**Northeastern University – Silicon Valley**

CS 5150 Game Artificial Intelligence

Homework #6

Sushruth Bhandary

**Note :** I worked on this assignment as a pair along with Michelle Lee and hence both our submissions shall have the same codes for some questions

1. **Markov State Machines: Basic Implementation**

The code for this question can be found in the file 'hw6\_q1.py'. It is a basic implementation of a 2 state FSM. Press 's' on the keyboard when the program is running to start or stop the rat sound and that will cause the transition from one state to another. To run the code, use the command :

> python3 hw6\_q1.py

1. **Fuzzy logic and fuzzy sets**

I used the skfuzzy library in python 2.7 for this question. I used Jupyter Notebook for this question since skfuzzy uses a non-GUI backend for matplotlib for visualization and hence I couldn't get the visualizations working without using Jupiter. The code can be found in the file 'hw6\_q2.ipynb'. I implemented a fuzzy rule set to calculate the winning probability of a character given the character's health and the number of enemies. The code includes two test examples.

1. **Goal oriented behavior (GOAP)**

Implement a basic GOAP algorithm application following the tutorial provided in references below, in the context of the 2D game design you developed above.

1. **Rule-based systems - RETE Algorithm**
2. **Blackboard Architectures**

The code for this question can be found in the file 'bb\_architecture.py'. I implemented a simple game which includes a non playable character, three enemy characters(Red, Blue, Green circles) and an ammo stockpile(Black circle) where the character can refill the ammo. The bottom right corner of the screen is the safe zone. The character has a certain ammo to begin with. The enemies have a health count. One ammo reduces the enemy health by 1. So the character can kill an enemy only if it has at least as much ammo as the enemy character's health.

The code uses a blackboard architecture to make decisions in runtime. The Balckboard class has 2 important variables called 'mode' and 'executing'. Mode shows the action that the character is performing. The modes can be 'attack', 'restock', 'flee' or 'idle'. The 'executing' variable is used to manage access to the blackboard class. The function strategyExpert is used to decide at runtime the next move for the character.

The character first checks if it can defeat any of the enemies and if it can, it attacks the nearest enemy that can be defeated. If none of the enemies can be defeated, it tries to find an ammo stock pile to restock ammo and then checks if it can now defeat any of the enemies. If there are still no enemies that can be defeated, it'll flee to the safe zone. The health, character mode and ammo values are displayed on the game screen.

To run the program, use the instruction :

> python3 bb\_architecture.py