Software Implementation and Testing Document

For

Group 15

Version 1.1

Authors:
Brian F
Cooper P
Chelsea W
Madison D
Richard S

1. Programming Languages (5 points)

List the programming languages use in your project, where you use them (what components of your project) and your reason for choosing them (whatever that may be).

Python, a programming language everyone in the group is familiar with and contains a vast amount of libraries with good readability and documentation. Python also has the Pygame library which we felt would be the best fit for the project.

2. Platforms, APIs, Databases, and other technologies used (5 points)

List all the platforms, APIs, Databases, and any other technologies you use in your project and where you use them (in what components of your project).

Pygame, contains a set of Python modules that provides a good starting point for developing a video game. Used in battling, npc encounter/dialog, exploring, menu screen, game startup and runtime loop components.

SQLite, in Python to create a database to store data on Pokemon entities, shop items, NPC items, items, and pokeballs.

3. Execution-based Functional Testing (10 points)

*Describe how/if you performed functional testing for your project (i.e., tested for the **functional requirements** listed in your RD).*

To test functional requirements, we ran the program and tried to trigger events that would test our implementations. For example, when testing if movement worked, we ran the program and used our arrow keys on the keyboard to see if the player model would move around the map. Another example would be for the functional requirement that an encounter should start when going into line-of-sight of a trainer, we move the player sprite into the line-of-sight of a trainer and see if dialog is triggered.

4. Execution-based Non-Functional Testing (10 points)

*Describe how/if you performed non-functional testing for your project (i.e., tested for the **non-functional requirements** listed in your RD).*

To test non-functional requirements, we would use the pygame.time.Clock().get_fps() method and print the value when running the game to determine if the FPS is 60.

To test how long it takes user input and a game section to load by storing the value provided by pygame.time.get_ticks() called right before a potential user input or load game section in a variable which will act as the start time. Then after the user input or load the game section get the difference by calling pygame.time.get_ticks() at this point and the start time to estimate how long it takes for something to be processed or loaded.

5. Non-Execution-based Testing (10 points)

Describe how/if you performed non-execution-based testing (such as code reviews/inspections/walkthroughs).

To do non-execution-based testing we would meet every other week and go through everyone's contributions to the code, discuss what changes have been made, what functionalities can be combined together, and how the code is working. We would also discuss our plans for our code for upcoming weeks and discuss any problems we would maybe have to encounter during the process. To perform non-execution-based testing, each team member will be assigned code that they were not involved in developing and review that code, noting down any mistakes.