Programming Project __

This assignment is worth 60 points (6% of the course grade) and must be **completed and turned in before 11:59 on ___, ___ ___, 2019.**

Assignment Overview

This assignment will give you more experience creating and using Python classes. We will create our own pokemon and move classes in the pokmeon.py file. Then we will utilize those classes to have pokemon battles in the project.py file.

Assignment Background

Pokémon is a very successful franchise not only in Japan and the United States, but worldwide. https://en.wikipedia.org/wiki/Pokémon. Some of you may be familiar with the franchise so we hope to increase your interest in programming by coding something similar. And for those of you unfamiliar, we hope this may generate more interest in the types of things you want to create with code.

CSV files for the pokemon and moves are received from the following sources:

- https://github.com/veekun/pokedex/tree/master/pokedex/data/csv
    - from veekun/pokedex GitHub Repository

- https://gist.github.com/armgilles/194bcff35001e7eb53a2a8b441e8b2c6
    - from armgilles GitHub Gist

Project Specifications

Pokemon.py

This file contains two classes: Move and Pokemon. It also contains three dictionaries that are used when calculating damage of moves. These dictionaries are already completed and should not be changed in any way. It is your job to fill out the methods in the Move and Pokemon classes. The initialization methods are also given.

Move class

Attributes:

- **Name**: the name of the move
- **Element**: the move's type
- **Power**: the base damage that the move inflicts
- **Accuracy**: how likely the move is to hit an opponent
- **Attack_Type**: classification if this is a physical attack, special attack or status move

Methods:

1. **__init__ (self, name, element, power, pp, accuracy, attack_type)**
   This method initializes attributes of the Move object. (already given)

2. **__str__ (self)**
   Returns just the name of the move (used for printing).

3. **__repr__ (self)**
   Returns just the name of the move (can utilize the __str__() method here).

4. **get_name (self)**
   Returns the name attribute.

5. **get_element (self)**
   Returns the element attribute.

6. **get_power (self)**
   Returns the power attribute.

7. **get_accuracy (self)**
   Returns the accuracy attribute.

8. **get_attack_type (self)**
   Returns the attack_type attribute.

Pokemon class
Attributes:

- **Name**: the name of the pokemon
- **Element1**: the first element of the pokemon
- **Element2**: the second element of the pokemon (None if the pokemon has one element)
- **HP**: the health points of the pokemon, when this reaches 0 the pokemon faints
- **Patt**: the physical strength of the pokemon
- **Pdef**: the physical defense of the pokemon
- **Satt**: the special strength of the pokemon
- **Sdef**: the special defense of the pokemon
- **Moves**: the list of moves that this pokemon has access to

Methods:

1. **__init__ (self, name, element1, element2, moves, hp, patt, pdef, satt, sdef)**
   This method initializes attributes of the Pokemon object. (already given)

2. **__str__ (self)**
   Returns a string containing the parts of the pokemon object divided into three lines. The first line will display in this order: name, hp, patt, pdef, satt and sdef, followed by a

newline character directly after sdef with no space inbetween sdef and the newline character. The second line will display the element1 and element2 with a newline character after element2. The third line will display all the moves of that pokemon.

Each attribute will take up exactly 15 spaces and be left adjusted no matter the type.

3. **__repr__ (self)**
   Utilize the __str__() method to return the same value.

4. **get_name (self)**
   Returns the name attribute.

5. **get_element1 (self)**
   Returns the element1 attribute.

6. **get_element2 (self)**
   Returns the element2 attribute.

7. **get_hp (self)**
   Returns the hp attribute.

8. **get_patt (self)**
   Returns the patt attribute.

9. **get_pdef (self)**
   Returns the pdef attribute.

10. **get_satt (self)**
    Returns the satt attribute.

11. **get_sdef (self)**
    Returns the sdef attribute.

12. **get_moves (self)**
    Returns the moves attribute.

13. **get_number_moves (self)**
    Returns the number of moves.

14. **choose (self, index)**
    Takes an index and returns the corresponding move from the moves list. If there is an IndexError returns None.

15. **show_move_elements(self)**

Displays the elements of the pokemon's moves

16. show_move_power(self)
Displays the power of the pokemon's moves

17. show_move_accuracy(self)
Displays the accuracy of the pokemon's moves.

18. add_move (self, move)
Adds the move parameter to the list of moves for this pokemon if this pokemon has three or less moves.

