 **Time Review**

* We spent over 60 hours working on the project. We started working on the project on September 29th 2014 and we ended on October 24th. We had three meetings as a large group and several other meetings as two separate groups.
* In our group we divided the front end and the back end. Half of us worked on the front end, the other half worked on the back end. The time spent coding could be broken down as follows.
* 10% Coding new features
* 20% Refactoring
* 20% Testing
* 5% Reading
* 20% Designing
* 20% Debugging
* 5% Documenting
* 100% Meeting with Team Members (the one with whom I worked on the Front end)
* 10% Meeting as a full team.
* I personally wrote my code by first writing the name of the classes that we knew from the beginning that our design would utilize. Then we added methods solely by name of the commands we thought we needed to have to have my turtle moving in the scream. Then we added the code that performed the specific the methods said they would. After each method proved to work, I pushed the code online. I pushed my code every time I added new code that did not break the rest of the code.
* The tasks that were the easiest were to add buttons/functionalities to the GUI. It was easy to add the text area, the text boxes, and all the other functionalities to the GUI. However it was much harder to create the GUI itself, coming up with the layout and the style was hard to design and implement. I had a hard time figuring out what type of pane was ideal to add to our code. I had to figure out the advantage and disadvantages of different types of Panes in order to figure out which one suited our design the most.

 **Teamwork**

* We spent around an hour coming up with the design of the large project as a whole. We decided to keep the back end and front end separated, but connected by our controller. In that way we could split and work on our separate sides without any hiccups. We kept to that design all the way to the end, occasionally asking the other group about what they expect to receive from the controller and what they planned on passing to the controller. Then as a subgroup we met almost every other day to work on how to design the layout of our code and implement the features that we planned on having from the beginning.
* Our team consisted of four members. Two of us were responsible for the back end, and the other two were responsible for the front end. On the front end we split up the tasks as we went along. I build the layout of the GUI while my partner build the turtle class. Then I added a toolbar, buttons on the left side, a resource bundle, and the textArea and hbox while my partner coded and research the functionalities for the buttons. On the back end I am not sure as to how they separated the tasks, but the backend as a whole had to parse the string they received then they had to perform the calculation indicated by the command. We collected that information from the front end after them.
* The communication was just right with the outlets of the social network sites which proved useful when we needed to communicate with our teammates. We asked questions directly to our teammates when we needed answer, and we got the answers fairly fast. The questions were never too much.
* The plan was for us to complete both sides of the assignment, back end and front end up two days before the parts of the assignments were due. Then once we met we made sure that the code worked congruently as a whole so that we could make sure that everything worked well. We also decided to limit most of the classes to just one person to start, that way we could make sure we had no merge conflicts and that our code was running well with no problems encountered on its path.
* The plan held up well to the extension. We did not have to rebuild our code in order to accommodate for the extension. We simply needed to add some extra functionalities and modify some others. All in all, our approach accounted for the extra features that we had to implement. We did not arrive at a deadlock because our code was not flexible enough to accommodate some extra features. In the other hand the addition of the extra functionalities was one of the easiest part of the assignment.

 **Commits**

* I pushed code 20 times over the course of the project. The average size of the push were around 100 lines of code. Each push usually were the result of the refined implementation of added methods.
* Yes, I think the messages accurately represent most of my contribution to the project. Some of the messages could have been a little bit more thorough, but for the most part most of the code was written exactly as it should have been.
* “Added saving capabilities” this commit consisted of adding the functionality of saving the users preferences. This commit did not cause any merge conflict because we added a brand new class that took care of saving and reading the files that saved the details regarding the users preferences.
* “Added file menu” this commit consisted of adding a file tab on the toolbar of our GUI which allows to save and write. This commit did cause a merge conflict because the class, in which the functionality was called, was changed. Yes the commit was done at the proper time as an extension to our code at the very end.
* “Added key movement” this code allowed the turtle to be moved using the key commands. No this commit did not have any merge conflicts with the rest of the code. Yes the commit was done in a timely manner relative to the rest of the team it was added when the extension functionalities were being added.

 **Conclusions**

* We underestimated the size of this project. We started with a good foundation and good plan however we did not allocate enough time to go through with our plans. We could estimate better in the future by basing our estimation on part of the code being written before hand and evaluating from there as oppose to before we started coding.
* I took on the responsibilities that I could. I was always working on something when we met as a group. I probably could have worked on more if it came to that but everything was kept within their own boundaries. Yes I kept my team informed about my progress as well as kept track of the status of their progress.
* The GUI’s main renderer class required the most editing. It was the class in which we needed to add all extra functionalities or buttons. Just by the nature of what this class had to do it had to be lengthy and as a result we had plenty to add to it as our functionalities grew and other stuff grew with it.
* To be a better designer I should start planning my code a lot more ahead of time. Too many times we did not know the specifics of a method and we ended up with multiple lines of code that did not do anything productive.
* To be a better teammate I could communicate better with my fellow teammates and not wait for them to ask me for my help but to help whenever I am free to help them out or ready and capable of doing exactly that.
* If I could work on one part it would be on the layout of the GUI. We used a borderPane but that proved to be problematic when we wanted to add new tabs to our design. I would try to implement the tabPane instead and see if it improves the extendibility of our code and if we could add more tabs in a much easier manner that way.