Activity No. 2	
Inheritance, Encapsulation, and Abstraction	
Course Code: CPE009	**Program:** BS Computer Engineering
Course Title: Object-Oriented Programming 2	**Date Performed:** 09 / 28 / 2024
Section: CPE21S4	**Date Submitted:** 09 / 29 / 2024
Name(s): ROALLOS, Jean Gabriel Vincent G.	**Instructor:** Prof. Maria Rizette Sayo
Documentation	

```
char1 = Character("Bob")
print(char1._username)
print(char1.getUsername())
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-1-ad88840c42ba> in <cell line: 61>()
     59
     60 char1 = Character("Bob")
---> 61 print(char1._username)
     62 print(char1.getUsername())

AttributeError: 'Character' object has no attribute '_username'
```

```
Char1 = Swordsman("Royce")
Char2 = Magician("Archie")
print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
print(f"{Char2.getUsername()} HP: {Char2.getHp()}")

Char1.slashAttack(Char2)
Char1.basicAttack(Char2)
print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
print(f"{Char2.getUsername()} HP: {Char2.getHp()}")

Char2.heal()
Char2.magicAttack(Char1)
print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
print(f"{Char2.getUsername()} HP: {Char2.getHp()}")
```

```
Royce HP: 110
Archie HP: 105
Royce performed Slash Attack! -10
Royce performed Basic Attack! -5
Royce HP: 110
Archie HP: 90
Archie performed Heal! + 10
Archie performed Magic Attack! -15
Royce HP: 95
Archie HP: 100
```

```
Char2.heal()
Char2.slashAttack(Char1)
print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
print(f"{Char2.getUsername()} HP: {Char2.getHp()}")
```

```
Royce HP: 110
Archie HP: 105
Royce performed Slash Attack! -10
Royce performed Basic Attack! -5
Royce HP: 110
Archie HP: 90
Archie performed Heal! + 10
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-11-12bd18ee7341> in <cell line: 16>()
     14
     15 Char2.heal()
---> 16 Char2.slashAttack(Char1)
     17 print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
     18 print(f"{Char2.getUsername()} HP: {Char2.getHp()}")

AttributeError: 'Magician' object has no attribute 'slashAttack'
```

```
[17] #from Swordsman import Swordsman
     #from Archer import Archer
     #from Magician import Magician
     #from Boss import Boss

     Char1 = Swordsman("Royce")
     Char2 = Boss("Archie")
     print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
     print(f"{Char2.getUsername()} HP: {Char2.getHp()}")

     Char1.slashAttack(Char2)
     Char1.basicAttack(Char2)
     print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
     print(f"{Char2.getUsername()} HP: {Char2.getHp()}")

     Char2.heal()
     Char2.basicAttack(Char1)
     Char2.slashAttack(Char1)
     Char2.rangedAttack(Char1)
     Char2.magicAttack(Char1)
     print(f"{Char1.getUsername()} HP: {Char1.getHp()}")
     print(f"{Char2.getUsername()} HP: {Char2.getHp()}")
```

```
Royce HP: 110
Archie HP: 145
Royce performed Slash Attack! -10
Royce performed Basic Attack! -5
Royce HP: 110
Archie HP: 130
Archie performed Heal! + 5
Archie performed Basic Attack! -5
Archie performed Slash Attack! -15
Archie performed Ranged Attack! -7
Archie performed Magic Attack! -10
Royce HP: 73
Archie HP: 135
```

