CMSC 113: Computer Science I
Final Project

Proposal due on Gradescope by the beginning of class on Thursday, April 12, 2018 (12 pts.)
Prototype to be checked during lab on April 24/26, 2018 (8 pts.)
Project due on Gradescope by 9:30am, May 9, 2018 (70 pts.)
Presentation to be given during 9:30-12:30, May 9, 2018 (5 pts.)
Paper due on Gradescope by 12pm, May 11, 2018 (5 pts.)

The final project is composed of five parts. First, you will write a detailed project proposal. Then, you will work on your project during class and for homework over the next few weeks. During this time, you will prepare a prototype – a version of your project that is still unfinished. After you finish your project, you will write a short paper and prepare a short presentation discussing its highlights and showing off your work.

This project is really a great opportunity to explore a certain topic that particularly caught your interest or to develop an idea you've had in the back of your head ever since you saw HelloWorld. Be creative; you will do far better in this if you are writing something that really excites you. You may want to choose to write a program relating to the work in some other course; it is a great way to apply what you have learned elsewhere. Or, you may want to write a game, which can be a lot of fun. Or maybe there's a program you wish someone had written, and now you can fulfill that wish.

I strongly encourage you to spend some time thinking about what you will enjoy doing before settling on a project. A little extra time now will help make the whole experience more enjoyable.

A list of possible projects appears at the end of this assignment description.

You are strongly encouraged (but not required) to work with a partner. You and your partner will work together on all parts of the project except for the paper due on the last day of class, which must be individual.

Project Proposal (12 pts.)

Your project proposal will be a detailed description of the final project you will be writing. It must include the following sections:

Program description (1 pt.)

Begin your proposal with a general description of the program you plan on writing and what it will do.

- **Rationale (1 pt.)**

  Include a section stating why you have chosen this particular project. You may wish to mention other projects you have considered, but it is not necessary.

- **Program interface (5 pts.)**

  How will the user experience your program? This section should include a **detailed** description of how the program will appear and how the user will interact with it. Some questions this section will answer include:

  - What will the program do when the mouse clicks on it?
  - Does that depend on where the mouse clicks?
  - What happens when the user presses a key?
  - How does this behavior change throughout the program?
  - What does the program look like? (You may wish to draw a **diagram** or **state machine** to properly illustrate your ideas.)

A user should basically be able to easily use your final program after reading this description. A fellow programmer should also be able to write a very similar program from reading this description. **Detail** is important. Being **specific** is important.

- **Project outline (2 pts.)**

  What are the main problems you must solve? What are the major areas of your program? This section should list classes you will need to write. What are you most worried about in writing this project?

  A good approach to writing this section is to develop the catalog of code elements you will need.

- **Schedule (2 pts.)**

  How do you plan to complete the project in the limited time available? This section should give a specific ordering of tasks and dates when you believe they will be completed. It must also include what specific features you will have ready for the prototype.

  A good approach to writing this section is to develop a series of smaller programs that will lead you to your final goal, as I did for you on the midterm project. One of these programs should be flagged as the prototype.

There is no specific length requirement, but you will likely need at least 2 pages to cover the above topics thoroughly. I expect that all proposals will be well written using proper English (1 pt). Grammatical and spelling mistakes are distracting and will hurt your performance on this part of the project.

It is possible that some of you will be unable to complete the project you propose if it has an inappropriate length or level of difficulty. I will let you know if this is the case by the end of the week that the proposal is due.

Prototype, to be checked in lab (8 pts.)
Project (70 pts.)

All projects must contain an appropriate use of an `ArrayList`. Additionally, you must divide your code into separate classes as appropriate. With a large project, it may be tempting to have some part of your code repeated many times. (For example, if you have 4 enemies, perhaps you might think of writing code for an enemy and just repeating it 4 times.) This is very poor style. Instead of repeating yourself, use a method or class. I encourage you to discuss your design with me first.

You are welcome to submit your code and ask questions during the course of the project. The more specific you make the questions you ask, the more specific (and helpful, usually) I make the answers. Use line numbers to ask about specific pieces of code.

Staying on top of the project and working steadily during the project duration is extremely important. It is foolhardy to leave a large chunk of the project to the last weekend. Do not let this happen.

Presentation, to be given during class (5 pts.)

Your project – and the course overall – will conclude with presentations on the last days the course meets. Your presentation will be 5 minutes long for the project of a student working alone and 8 minutes long for a partnership. The presentation will highlight the different features of the project and will show some of the more interesting and/or intricate pieces of code. You do *not* need to make slides for this presentation; it can be rather informal. Show off your hard work!

Paper (5 pts.)

After you finish the code for your project, you will write a brief (1-2 pages) paper reflecting on the experience of the project as well as the course. **Members of a project partnership will write their papers separately.**

Addressing the project, what was the hardest part of it? What do you wish you had done differently? Would you choose the same project again?

Addressing the course, what are your overall opinions of the course? What could be improved? What should be removed from the course? What should definitely remain? What are your opinions about the various homework assignments and the format of the homework assignments in general? What do you think about the four major projects (Robin, Midterm, Data Visualization, Final)? I value your feedback greatly. Improve the course for future computer science students!

As with the proposal, I expect this paper to be well-written using proper English.

Possible Projects

These projects are merely suggestions to narrow your focus. Your project need not remotely resemble any of these. The number of stars after a project indicates its relative level of difficulty. Five stars is the most difficult; one is the easiest.

- Tetris *****
- Brickles/Arkanoid/Super Breakout (they are all similar) **
- Snood *****
- 3D Pong ****
- SuperPong – Pong including different options, including 2-player & FoozPong *
- Space Invaders **
- Asteroids/Maelstrom ****
- A simple RPG (either visual or text-based) **
- Blackjack or Poker ***
- Dance Dance Revolution **
- Solitaire ***
- A side-view action game, like the original Mario Bros. or Donkey Kong ****
- Sokoban, a box-pushing puzzle game ***
- Frogger ***
- Mini-golf **
- Concentration ***

- A simple accounting program holding multiple accounts with different balances **
- A program to calculate digits of π in different ways & report information about the calculations (rate of convergence, etc.) **
- A basic physics modeler *****
- A chemistry calculator that can name compounds or do stoichiometry, for example **
- A genetics modeler **
- A timeline generator that keeps track of events entered by the user and draws a timeline of them **

- A fancy data visualization, for example like the first one at https://www.nytimes.com/interactive/2017/08/24/us/hurricane-harvey-texas.html ****
- A program to do data analysis on data you've collected (or are working with) in another class (difficulty varies)
- A stock picker, using data to help your user make decisions ***

- A program you might find useful in your life outside of CS 113

- Many, many more…