19. attack (self, move, opponent)
This method takes the move used by the attacker (self) and deals damage to the opponent (who should also be an instance of class Pokemon). The equation for calculating damage will be a slight variation of the actual calculator because Pokemon levels are not taken into account. It is as follows:

Damage = $\left[\frac{\left[mp*\left(\frac{A}{D}\right)*20\right]}{50}+2\right]*modifier$

mp: the power of the move
A: The patt or satt of the attacking Pokemon
D: The pdef or sdef of the defending Pokemon
modifier: takes into effect same-type attack bonus (STAB) if move if super effective/not effective and critical hits

First the power of the move is extracted using the get_power() method from the move object. Then A and D are determined by obtaining the attack_type from the move using the get_attack_type() method. If the attack_type is equal to 2 or '2' (depending on how you process the file) then A is the attacker's patt attribute and D is the defender's pdef attribute. If the attack_type is equal to 3 or '3' then A is the attacker's satt attribute and D is the defender's sdef.

Next an accuracy value is created using randint from the random module. The accuracy value will be a random integer inbetween 1 to 100 and will be compared to the accuracy attribute of the move object (using the get_accuracy() method). If this randomly created accuracy value is greater than the accuracy attribute of the move, then the move misses (a message is printed) and no damage is dealt.

Next the modifier is calculated. The modifier is first initialized to 1.0. The modifier doubles if the element of the opponent is weak to the element of the attacking move. The modifier is halved if the element of the opponent is strong against the element of the attacking move. And no damage is done to the opponent if the element of the move has no effect on the element of opponent.

Remember to take into account if an opponent has two elements.

- If the element of the move is super effective on both of the opponent's elements, then the modifier should be multiplied by four not two.
- If the element of the move is not very effective on both of the opponent's elements, then the modifier should be divided by four not two.
- If the element of the move is super effective on one of the opponent's elements and not very effective on the other element, then the move does normal damage and a super effective/not very effective message is not printed.
- If the element of the move has no effect on one of the elements, then no damage is done despite the relationship between the move's element and the opponent's other element.

If the move is super effective/ not very effective/ has no effect, then a **single** message should be printed to display that.

Finally, the same-attack type bonus (STAB) gives a boost in damage if the element of the move is the same as one of the elements from the attacking Pokemon. If this is true, then the modifier is multiplied by 1.5 before being multiplied to the rest of the damage.

The damage should be of type int before being passed as a parameter to the opponent's subtract_hp() method.

20. **subtract_hp (self, damage)**
    This method takes the damage variable and subtracts it from the hp of the Pokemon object. The hp of the Pokemon object becomes the maximum of the hp - damage or 0.

Project.py
This contains the main and other functions that are to be filled out.

1. **read_file_moves(fp)**
   This function takes in the file pointer created from opening the moves.csv file and returns a list of move objects where the generation_id column equals '1'.
   In this function you should iterate through each line of the file creating a move object with the contents of the line and adding that move object to the list of moves.

   The damage_classification_id has three values, '1', '2' and '3', where '2' denotes a physical attack, '3' denotes a special attack and '1' denotes a stat changing or status effect move. If the damage_classification_id is ‘1’ then that move will not be added to the move list.

Some moves do not have a value for power listed and others do not have a value for accuracy listed. Moves like this should not be added to the move list.

Remember to change the power and accuracy attributes to integers before using them to create the move instance. Also, for elements this file gives only integers. Remember to use the actual element when creating the move object (utilize the given element_id_list here).

2. **read_file_pokemon(fp)**
   This function takes in the file pointer created from opening the pokemon.csv file and returns a list of pokemon objects where the Generation column equals ‘1’. In this function you should iterate through each line of the file creating a pokemon object with the contents of the line and adding that pokemon object to the list of pokemon.

   Remember to change the hp, attack, defense, sattack and sdefense attributes to integers before using them to create the pokemon instance.

3. **choose_pokemon (choice, pokemon_list)**
   This function takes in user input (called choice) and the list of available pokemon. If the user input is an integer the integer becomes the index for selecting a pokemon from the pokemon_list and **a deepcopy** of the pokemon object at that index is returned. If the input is a string, then that string is compared to the name attribute of the pokemon in the list and if a name matches then **a deepcopy** of the pokemon object with that name is returned. If the index is out of range or the string is not found, then None is returned.

4. **add_moves (pokemon, moves_list)**
   This function first adds one random move to the pokemon’s move list, then adds three more moves that match one of the elements of the pokemon. Each move is to be added using the pokemons object’s add_move() method.

