Name : **Shawn Louis** Batch : **B** Roll No : **31**

Experiment No: 7

Topic:	To write program using classes and objects in python.
Prerequisite:	Knowledge of some programming language like C, Java
Mapping With COs:	CSL405.2
Objective:	Ability to write program by implementing Object Oriented concepts in python.
Outcome:	To learn how to design object-oriented programs with Python classes. To learn how to use class inheritance in Python for reusability.
Bloom's Taxonomy :	Apply

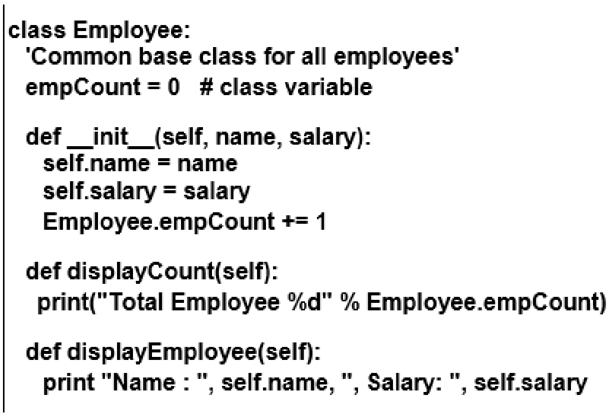
Theory/ Steps/ Algorithm/ Procedure:

Creating Classes:

The class statement creates a new class definition. The name of the class immediately follows the keyword class followed by a colon as follows −

class ClassName:
'Optional class documentation string' class_suite

- The class has a documentation string, which can be accessed via ClassName. doc .
- The class_suite consists of all the component statements defining class members, data attributes and functions.



```
class Employee:
  'Common base class for all employees'
  empCount = 0   # class variable

  def __init__(self, name, salary):
    self.name = name
    self.salary = salary
    Employee.empCount += 1

  def displayCount(self):
   print("Total Employee %d" % Employee.empCount)

  def displayEmployee(self):
    print "Name : ", self.name, ", Salary: ", self.salary
```

The variable empCount is a class variable whose value is shared among all instances of a this class. This can be accessed as

Employee.empCount from inside the class or outside the class.

- The first method init () is a special method, which is called class constructor or initialization method that Python calls when you create a new instance of this class.

• You declare other class methods like normal functions with the exception that the first argument to each method is self. Python adds the self argument to the list for you; you do not need to include it when you call the methods.

Creating Instance Objects

To create instances of a class, you call the class using class name and pass in whatever arguments its__init__method accepts.

"This would create first object of Employee class"

emp1 = Employee("Zara", 2000)

"This would create second object of Employee class"

emp2 = Employee("Manni", 5000)

Experiments:

1. Write a Python class named Rectangle constructed by a length and width and a method which will compute the area of a rectangle.

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

3. Write a Python program to create Bankaccount class with deposit, withdraw, balanceCheck functions.

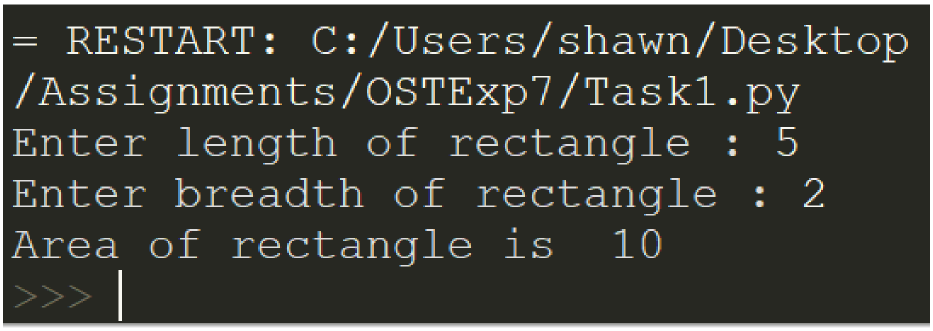
4. Implement Inheritance concept in Python. (For any Realtime Application).

Deliverables:

1. **Write a Python class named Rectangle constructed by a length and width and a method which will compute the area of a rectangle**

```
class Rectangle:
    """ This a Rectangle class
   which stores attributes and methods
    """

    def __init__(self, length, breadth):
        self.l = length
        self.b = breadth

    def area(self):
        a = l * b
        print("Area of rectangle is ", a)

l = int(input("Enter length of rectangle : "))
b = int(input("Enter breadth of rectangle : "))

#print(Rectangle.__doc__)

rect = Rectangle(l, b)
rect.area()
```

Output :