6. Supplementary Activity

Python

```python
from random import randint
#from Swordsman import Swordsman
#from Archer import Archer
#from Magician import Magician
#from Boss import Boss

class Game:
  def __init__(self):
    self.singleplay_wins = 0
    self.pvp_wins = 0
    self.roles = ["Swordsman", "Archer", "Magician"]

  def select_mode(self):
    mode = input("Select Game Mode: [1] Single Player, [2] PvP: ")
    if mode == "1":
      self.singleplay()
    elif mode == "2":
      self.pvp()
    else:
      print("Invalid Input")
      self.select_mode()

  def select_role(self, role):
    if role == "1":
      player = Swordsman(input(str("Enter player name: ")))
    elif role == "2":
      player = Archer(input(str("Enter player name: ")))
    elif role == "3":
      player = Magician(input(str("Enter player name: ")))
    else:
      print("Invalid Input. Selecting Novice...")
      player = Novice(input(str("Enter player name: ")))

    return player

  def start_game(self):
    print("Welcome to OOP RPG!")
    self.select_mode()

  def singleplay(self):
    player = Novice(input(str("Enter player name: ")))
    opponent = Boss("Monster")

    if self.singleplay_wins >= 2:
      print(f"Congratulations, {player.getUsername()}! Pick a class!\n")
      print("[1] Swordsman, [2] Archer, [3] Magician\n")
      role = input("Select Role: ")
      self.select_role(role)

    self.singleplay_session(player, opponent)11

    #print(f"Welcome {player.getUsername()}!")
```

```
  def pvp(self):
    pass

  def sp_player(self, player, opponent, player_choice):
    if player_choice == "1":
      player.basicAttack(opponent)

    else:
      print(f"{player.getUsername()} flinched!")

  def opponent_computer(self, player, opponent, opponent_choice):
    if opponent_choice == 0:
      opponent.basicAttack(player)

    elif opponent_choice == 1:
      opponent.slashAttack(player)

    elif opponent_choice == 2:
      opponent.rangedAttack(player)

    elif opponent_choice == 3:
      opponent.magicAttack(player)

    else:
      print("Opponent flinched!")

  def singleplay_session(self, player, opponent):
    print("\nWelcome to the battle!\n")

    while player.getHp() > 0 and opponent.getHp() > 0:
      print(f"{player.getUsername()} HP: {player.getHp()}")
      print(f"{opponent.getUsername()} HP: {opponent.getHp()}")

      player_choice = input("What to do? [1] Attack: ")
      print()
      self.sp_player(player, opponent, player_choice)

      opponent_choice = randint(0, 25)
      self.opponent_computer(player, opponent, opponent_choice)
      print()

    if player.getHp() <= 0:
      print(f"{opponent.getUsername()} won!")

    else:
      print(f"{player.getUsername()} won!")
      self.singleplay_wins += 1
      print(f"Total Singleplayer Wins: {self.singleplay_wins}")
      print("Returning to menu..\n\n")
      self.start_game()

if __name__ == "__main__":
  game = Game()
  game.start_game()
```

Questions

1. Why is Inheritance important?

   Inheritance of attributes for different classes can be beneficial for avoiding code redundancy.

2. Explain the advantages and disadvantages of using/applying inheritance in an Object-Oriented Program.

   It can be used to show relationships between objects. In a more practical sense, It shortens code and makes multiple lines of code reusable for different instances. A downside in using inheritances is that a slight modification in the parent classes would create cascading effects to its child classes.

3. Differentiate single inheritance, multiple inheritance, and multi-level inheritance.

   Single inheritance is a type of inheritance where a child class inherits attributes and behavior from a parent class. A function with multiple inheritance, on the other hand, has two or more parent classes from which it inherits its attributes. Multi-level inheritance happens when a parent class of a child class has another class that it inherits its attributes from.

4. Why is *super().__init__(username)* added in the codes of Swordsman, Archer, Magician, and Boss?

   It is a reinitialization of the username variable from their parent class/superclass Character. This gives the child classes Swordsman, Archer, Magician, and Boss access to the username entered into the Character class.

5. How do you think Encapsulation and Abstraction helps in making good Object-Oriented Programs?

   Encapsulation helps with grouping certain attributes of an object and processes concerned with it. Abstraction, on the other hand, helps with hiding and securing data values through private variables.

7. Conclusion

This laboratory activity helped me deepen my understanding of class inheritance and the different types of inheritances. In the supplementary activity, I think I have implemented all possible attributes for the single player mode, I have struggled to build code for the PVP mode for the multiplayer part of the game.

8. Assessment Rubric