Assignment 4

COMP SCI 2ME3, SFWR ENG 2AA4

March 31, 2021

1 Dates and Deadlines

Assigned: March 16, 2021

Spec and Code: April 12, 2021

Last Revised: March 31, 2021

All submissions are made through git, using your own repo located at:

`https://gitlab.cas.mcmaster.ca/se2aa4_cs2me3_assignments_2021/[macid].git`

where `[macid]` should be replaced with your actual macid. The time for all deadlines is 11:59 pm.

2 Introduction

The purpose of this assignment is to design and specify modules for playing the game 2048. An on-line version of 2048 can be found at:

https://play2048.co/

The modules you are required to specify and implement are the Model and View portions of the Model View Controller design pattern. You do not need to write a controller. Your unit tests will take the role of controller. Bonus marks are available if you choose to specify and implement the entire game, including the Controller, or alternatively combining the Controller with the View for a Model View design pattern.

Your assignment is for the module that stores the state of the game board and the status of the game. You also need a module that can display (view) the state of the game board using text based (ASCII) graphics. Alternatively, for a Bonus grade, you can provide a GUI view of the game board, if you prefer. In addition to the model and view modules, you may also include other modules in your design, as necessary. If you use, and clearly identify, one or more design patterns, the TA may award bonus marks.

Your specification can assume that you have access to a function random that returns a random number between 0 and 1.

Part 1

Step 1

Submit your design specification, written in LaTeX, of the MIS for the game state module and view module. If your specification requires additional modules, you should include their MISes as well. It is up to you to determine your modules interface; that is, you decide on the exported constants, access programs, exceptions etc. You also determine your state variables and specify the semantics for your access program calls. Your design does not need to concern itself with performance. Your specification should be clear, unambiguous, understandable, consistent, complete, validatable and abstract.

At the beginning of your specification document, list the likely changes that you design considers. At this point also give an informal overview of your design. If it is helpful, you can include a diagram of your design, either as a uses relation, or a UML class diagram.

At the end of your specification document you should include an overview/critique of your own design. Please be specific. In particular, you should self-assess how your design performs with respect to all of the following qualities: consistency, essentiality, generality, minimality, cohesion and information hiding. As appropriate, you also should use comments in your MIS to explain your intentions. Your goal is to quickly communicate to the marking TA your design decisions, so that they can make a fair assessment.

At the end of your design specification, provide answers to the following questions:

1. Draw a UML diagram for the modules in A3.

2. Draw a control flow graph for the convex hull algorithm. The graph should follow the approach used by the Ghezzi et al. textbook. In particular, the code statements should be edges of the graph, not nodes. Code for the convex hull algorithm can be found at: https://startupnextdoor.com/computing-convex-hull-in-python/. To match the diagrams available from Ghezzi, replace the for loop in the code with a while loop.

Part 2

Step 2

Submit Java code that matches the specification given in the previous step. You should also submit code that tests your module(s) using JUnit. You do not need to write unit tests for your View module, although you should test it manually. Document your source code using doxygen. You code should include a makefile, with rules `make test` and `make doc`. Performance will not be considered in the grading.

Notes

1. Your git repo is organized with the following directories at the top level: `A1`, `A2`, `A3`, and `A4`. Your specification and code files should be placed in the `A4` folder

2. Please put your name and macid at the top of each of your source files.

3. Your program must work in the ITB labs on mills, as specified for A3.

4. For any figures needed to answer the questions, you can draw them by hand and scan, or you can use software (like draw.io, or Powerpoint, etc) to draw them digitially.

5. As stated above, Bonus grades are available for:

   - Providing a GUI. If you provide a GUI, your program does not have to work on mills. It is assumed that the TAs will run your software on their local machine. You should document for the TAs what libraries are used for the GUI. You should test your software on another machine, or VM, to make sure that you list all required files.
   - Providing a full specification and implementation for the entire game.

6. **Any changes to the assignment specification will be announced in class. It is your responsibility to be aware of these changes. Please monitor all pushes to the course git repo.**