Assignment Requirements Information

What to hand in
You are required to submit your code electronically. GitHub or other repository URLs, Repl or other online compiler links are not accepted. We accept only file uploads.

1.1 Electronic Submission

Your electronic submission as a tar or zip file must include all of the C++ or Java files that you wrote. Also include any secondary files that are needed to run the program. In your tar or zip file, also include the README file discussed below.

1.1.1. README

Your README file should contain the following information:

Clear instructions as to how to extract your code (if necessary), indicate where the files should be placed (if other than in the immediate directory), how to compile your code (if required to set up any project properties), and run your program. Include any other instructions necessary to execute the program.

Who is the author(s).

Mention any problems with your code (documented problems will result in deduction of fewer marks)

1.1.2. Program Coding and Commenting

Your mark will reflect the correctness of your code as well as its style. We expect you to use meaningful variable names, a consistent indenting scheme, and comments. Besides header comments, you should have normal commenting within methods to document local variables and any computations that aren't obvious. Below are some guidelines as to what should be included in header comments:

- Class headers: This is the header comments at the top of a file containing a class definition. This header should contain the following information:
  - Outline of the purpose of the class
  - Known problems/limitations, if any

- Function headers: This is the header at the beginning of any function definition. These headers should contain the following information:
  - Outline of the purpose of the function
  - Program input, and how the input is used/changed
  - Function return value(s)
  - Known problems/limitations, if any

You are required to use good programming style, including header and other comments, as you go along.

Do not use any OS-specific functions in your code. Programming problems are generic, so make your code generic as well. You must test your program to show that it works correctly.

1.2. If Your Program Doesn't Work

If you get to the deadline and your program still has bugs or doesn't work at all, don't give up. You can still get partial points for correctness, as well as marks for programming style, header comments and test runs. Attach a note to your assignment explaining how far you got, what bugs exist, and any ideas you have about where the problem is. Annotate your test runs with explanations of what the output should have been. If your program doesn't run at all, outline a testing plan: what test inputs you would have used and what the output should have been.