5. **main()**
   The main function. This function simulates a 1 on 1 pokemon battle between Player 1 and Player 2. First the relevant moves and pokemon should be added to respective lists using the read_file functions. Afterward each player picks a pokemon. Four moves are added to each pokemon using the add_moves() function. Finally, the opponents battle until one of the player’s pokemon faints or one player quits with the ‘Q’ or ‘q’ command. After a battle ends the user is prompted if they would like to have another. If the user responds with ‘Y’ or ‘y’, then a new battle ensues. If the user responds with ‘N’, ‘n’, ‘Q’, or ‘q’ then the program finishes execution. All other inputs are invalid.

Assignment Deliverables

The deliverables for this assignment are the following two files:

project.py – the source code for your Python program
pokemon.py – the source code for the Pokemon and Move classes

Assignment Notes
<u>Different from real Pokemon</u>:

- For those of you knowledgeable about Pokemon you will notice a few things missing. The speed stat is not utilized.
- Moves that take more than one turn to perform, have multiple hits and have cooldowns are simplified to single hit, single turn moves.
- Status effect moves, moves that lower stats, moves that change terrain and recovery/protection moves are also not utilized.
- Finally, there are no critical hits or levels to take into consideration.

All of these factors are to contribute to the simplicity of the project as the focus here is on classes.

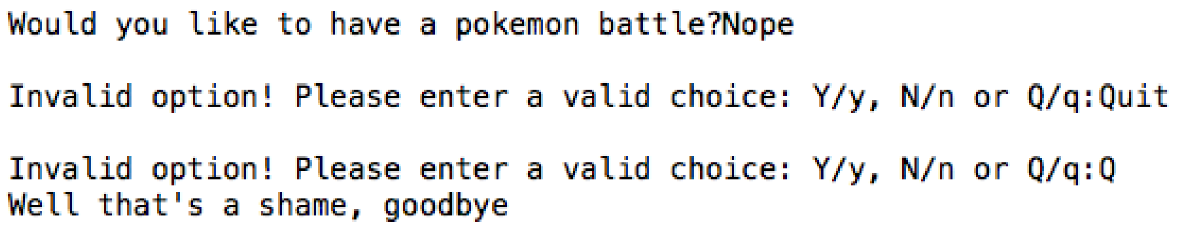
<u>Avoid two instances of the same ID</u>:
When working with the read_file_pokemon() method, be wary of pokemon with mega forms. Their ID will appear more than once in the file; however, we only want the first instance of the pokemon. For example, ID 3 contains the pokemon name "Venusaur", another instance of ID 3 also contains the name "VenusaurMega Venusaur". We only want the pokemon that appears with the first instance of the ID. One way to work with this is to utilize a set of IDs.

<u>First generation only</u>:
The element and pokemon lists are created from the first generation only for this project.

<u>Remember to use deepcopy</u>:
When selecting a pokemon from the pokemon list, use deepcopy. If you assign a variable equal to one of the pokemon, then that variable is a reference to the pokemon from the list and not a copy. Think of it like this. Let's say we both choose the pokemon Pikachu. If we do not use deepcopy, then my Pikachu and your Pikachu are the same Pikachu. If we do use deepcopy then my Pikachu is different from your Pikachu and their moves can be different.

