**St. Francis Institute of Technology, Mumbai-
400 103 Department Of Information
Technology**

**A.Y. 2023-24
Class: SE-ITA/B, Semester:
IV**

Subject: Python Lab

Experiment – 7: Python program to implement Classes, inner-Classes and Constructors

1. Aim: To design a python program to implement the following:
i) Declare a Class with class-name Student which accepts the Student details, creates an inner class of Student Marks with a constructor that takes marks as arguments and returns the total and percentage of marks along with the student details

2.Prerequisite: Basic knowledge of Python.

3. Objective: Knowledge of classes and objects in python

4. Requirements: Personal Computer (PC), Windows /Linux Operating System, IDLE 3.10 for Python3.

**5. Pre-Experiment
Exercise:
Theory:
Classes and Objects:**
Classes provide a means of bundling data and functionality together. Creating a new class creates a
new type of object, allowing new instances of that type to be made. Each class instance can have
attributes attached to it for maintaining its state. Class instances can also have methods (defined by
its class) for modifying its state.
Syntax:
instance = class (arguments)

Static Method:
The staticmethod () built-in function returns a static method for a given function. Using staticmethod () is considered un-Pythonic way of creating a static function. So, in newer versions of
Python, you can use the Python decorator @staticmethod. The syntax of @staticmethod is:
@staticmethod def func (args ...)

Inner Classes:
An inner class or nested class is a defined entirely within the body of another class. If an object is created using a class, the object inside the root class can be used. A class can have more than one inner classes, but in general inner classes are avoided.
Constructors:

A constructor is a special kind of method that Python calls when it instantiates an object using the
definitions found in your class. Python relies on the constructor to perform tasks such as initializing
(assigning values to) any instance variables that the object will need when it starts.
The name of a constructor is always the same, init__(). The constructor can accept arguments when necessary to create the object. When you create a class without a constructor, Python automatically creates a default constructor for you that doesn't do anything.

6. Laboratory Exercise
A. Procedure

i. Open Idle for python
ii. Open editor in Idle from menu file-new
iii. Type python code with proper syntax
iv. Save file with .py extension
v. Execute the code inside the saved file using shortcut key F5 or using menu: Run-Run module

B. Results/Observations/Program output:
Present the program input/output results and comment on the same.

7. Post-Experiments Exercise
A. Extended Theory:
1. Explain the concept of Inner Classes in Python with one example.
2. Explain self-variable with example.

B. Questions/Programs:
1. Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.
2. Write a python program to create a Bank class where deposits and withdrawals can be handled by using instance methods.

C. Conclusion:
A. Write what was performed in the program(s).
B. What is the significance of the program(s).

D. References
1. James Payne, "Beginning Python: Using Python 2.6 and Python 3.1", WroxPublication.
2. Dr. R. Nageshwara Rao, "Core Python Programming", Dreamtech Press.
3. https://www.python.org/
4. www.pythonforbeginners.com

