Attribution

Method/Function	Primary Author	Techniques Demonstrated
__init__	Noah Kandel	Optional parameters
__eq__	Ross Zaslavsky	Magic methods other than __init__
__repr__	Ross Zaslavsky	Magic methods other than __init__
__str__	Ross Zaslavsky	Magic methods other than __init__
__contains__	Ross Zaslavsky	Magic methods other than __init__
welcome	Ryan Borak	F-strings containing expressions
load_fighters	Noah Kandel	With statements
load_fighters	Noah Kandel	Sequence unpacking
select_fighter	Ross Zaslavsky	Set operation
player_move	Ross Zaslavsky	Regular expressions
determine_winner	Ryan Borak	Conditional expressions
play_again	Ryan Borak	Conditional expressions
play_game	Ryan Borak	F-strings containing expressions
main	Ross Zaslavsky	Regular expressions

Repository Files

Sample_game.py: sample_game.py implements a turn-based fighter game called "Showup, Showout, Showdown" where the user fights against a CPU fighter. The game has two players: the user and the computer. Each player takes turns to choose a move, which can either be "heal" or "attack," with a 10% chance of landing a critical. Fighters have different heal and attack power, and the game ends when a player's health points reach zero.

The code consists of two classes called "ShowupShowoutShowdown" and "QuitInCase". The first class has several methods that initialize attributes, load fighters' data, allow players to select their fighters, implement the game rules, and display the results. It also includes some magic methods (e.g., eq, str) that enable object comparison and string representation. The "QuitInCase" class creates an exception that allows the user to back out of the game.

The code also includes one function: "quitinput". The "quitinput" function takes a prompt as input and allows the user to quit the game by typing "quit."

Fighters.txt: The fighters.txt file contains the data for different fighters used in the game Showup, Showout, Showdown. Each line in the file represents a single fighter. fighter_name is the name of the fighter, attack_power is an integer representing the fighter's attack power, and heal_power is an integer representing the fighter's healing power.

The load_fighters method in the ShowupShowoutShowdown class reads the data from the fighters.txt file, creates a dictionary where each fighter's name is the key and its attack and heal power are stored as a tuple of integers, and then stores this dictionary in the fighters attribute of the class instance. This allows the game to retrieve the stats for each fighter by name when a player selects a fighter for battle.

Running the Program from the Command Line

To run the program, input python3 (or just python) sample_game.py. Since the fighters.txt file is hardcoded into the program, it will not have to be included in the command line when typed.

How to Use:

To use our program upon running it, the program will initially ask for your name. It isn't case sensitive, but it will only allow you to input letters, so numbers and spaces would yield an invalid name. After entering your name, the rules to Showup, Showout, Showdown will print to the console, as well as the eight different fighters that you can choose from, along with their attack and heal powers. In order to select a fighter, you need to type the fighter's name exactly as it reads on the character selection, making it case sensitive. For example, if you wanted to play as Walter White, you would type his name with both Ws capitalized, or if you were playing as Harley Quinn, the H and Q would be capitalized. Once you and the computer have selected fighters, the game will immediately begin. The console will read two moves that you can choose from, attack or heal. In order to choose from one of the two, you can type heal or attack case insensitively. The console will keep track of the remaining health of both the user and computer player. Once either the user or the computer wins the game, the program will ask if you desire to play again. Here, you can type either yes or no case insensitively. If you type no, the program will end and read a farewell message. If you type yes, the program will bring you back to the

fighter selection screen. At any point while the program is running, if you type quit, the program will end and read a different farewell message.