Test Cases

Test 1

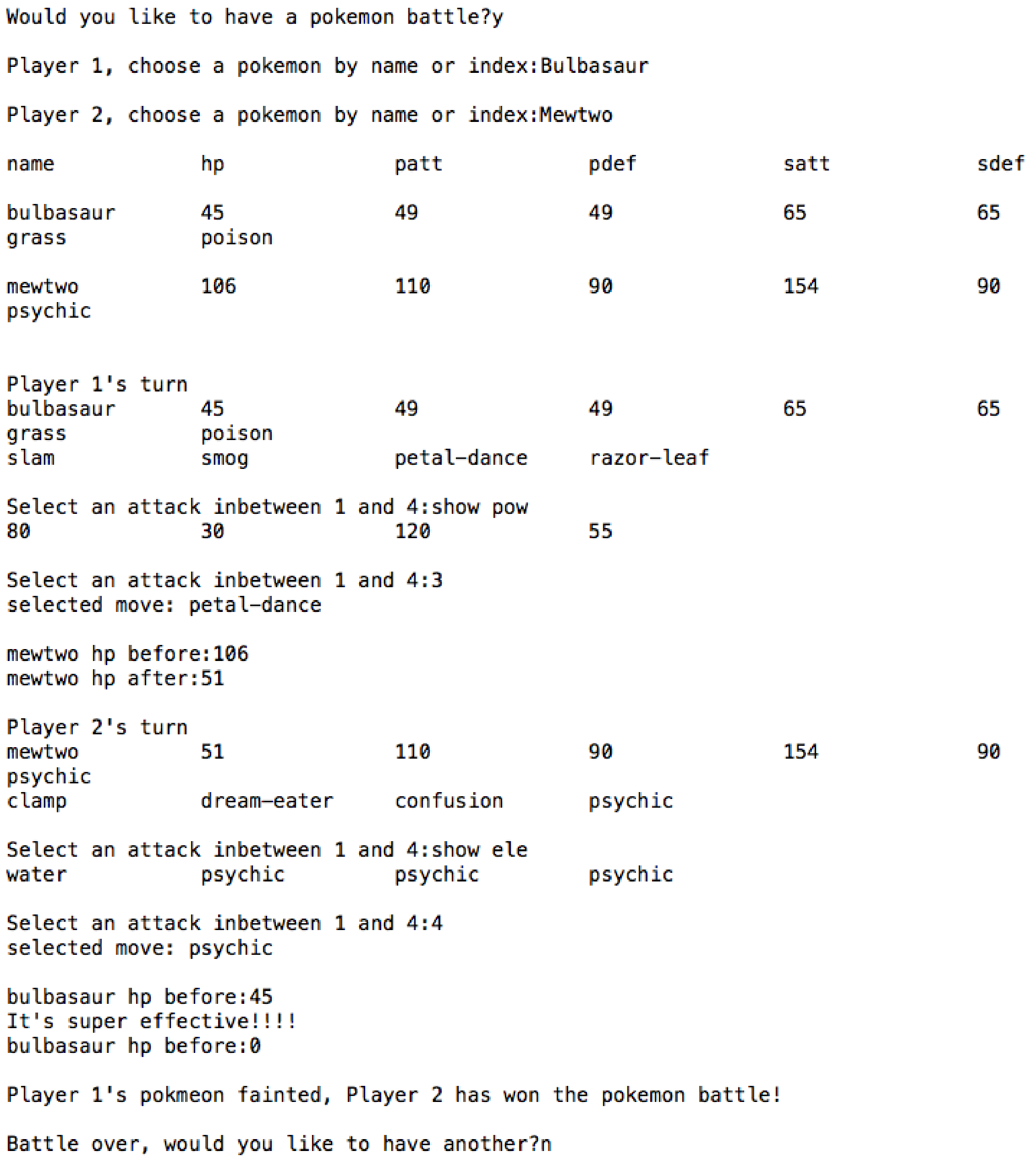


```
Would you like to have a pokemon battle?Nope

Invalid option! Please enter a valid choice: Y/y, N/n or Q/q:Quit

Invalid option! Please enter a valid choice: Y/y, N/n or Q/q:Q
Well that's a shame, goodbye
```

Test 2



```
Would you like to have a pokemon battle?y

Player 1, choose a pokemon by name or index:Bulbasaur

Player 2, choose a pokemon by name or index:Mewtwo

name            hp              patt            pdef            satt            sdef

bulbasaur       45              49              49              65              65
grass           poison

mewtwo          106             110             90              154             90
psychic


Player 1's turn
bulbasaur       45              49              49              65              65
grass           poison
slam            smog            petal-dance     razor-leaf

Select an attack inbetween 1 and 4:show pow
80              30              120             55

Select an attack inbetween 1 and 4:3
selected move: petal-dance

mewtwo hp before:106
mewtwo hp after:51

Player 2's turn
mewtwo          51              110             90              154             90
psychic
clamp           dream-eater     confusion       psychic

Select an attack inbetween 1 and 4:show ele
water           psychic         psychic         psychic

Select an attack inbetween 1 and 4:4
selected move: psychic

bulbasaur hp before:45
It's super effective!!!!
bulbasaur hp before:0

Player 1's pokmeon fainted, Player 2 has won the pokemon battle!

Battle over, would you like to have another?n
```

