SCRUM Team 5
March 21, 2021
Professor Tang

Design Justification

In order to continue contracting with UVU IED felt that it should change the prototype of the UVSim slightly in order to better align it's program with the values of the company. Because UVSim is programmed with scalability in mind, we were able to keep the majority of our program and expand it to include an integrated GUI. One idea that we kept was making the main memory class a singleton, this ensures that only one instance of memory runs at a time and other methods get a pointer to this instance. There are five members of our group working on UVSim and the singleton made sure that, although we were all working on different parts of the program there was only ever one memory we had to worry about. We also had different classes to handle each of the opcodes that UVSim needed to handle. This worked quite nicely because it allowed us to stay in line with the new specifications given to us by UVU. This became of vital importance when the GUI was implemented because it allowed for the GUI easy access to each of the backend handlers. Because IED had the foresight to program with expansion in mind, IED was able to keep the majority of our program and still fulfill all the requirements given to us by the university.

While we do believe in our overall design, some small changes will have to be made to fit with the gui, and also to keep low coupling. In our original design we had a single invoker class, which held all of the command objects and decided which one to execute based on input, but because that violates the coupling rule, we split our original design into several invokers and different hierarchies of command objects. So we still keep our original command pattern design, and also adhere to the new coupling guidelines.

Other small changes will have to be made to fit with our new GUI. For one we will need a new Facade class that contains classes and methods that will be used by the GUI, this is convenient for the people programming the GUI since they will only have to interact with one class, and it fits perfectly into our established structure. Our interpreter class and executor class will need very small changes in order to function with the GUI. In milestone 1 we had the instantiation of the interpreter essentially run the whole program, and the executor ran all of the commands in one while loop. To make these work with the GUI we have to get rid of the while loops in each and change each so that they have methods that interpret only one line at a time, or execute one line at a time and make the GUI responsible for handling the commands that go inside each. Finally a few of the commands have to be moved to a new hierarchy because they require either user input or output, which need to be handled directly by the GUI and therefore will be added to the facade.

IED programmed with expansion in mind, so we were able to keep the majority of our program while changing small portions to stick to the coupling rule and work with the implementation of the GUI. We are grateful for the opportunity we were given to work closely

with UVU in programming UVSim and have continued to program in line with the expectations provided to us in order to ensure smooth expansion in the future.