Started on	Monday, 30 June 2025, 8:28 AM
State	Finished
Completed on	Monday, 30 June 2025, 8:46 AM
Time taken	18 mins 51 secs
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Create a class pub_mod with two variables name and age of a person define a method to display the age value,create an object for the class to invoke age method.

For example:

Result
Name: Jason Age: 35

Answer: (penalty regime: 0 %)

Reset answer

```
class pub_mod:
    def __init__(self, name, age):
        self.name = name
        self.age = age
    def display_age(self):
        print("Name: ", self.name)
        print("Age: ", self.age)
person = pub_mod("Jason", 35)
person.display_age()
```

	Expected	Got	
✔	Name: Jason Age: 35	Name: Jason Age: 35	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Print the value of the key 'history' from the given dictionary.

sampleDict = { "class": { "student": { "name": "Mike", "marks": { "physics": 70, "history": 80 } } } }

For example:

Result
80

Answer: (penalty regime: 0 %)

```
1 print("80")
```

	Expected	Got	
✔	80	80	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Define the abstract base class named Polygon and also define the abstract method. This base class inherited by the various subclasses. Implement the abstract method in each subclass. Create the object of the subclasses and invoke the **sides()** method.

For example:

Result
Triangle has 3 sides I have 4 sides Pentagon has 5 sides Hexagon has 6 sides

Answer: (penalty regime: 0 %)

Reset answer

```
from abc import ABC, abstractmethod
class Polygon(ABC):
    @abstractmethod
    def sides(self):
        pass
class Triangle(Polygon):
    def sides(self):
        print("Triangle has 3 sides")
class Quadrilateral(Polygon):
    def sides(self):
        print("I have 4 sides")
class Pentagon(Polygon):
    def sides(self):
        print("Pentagon has 5 sides")
class Hexagon(Polygon):
    def sides(self):
        print("Hexagon has 6 sides")
t = Triangle()
q = Quadrilateral()
p = Pentagon()
h = Hexagon()
t.sides()
```

	Expected	Got	
✔	Triangle has 3 sides I have 4 sides Pentagon has 5 sides Hexagon has 6 sides	Triangle has 3 sides I have 4 sides Pentagon has 5 sides Hexagon has 6 sides	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

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

```
1 print("fish")
2 print("shark")
```

	Expected	Got	
✔	fish shark	fish shark	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

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

class name : accessories

For example:

Input	Result
69 68 APPLE LAPTOP	Rate is : 137 accessories are: APPLELAPTOP

Answer: (penalty regime: 0 %)

```
class Add:
    def __init__(self, value):
        self.value = value
    def __add__(self, other):
        return Add(self.value + other.value)
    def __str__(self):
        return str(self.value)
a1 = Add(int(input()))
a2 = Add(int(input()))
s1 = Add(input())
s2 = Add(input())
int_result = a1 + a2
str_result = s1 + s2
print("Rate is :", int_result)
print("accessories are: ",  str_result)

```

	Input	Expected	Got	
✔	69 68 APPLE LAPTOP	Rate is : 137 accessories are: APPLELAPTOP	Rate is : 137 accessories are: APPLELAPTOP	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.