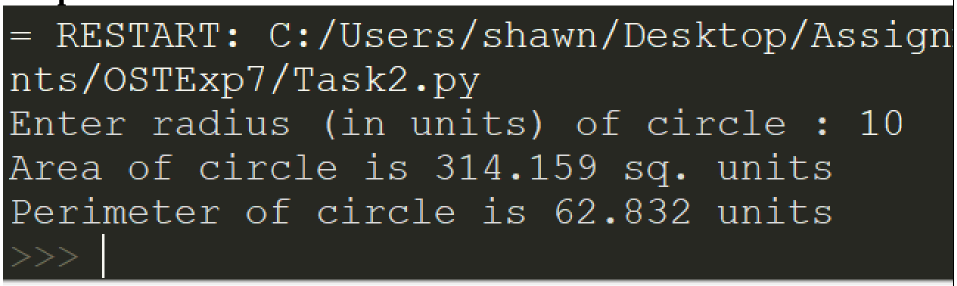


```
= RESTART: C:/Users/shawn/Desktop
/Assignments/OSTExp7/Task1.py
Enter length of rectangle : 5
Enter breadth of rectangle : 2
Area of rectangle is  10
>>> |
```

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

```
from math import pi

class Circle:
    """This class is a template of
  attributes and methods of a circle"""

    def __init__(self, radius):
        self.r = radius

    def area(self):
        a = pi * r * r
        print("Area of circle is {} sq.
units".format(round(a, 3)))

    def perimeter(self):
        p = 2 * pi * r
        print("Perimeter of circle is {}
units".format(round(p, 3)))

r = int(input("Enter radius (in units) of circle : "))

c = Circle(r)
c.area()
c.perimeter()
```

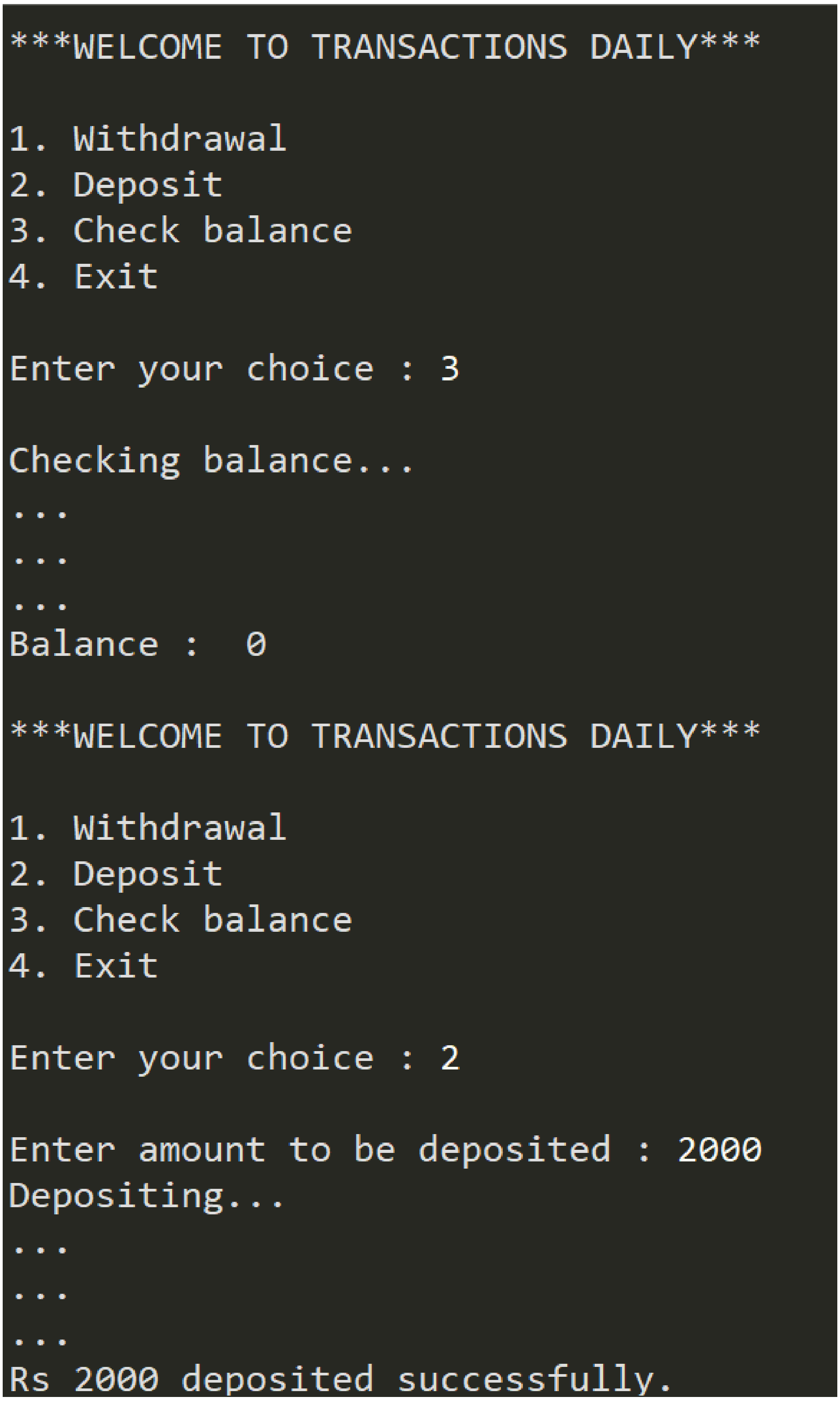
Output :



