemented. Be sure to mention what, if anything, changed from your original proposal and why those changes were necessary.
- **CLASS DESCRIPTION:** What was your custom class you created? What data and functionality was encapsulated? How did creating the class help the struture of your program?
- **LIST DATA STRUCTURE:** Which list data structure did you choose to use? Why was this structure chosen? How did it apply to the task at hand?
- **FILE I/O:** Where was File I/O incorporated? Why was it necessary to the context of your program?

- **REFLECTIONS:** Include at least a one-paragraph description of what you learned from this project. It might help to think about what problems you encountered, and what you would do differently if you had to do another project.

Resources

While you may search on the Internet for hints as to how certain things are done in the C++ language, you ***cannot*** directly copy and paste code found from resources outside our course. Your project must be representative of your own work, effort, and ideas.

That said, any resources we have used in this class from previous labs and homework assignments are fair game for use in your project.

Incremental Development

Now that you are designing and writing a large project from scratch, the "**Incremental Build**" model of software development is more important than ever before. This is a software development methodology where the model is designed, implemented, and tested incrementally, adding a little more functionality each time, until the product is finished. In other words, write a small amount of code to do one specific task, then run the program to be sure what you have done so far works. Only when you are satisfied with what you have so far do you move on to the next part of the program.

In short, implement and test small parts of your program as you work!

Project Possibilities

There are numerous different project ideas possible. Here are a few examples:

Breakout

In the classic arcade game **Breakout** a layer of bricks lines the top third of the screen. A ball travels across the screen, bouncing off the top and side walls of the screen. When a brick is hit, the ball bounces away and the brick is destroyed. The player loses a turn when the ball touches the bottom of the screen. To prevent this from happening, the player has a movable paddle to bounce the ball upward, keeping it in play.

In this game, you might want to develop the following three classes: `Paddle` , `Brick` and `Ball` .

Frogger

Another classic arcade game, **Frogger** is a game in which the object is to direct frogs to their homes one by one. To do this, each frog must avoid cars while crossing a busy road and navigating a river full of hazards.

Classes such as `Frog` , `Car` , `Truck` , `Log` , `Turtle` , `Crocodile` , and others could be used in implementing this game.

Othello

There are several games with two-dimensional arrays as playing areas. Possibilities include **Connect Four**, **Reversi** (aka, Othello) and **Battleship**.

Non-Games

Finally, there is no requirement that your final project be a game. For example, if you are passionate about bike riding, you might create a program that calculates the optimum front and rear gears that should be selected on a bicycle, given a degree of incline and current velocity. Users select the type of bicycle, specify their speed, pain threshold, and degree of incline. The program then informs the user of the front and rear gears that should be selected.

Or maybe there is something you could write that would be useful for your major or other classes. Anything that meets the requirements of the project (see above) is fair game.

Other Ideas

If you have other ideas but need a bit of help with the design, please feel free to talk to your instructor or a tutor.

Grading Rubric

Your submission will be graded according to the following rubric:

Points	Requirement Description
20	Project behaves as expected.
25	Project makes appropriate use of a class.

15	Project makes appropriate use of a list (array/vector/linked list/stack/queue).
10	Project makes appropriate use of File I/O.
5	Project makes appropriate use of functions, constants, templates, and data types.
5	Project description meets requirements (due Wednesday, October 11, 2023 11:59 PM).
5	`final.txt` meets requirements.
10	**Best Practices** and **Style Guide** followed.
5	Project submitted correctly and builds without errors nor warnings.
100	**Total Points**

Submission

Always, **always**, **ALWAYS** update the header comments at the top of your main.cpp file. And if you ever get stuck, remember that there is LOTS of **help** available.

Zip together your `main.cpp, Makefile, *.hpp, *.h, *.cpp, final.txt, *.*` files and name the zip file `FP.zip`. Upload this zip file to Canvas under Final Project.

This assignment is due by Thursday, December 07, 2023, 11:59 PM.
NO LATE SUBMISSIONS WILL BE ACCEPTED

Last Updated: 09/29/23 09:23

Any questions, comments, corrections, or request for use please contact jpaone {at} mines {dot} edu.

[Jump to Top] [Site Map]