After the first rubric was developed, we split the work into two main sections: the backend, and the front end. The back end is all of the data work, the algorithms, and the OOP. The front end consists mainly of the graphics, and the part the user will see. We have outlined a majority of the classes before the actual programming, but as we get further into the development we add changes and create new structures.

The backend programming is coming along just fine. The first things that were done were the development of the Location and the BoardPiece classes. The Location class is a class that will be used to store the location of objects on our board. It is mainly a class that will just be composed into other object classes. The BoardPiece class is a little more substantial. It has a Location object in it, as well as integer values that will store if it can be interacted with(it is a tree that can be cut down for example) and if the user can walk over it. It also stores if there is a trainer present. The BoardPiece is an abstract class which will have two major subdivisions: EnvironmentPiece (for grass, trees, rocks, ect) and then HumanPiece (for trainers). Those two have not yet been implemented but are in the next coming steps.

Those were the only backend classes described in our rubric for this lab, but I finished them in a reasonable amount of time, and continued to move onto the next steps. The next class that was created was the Pokemon class. This has a laundry list of values that store its attack, defense, HP, name, if its wild, its level and a lot more. Each of its values have getter and setter methods associated with it. We can successful read information off a text file to create a list of Pokemon objects. The initialization of the Pokemon objects is implemented in the Game class which will be the running class for the entire project. Here the Pokemon (and soon all the other data) will be initialized and then be able to interact with each other.

This was all tested in pieces little by little. The location class was first tested by created some dummy objects and printing out all of its values. Once that was successful dummy BoardPieces were created and made sure they were properly holding the Locaton object along with their other data members. Then we moved onto the Pokemon. We created a text file to read in the data from. We tested that it could read in a single Pokemon before adding multiple Pokemon onto the list and confirming that the vector of Pokemon objects in the Game class was being created successfully. Currently we have 6 Pokemon in the game, but to add more all we will need to do is enter the corresponding data into the text file.

On the back end side, we accomplished all the desired tasks set out in our rubric. Once we completed those we continued to go further and started to develop more than we originally had stated that we wanted to do. We deserve the full credit for the rubric because we completed the rubric in full, and then completed multiple “bonus” tasks that were more than the rubric asked.

The next classes and tasks for the next lab include creating the Moves, Types, Sprite, and Item classes, and implement some interactions so that two Pokemon can battle and the user can possible catch a wild Pokemon.

The front end of the programming is reliant on the number of SDL tutorials completed. Thus, the majority of our time this week working on the front end was completing tutorials. At this point, all the tutorials needed for the project have been completed, as far as the basics of the program are concerned. We found Pokemon sprite sheets online and, after completing the sprite sheet tutorial, cut a sprite out of that sheet and displayed it on the screen. We then got the sprite to move when pressing arrow keys. This worked with SDL functions that cut a sprite out of an image and overlaid it on the screen as well as with SDL events to check for arrow key presses. We verified that this worked just by playing around with it and trying to move the sprite out of the screen’s visible region. Additionally, getting the sprite to move was a “bonus task” for us, which is one reason we deserve full credit.