Started on	Friday, 30 May 2025, 11:27 AM
State	Finished
Completed on	Friday, 30 May 2025, 8:03 PM
Time taken	8 hours 35 mins
Overdue	6 hours 35 mins
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a Program in Python to Generate Fibonacci series for the number '5'

For example:

Input	Result
---	0 1 1 2 3

Answer: (penalty regime: 0 %)

```
a=0
b=1
c=0
print(a)
print(b)
for i in range(2,5):
    c=a+b
    print(c)
    a=b
    b=c
```

	Input	Expected	Got	

✔	`---`	`0`
		`1`
		`1`
		`2`
		`3`
			`0`
			`1`
			`1`
			`2`
			`3`
				✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Write a Python program for simply using the overloading operator for adding two objects.

class name : fruits

object name : apple, mango, a and b

For example:

Input	Result
100 200 banana orange	apple and mango mixed: 300 fruit mix: bananaorange

Answer: (penalty regime: 0 %)

```
        self.item = item

    def __add__(self, other):

        total_price = self.price + other.price
        combined_items = self.item + other.item
        return Accessories(total_price, combined_items)

    def display_result(self):
        print(f"apple and mango mixed: {self.price}")
        print(f"fruit mix:  {self.item}")

# Example usage:
a=int(input())
b=int(input())
c=input()
d=input()
obj1 = Accessories( a,c)
obj2 = Accessories(b,d)

result_obj = obj1 + obj2
result_obj.display_result()
```

	Input	Expected	Got	
✔	100 200 banana orange	apple and mango mixed: 300 fruit mix: bananaorange	apple and mango mixed: 300 fruit mix: bananaorange	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Create a parent class `Fish` and define a class method `type`, then create a child class called `Shark` while overriding the `type` method so that objects instantiated from the `Shark` class use the overridden method.

For example:

Result
fish shark

Answer: (penalty regime: 0 %)

Reset answer

```
class Fish:
    def diet(self):
        print("fish")

class Shark:
    def diet(self):
        print("shark")

obj_fish=Fish()

obj_shark=Shark()
obj_fish.diet()
obj_shark.diet()
```

	Expected	Got	
✔	`fish` `shark`	`fish` `shark`	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Correct

Mark 20.00 out of 20.00

Create an abstract base class has a concrete method sleep() that will be the same for all the child classes. So, we do not define it as an abstract method, thus saving us from code repetition. On the other hand, the sounds that animals make are all different. For that purpose, define the sound() method as an abstract method. then implement it in all child classes.

For example:

Result
I am going to sleep in a while I can meow I can hiss

Answer: (penalty regime: 0 %)

Reset answer

```
from abc import ABC,abstractmethod

class Animal(ABC):

    #concrete method
    def sleep(self):
        print("I am going to sleep in a while")

    @abstractmethod
    def sound(self):
        print("This function is for defining the sound by any animal")
        pass

class Snake(Animal):
    def sound(self):
        print("I can hiss")

class Dog(Animal):
    def sound(self):
        print("I can bark")

class Lion(Animal):
```

	Expected	Got	
✔	I am going to sleep in a while I can meow I can hiss	I am going to sleep in a while I can meow I can hiss	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Not answered

Mark 0.00 out of 20.00

Create `Counter` class which has one attribute called `current` which defaults to zero. And it has three methods:

- `increment()` increases the value of the `current` attribute by one.
- `value()` returns the current value of the `current` attribute
- `reset()` sets the value of the `current` attribute to zero

create a new instance of the `Counter` class and calls the `increment()` method three times before showing the current value of the counter to the screen

For example:

Result
3

Answer: (penalty regime: 0 %)

Reset answer

```
class Counter:
    def __init__(self):
        self.current = 0

    def increment(self):
        self.current += 1

    def value(self):
        return self.current

    def reset(self):
        self.current = 0

counter = Counter()


#call the increment method three times

print(counter.value())
```