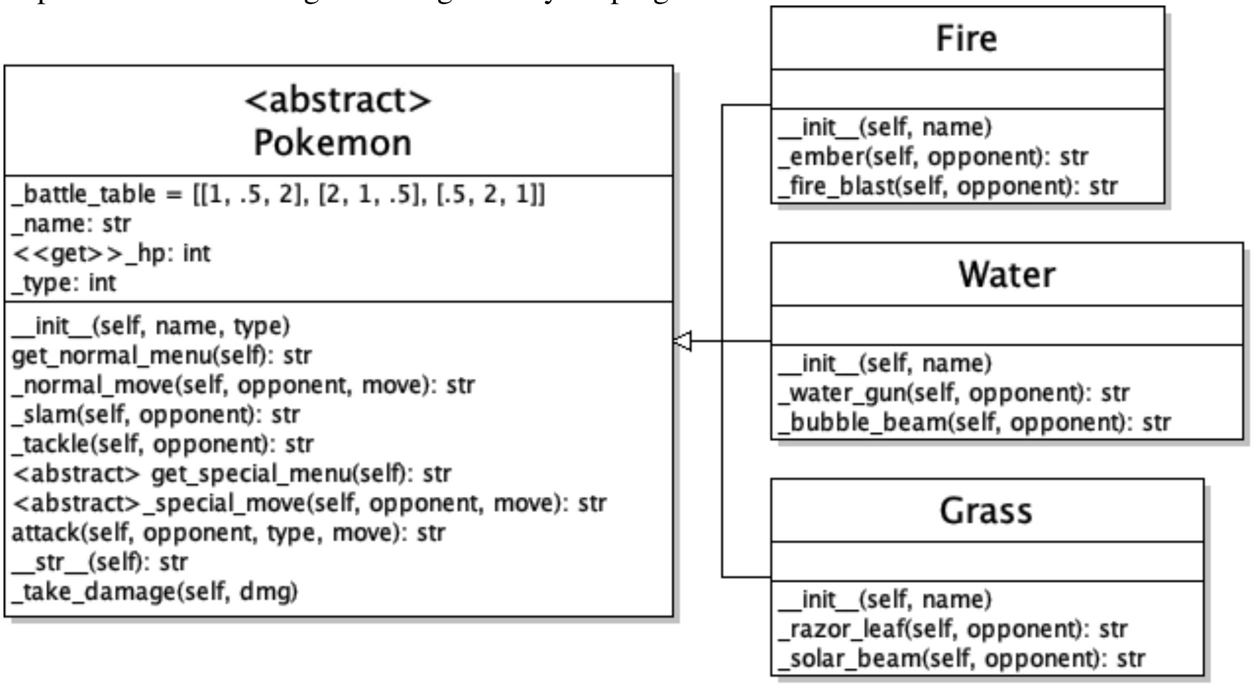
CECS 277 – Lab 8 – Abstract

Pokémon Gym Battle

Create a game where the user must defeat three pokémon to win the game. Use inheritance to implement the following class diagram in your program.


<abstract>
Pokemon
_battle_table = [[1, .5, 2], [2, 1, .5], [.5, 2, 1]]
_name: str
<<get>>_hp: int
_type: int
__init__(self, name, type)
get_normal_menu(self): str
_normal_move(self, opponent, move): str
_slam(self, opponent): str
_tackle(self, opponent): str
<abstract> get_special_menu(self): str
<abstract>_special_move(self, opponent, move): str
attack(self, opponent, type, move): str
__str__(self): str
_take_damage(self, dmg)
Fire
__init__(self, name)
_ember(self, opponent): str
_fire_blast(self, opponent): str
Water
__init__(self, name)
_water_gun(self, opponent): str
_bubble_beam(self, opponent): str
Grass
__init__(self, name)
_razor_leaf(self, opponent): str
_solar_beam(self, opponent): str

<u>Abstract Pokemon Class</u> (pokemon.py) –

1. __init__(self, name, type) – set the _name, and _type using the parameters, assign the _battle_table the 2D list given above, and set _hp to 25.
2. hp property – use a decorator to get (not set) the value of _hp.
3. get_normal_menu(self) – returns a string with the menu options for the normal moves: slam and tackle.
4. _normal_move(self, opponent, move) – use the move parameter to choose to call either slam or tackle method, returns the string returned from those methods.
5. _slam(self, opponent) and _tackle(self, opponent) – randomize some damage (slam 1-8, tackle 3-6), call take_damage on the opponent and return a string description of the move using both pokemons names, the type of move, and the amount of damage taken.
6. get_special_menu(self) – abstract (overridden in the subclasses) - returns the menu for the special moves of each type.
7. _special_move(self, opponent, move) – abstract (overridden in the subclasses) – uses the move parameter to choose to call either of the special moves for that type.
8. attack(self, opponent, type, move) – use the type parameter to choose to call either _normal_move or _special_move.
9. __str__(self) – display the pokemon's name and hp in the format "Name: hp/25".
10. _take_damage(self, dmg) – the damage the pokemon takes. Subtract the dmg value from the pokemon's _hp. Check that the _hp doesn't go past 0 (if it's negative, reset it to 0).

Fire/Water/Grass Classes (fire.py/water.py/grass.py) – each inherit from Pokemon

1. __init__(self, name) – call super to set the name and type (fire = 0, water = 1, grass = 2).
2. Override the get_special_menu and _special_move methods for this class.
3. Create the two move methods – each pokemon type has two moves (listed above in the class diagram). Randomize damage (move1 1-5, move2 3-4), look up the multiplier in the battle table based on the two pokemon's type and calculate the total damage the opposing pokemon will take (make sure it's a whole number, not a decimal). Then return a string description of the move using both pokemons names, the type of move, the amount of damage taken, and whether it was effective (2) or not (.5).

