Shanie Hsieh
Shooter.py - a player vs bot shooter game to practice aim.

TESTING AND USAGE

By running “python shooter.py” on the command line immediately starts up the program. The starting screen prompts a “help” or “enter” user input. “Help” pulls up a help menu that contains a bit more information about the game. “Enter” starts the actual game play. The goal is to move your team so they can shoot the enemy team. The “shoot” only works if a member of your team is in the same row or column as the enemy team. “Move” only works to move an agent to any othe nearby squares. Another way to win is by defusing the bomb that is planted. You must move your team without them getting hit to the bomb location. Then, defuse the spike 3 times in order for victory. By following the prompts on the screen, it will tell you exactly what kind of inputs are accepted. In the “agent select” portion of the game, a player can also choose to “surrender” to quickly end the game with defeat.

COMPLETED

The original idea of this project proved to be much more ambitious than can be expected in the 2 week timeframe for this project. However, I have completed the player as a defender idea and allow the player to defuse a planted bomb. I have also completed the basic components of this shooter game; allowing a player to move their agents and shoot the enemy team. The overall template of the game has also been shaped and a rough idea is drawn out.

UNCOMPLETED

There is still a lot of room for improvement in this project. This includes: better graphics, player as attackers, a higher level computer opponent that can move and do more, player versus player UI, as well as leveled-up agents that have abilities such as healing and instant death damage. These are all ideas from the original proposal but would take much more time to implement. The text also seems very crowded and a better UI would be ideal. I would also like to make a gui code to allow easier interaction and visualization for the user.

CHALLENGES

I think the most challenging part of the object-oriented game was the part of starting the code itself. It was difficult to figure out which classes were necessary and which methods were necessary in each class. Making sure the code worked how I wanted it to was also very difficult. It was frustrating to see error message after error message and not know where the root of the problem was. After figuring out main classes to hold attributes, another challenge was figuring out how to get these classes to interact with each other. It was a bit confusing to call things through self and not confuse it with the main class instance. The most difficult part of this specific project was figuring out how to separate the teams, yet have them all part of the same map. Another difficulty was pulling a singular agent and figuring out which team they are on and what to do with them.