In-Lab Program

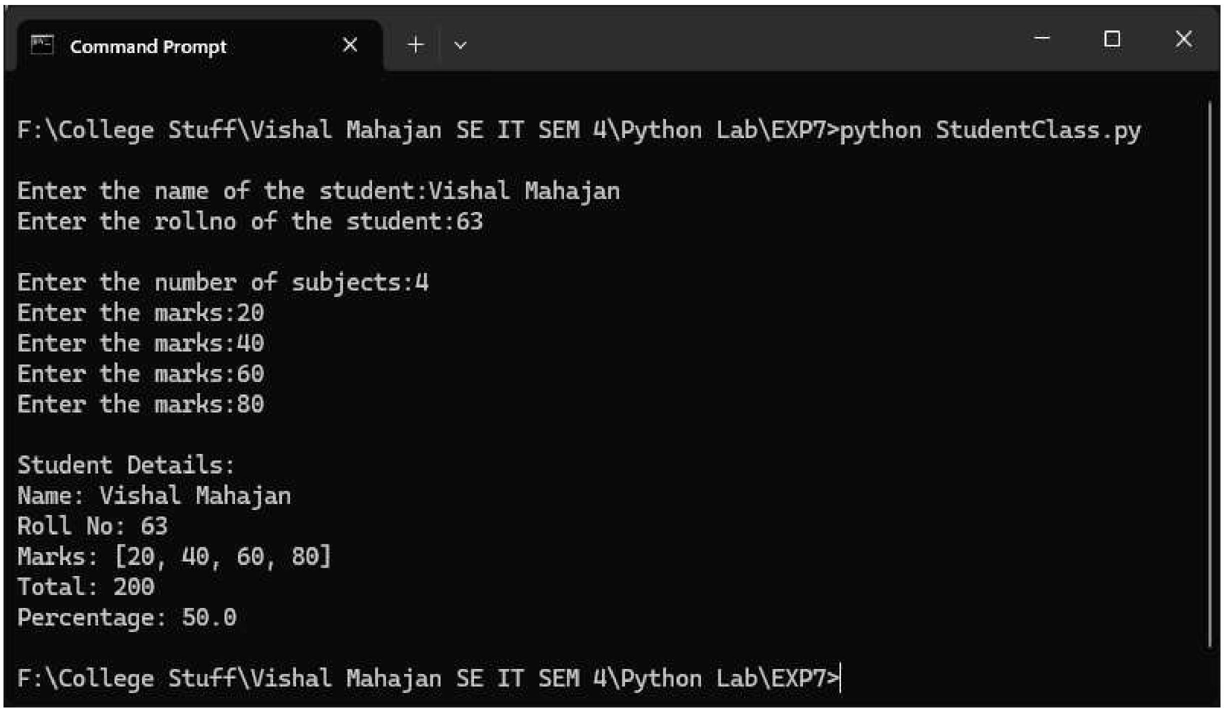
Q1 Declare a Class with class-name Student which accepts the Student details, creates an inner class of Student Marks with a constructor that takes marks as arguments and returns the total and percentage of marks along with the student details

Code:

```python
class Student:
    def __init__(self, name, rollno):
        self.name = name
        self.rollno = rollno

    class StudentMarks:
        def __init__(self, student, marks):
            self.student = student
            self.marks = marks

        def total(self):
            return sum(self.marks)

        def percentage(self):
            return (sum(self.marks) / len(self.marks))

        def display(self):
            print("\nStudent Details:")
            print("Name:", self.student.name)
            print("Roll No:", self.student.rollno)
            print("Marks:", self.marks)
            print("Total:", self.total())
            print("Percentage:", self.percentage())
```

```python
name = input("\nEnter the name of the student:")
rollno = int(input("Enter the rollno of the student:"))

Student_obj = Student(name, rollno)


limit = int(input("\nEnter the number of subjects:"))


marks = []
for i in range(limit):
    marks.append(int(input("Enter the marks:")))
StudentMarks_obj = Student_obj.StudentMarks(Student_obj,marks)
StudentMarks_obj.display()
```

OUTPUT:



```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP7>python StudentClass.py

Enter the name of the student:Vishal Mahajan
Enter the rollno of the student:63

Enter the number of subjects:4
Enter the marks:20
Enter the marks:40
Enter the marks:60
Enter the marks:80

Student Details:
Name: Vishal Mahajan
Roll No: 63
Marks: [20, 40, 60, 80]
Total: 200
Percentage: 50.0

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP7>
```

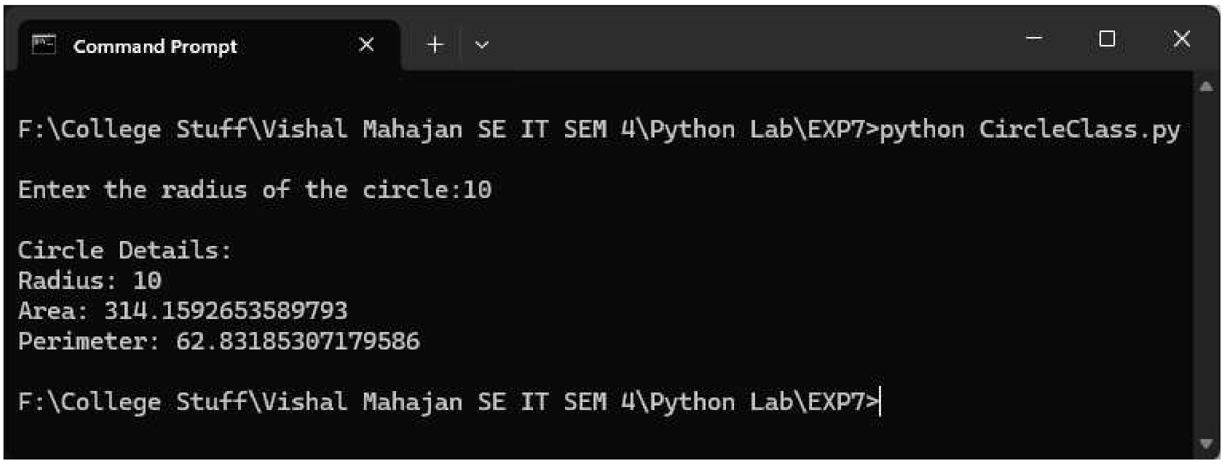
POST EXPERIMENT

1.Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.