Main (main.py) – Choose a random type of pokemon that the gym leader will have (ie. fire, water, or grass). Create a list of three pokemon objects all of the same type and give them names. Prompt the user to choose a single pokemon of one of the three types. Allow the user to choose what type of attack to do (ie. normal or special), and then which move to do of that type (ie. if the user chose normal, then they can either choose slam or tackle), call the pokemon's attack method and display the resulting string. If the gym leader's pokemon still has hit points, then it will choose a random type and move to attack back with. When one of the gym leader's pokemon is defeated, remove it from the list. Repeat until the user has defeated all three of the gym leader's pokemon, or until the user's pokemon is defeated. Display a message when the user wins/loses. Check all input for validity. Document all classes, methods, and functions.

Example Output (user input is in italics)**:**

```
PROF OAK:  Hello Trainer!  Today
you're off to fight your first gym
battle of 1 vs. 3 GRASS pokemon.
Select the pokemon that you will
fight with.
1. I choose you, Charmander.
2. Squirtle!  GO!
3. We can do it, Bulbasaur!
Please choose a pokemon: 1

  -- GYM BATTLE --
GYM LEADER: I choose you:
Oddish HP: 25/25

Charmander HP: 25/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
2. Fire Blast
Enter move: 1

Charmander engulfs Oddish in EMBERS
for 8 damage.
It was SUPER EFFECTIVE!
Oddish slices Charmander with RAZOR
sharp LEAVES for 2 damage.
It was not very effective.
```

```
GYM LEADER: I choose you:
Oddish HP: 17/25

Charmander HP: 23/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
2. Fire Blast
Enter move: 2

Charmander BLASTS Oddish with FIRE
for 8 damage.
It was SUPER EFFECTIVE!
Oddish TACKLES Charmander for 3
damage.

GYM LEADER: I choose you:
Oddish HP: 9/25

Charmander HP: 20/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
```

```
2. Fire Blast
Enter move: 1

Charmander engulfs Oddish in EMBERS
for 2 damage.
It was SUPER EFFECTIVE!
Oddish blasts Charmander with a
SOLAR BEAM for 2 damage.
It was not very effective.

GYM LEADER: I choose you:
Oddish HP: 7/25

Charmander HP: 18/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 1

Choose a Move:
1. Slam
2. Tackle
Enter move: 1

Charmander SLAMS Oddish for 7
damage.
GYM LEADER: NOOOOO!  You defeated
my pokemon!

GYM LEADER: I choose you:
Bellsprout HP: 25/25

Charmander HP: 18/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
2. Fire Blast
Enter move: 2

Charmander BLASTS Bellsprout with
FIRE for 8 damage.
It was SUPER EFFECTIVE!
Bellsprout blasts Charmander with a
SOLAR BEAM for 1 damage.
It was not very effective.

...
GYM LEADER: I choose you:
Bellsprout HP: 2/25

Charmander HP: 13/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
2. Fire Blast
Enter move: 1

Charmander engulfs Bellsprout in
EMBERS for 10 damage.  It was SUPER
EFFECTIVE!
GYM LEADER: NOOOOO!  You defeated
my pokemon!

GYM LEADER: I choose you:
Exeggcute HP: 25/25

Charmander HP: 13/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
2. Fire Blast
Enter move: 2

Charmander BLASTS Exeggcute with
FIRE for 6 damage.
It was SUPER EFFECTIVE!
Exeggcute TACKLES Charmander for 6
damage.

...
GYM LEADER: I choose you:
Exeggcute HP: 7/25

Charmander HP: 4/25
Choose an Attack Type:
1. Normal
2. Special
Enter attack type: 2

Choose a Move:
1. Ember
2. Fire Blast
Enter move: 2

Charmander BLASTS Exeggcute with
FIRE for 8 damage.
It was SUPER EFFECTIVE!
GYM LEADER: NOOOOO!  You defeated
my pokemon!

YOU WON!  You defeated the gym
leader.
```

Notes:

1. You should have 5 different files: pokemon.py, fire.py, water.py, grass.py, and main.py.
2. Use docstrings to document each of the classes, their attributes, and each of their methods. See the lecture notes for examples.
3. Place your names, date, and a brief description of your program in a comment block at the top of your main file. Place brief comments throughout your code.
4. Do not create any extra attributes, methods, or parameters.
5. Please do not create any global variables or use the attributes globally. Use the methods instead. You can use the attributes in the subclasses when needed. You can access the hp in main using the property. Only use the public methods (non-underscored) in main.
6. Check all user input using the get_int_range function in the check_input module.
7. Feel free to add text to make the game more interesting and to add flair. You may also modify the random amounts of damage to better balance the game.
8. List of pokemon names and types: https://pokemondb.net/pokedex/game/red-blue-yellow.
9. Thoroughly test your program before submitting:
    a. Make sure that your classes are inherited properly. Fire, Water, and Grass should inherit from Pokemon, and Pokemon should be abstract with two abstract methods.
    b. Make sure user input is validated.
    c. Make sure that each type of pokemon does the proper special attacks.
    d. Make sure that the normal attacks do not use the multiplier from the battle table.
    e. Make sure that the special attacks do use the multiplier from the battle table.
    f. Make sure that the correct multiplier is used for the two types of pokemon (ie. Fire is super effective against Grass, Water is super effective against Fire, and Grass is super effective against Water).
    g. Make sure that the damage dealt is correctly subtracted from the opponent.
    h. Make sure that the defeated pokemon are removed from the list.
    i. Make sure the game ends when the user defeats all 3 pokemon, or when the user's pokemon runs out of hp.