Test 3

```
Would you like to have a pokemon battle?Y

Player 1, choose a pokemon by name or index:charmander

Player 2, choose a pokemon by name or index:4

name            hp              patt            pdef            satt            sdef

charmander      39              52              43              60              50
fire

charmander      39              52              43              60              50
fire


Player 1's turn
charmander      39              52              43              60              50
fire
slam            fire-blast      fire-spin       ember

Select an attack inbetween 1 and 4:4
selected move: ember

charmander hp before:39
Not very effective...
charmander hp after:24

Player 2's turn
charmander      24              52              43              60              50
fire
bite            fire-blast      flamethrower    fire-spin

Select an attack inbetween 1 and 4:2
selected move: fire-blast

charmander hp before:39
Not very effective...
charmander hp before:0

Player 1's pokmeon fainted, Player 2 has won the pokemon battle!

Battle over, would you like to have another?Y
```

```
Player 1, choose a pokemon by name or index:Machamp

Player 2, choose a pokemon by name or index:Golem

name            hp              patt            pdef            satt            sdef

machamp         90              130             80              65              85
fighting

golem           80              120             130             55              65
rock            ground


Player 1's turn
machamp         90              130             80              65              85
fighting
egg-bomb        high-jump-kick  jump-kick       rolling-kick

Select an attack inbetween 1 and 4:show ele
normal          fighting        fighting        fighting

Select an attack inbetween 1 and 4:show acc
75              90              95              85

Select an attack inbetween 1 and 4:show pow
100             130             100             60

Select an attack inbetween 1 and 4:2
selected move: high-jump-kick

golem hp before:80
It's super effective!!!!
golem hp after:0

Player 2's pokmeon fainted, Player 1 has won the pokemon battle!

Battle over, would you like to have another?N
```

Test 4

```
Would you like to have a pokemon battle?y

Player 1, choose a pokemon by name or index:pikachu

Player 2, choose a pokemon by name or index:18

name            hp              patt            pdef            satt            sdef

pikachu         35              55              40              50              50
electric

pidgeot         83              80              75              70              70
normal          flying


Player 1's turn
pikachu         35              55              40              50              50
electric
slam            thunder-punch   thunderbolt     thunder

Select an attack inbetween 1 and 4:3
selected move: thunderbolt

pidgeot hp before:83
It's super effective!!!!
pidgeot hp after:0

Player 2's pokmeon fainted, Player 1 has won the pokemon battle!

Battle over, would you like to have another?y
```

```
Player 1, choose a pokemon by name or index:Rhydon

Player 2, choose a pokemon by name or index:18

name            hp              patt            pdef            satt            sdef

rhydon          105             130             120             45              45
ground          rock

pidgeot         83              80              75              70              70
normal          flying


Player 1's turn
rhydon          105             130             120             45              45
ground          rock
bubble          rock-throw      bone-club       dig

Select an attack inbetween 1 and 4:show ele
water           rock            ground          ground

Select an attack inbetween 1 and 4:3
selected move: bone-club

pidgeot hp before:83
No effect!
pidgeot hp after:83

Player 2's turn
pidgeot         83              80              75              70              70
normal          flying
wrap            horn-attack     self-destruct   egg-bomb

Select an attack inbetween 1 and 4:show ele
normal          normal          normal          normal

Select an attack inbetween 1 and 4:4
selected move: egg-bomb

rhydon hp before:105
Not very effective...
rhydon hp before:84

P1 hp after: 84
P2 hp after: 83
```

```
Player 1's turn
rhydon          84              130             120             45              45
ground          rock
bubble          rock-throw      bone-club       dig

Select an attack inbetween 1 and 4:4
selected move: dig

pidgeot hp before:83
No effect!
pidgeot hp after:83

Player 2's turn
pidgeot         83              80              75              70              70
normal          flying
wrap            horn-attack     self-destruct   egg-bomb

Select an attack inbetween 1 and 4:1
selected move: wrap

rhydon hp before:84
Not very effective...
rhydon hp before:80

P1 hp after: 80
P2 hp after: 83

Player 1's turn
rhydon          80              130             120             45              45
ground          rock
bubble          rock-throw      bone-club       dig

Select an attack inbetween 1 and 4:q
Player 1 quits, Player 2 has won the pokemon battle!

Battle over, would you like to have another?n
```

Grading Rubric

Computer Project #_ Scoring Summary

General Requirements:

(5 pts)	Coding Standard 1-9

(descriptive comments, mnemonic identifiers, format, etc…)

Implementations:

(5 pts)	Move class
(10 pts)	Pokemon class (not including attack method)
(15 pts)	attack method (Pokemon class)
(5 pts)	choose_pokemon function
(5 pts)	add_moves function
(15 pts)	main function