Code:

```
class Circle:
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return math.pi * self.radius ** 2

    def perimeter(self):
        return 2 * math.pi * self.radius

    def display(self):
        print("\nCircle Details:")
        print("Radius:", self.radius)
        print("Area:", self.area())
        print("Perimeter:", self.perimeter())

radius = int(input("\nEnter the radius of the circle:"))
Circle_obj = Circle(radius)
Circle_obj.display()
```

Output:



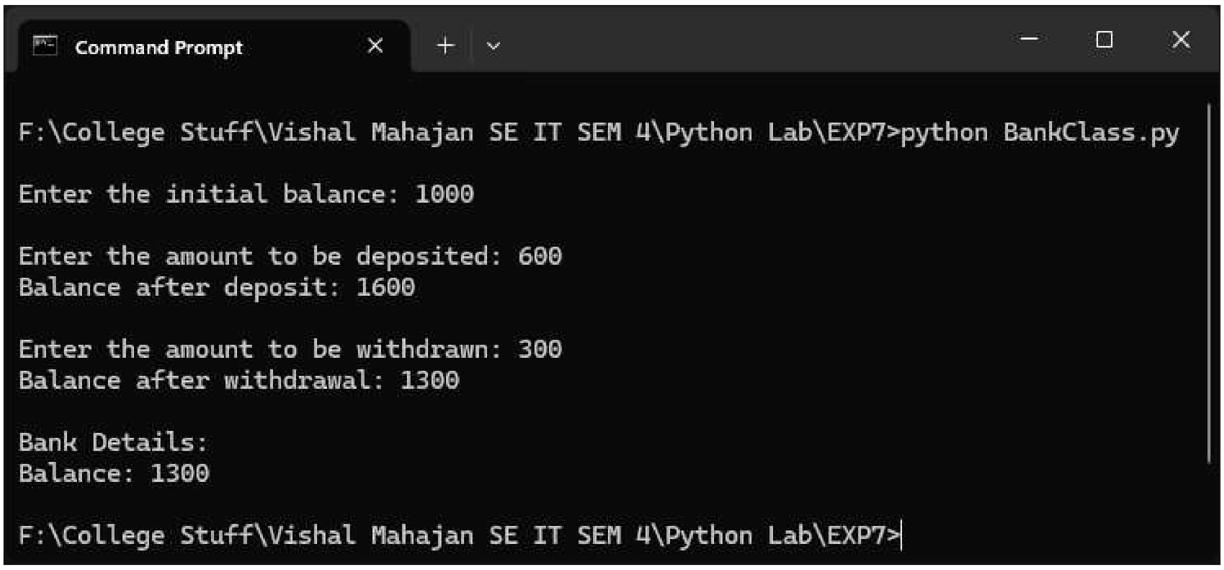
```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP7>python CircleClass.py

Enter the radius of the circle:10

Circle Details:
Radius: 10
Area: 314.1592653589793
Perimeter: 62.83185307179586

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP7>
```

2.Write a python program to create a Bank class where deposits and withdrawals can be handled by using instance methods.

Code:

```python
class Bank:
    def __init__(self, balance):
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        return self.balance

    def withdraw(self, amount):
        if amount > self.balance:
            return "Insufficient balance"
        else:
            self.balance -= amount
            return self.balance

    def display(self):
        print("\nBank Details:")
        print("Balance:", self.balance)

balance = int(input("\nEnter the initial balance: "))
Bank_obj = Bank(balance)

amount = int(input("\nEnter the amount to be deposited: "))
print("Balance after deposit:", Bank_obj.deposit(amount))

amount = int(input("\nEnter the amount to be withdrawn: "))
print("Balance after withdrawal:", Bank_obj.withdraw(amount))

Bank_obj.display()
```

Output:



```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP7>python BankClass.py

Enter the initial balance: 1000

Enter the amount to be deposited: 600
Balance after deposit: 1600

Enter the amount to be withdrawn: 300
Balance after withdrawal: 1300

Bank Details:
Balance: 1300

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP7>
```