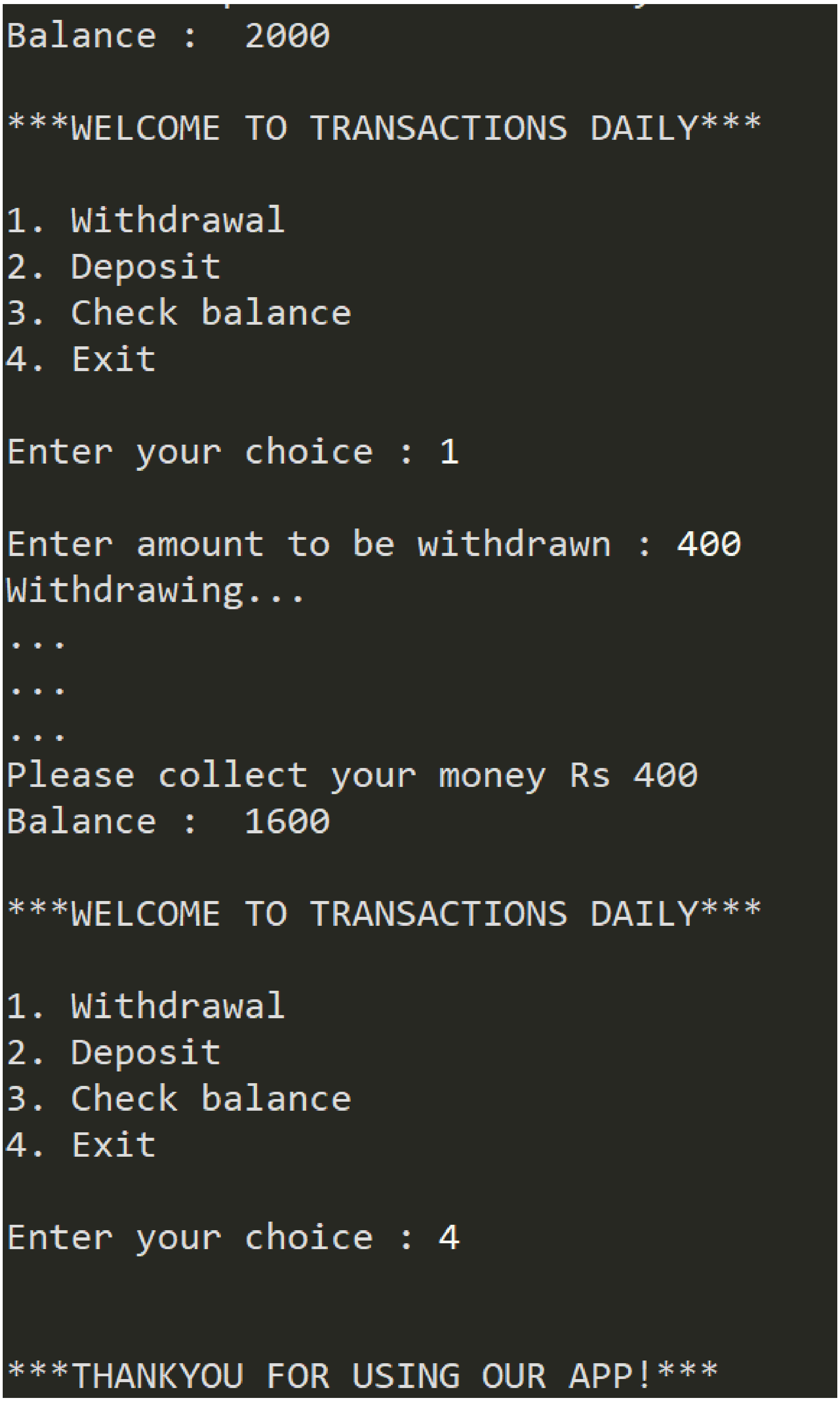
```
= RESTART: C:/Users/shawn/Desktop/Assign
nts/OSTExp7/Task2.py
Enter radius (in units) of circle : 10
Area of circle is 314.159 sq. units
Perimeter of circle is 62.832 units
>>> |
```

3. Write a Python program to create Bankaccount class with deposit, with- draw, balanceCheck functions.

```
import time

class BankAccount:
    def __init__(self):
        self.acc = 0
    def deposit(self, amount):
        self.acc += amount
    def withdraw(self, amount):
        self.acc -= amount
    def balanceCheck(self):
        print("Balance : ", self.acc)

acc = BankAccount()
while True:
    print("\n***WELCOME TO TRANSACTIONS DAILY***\n")
    print("1. Withdrawal")
    print("2. Deposit")
    print("3. Check balance")
    print("4. Exit")

    choice = int(input("\nEnter your choice : "))

    if choice == 1:
        amount = int(input("\nEnter amount to be 
withdrawn : "))
        acc.withdraw(amount)
        print("Withdrawing...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("Please collect your money Rs 
{}".format(amount))
        acc.balanceCheck()

    elif choice == 2:
        amount = int(input("\nEnter amount to be 
deposited : "))
        acc.deposit(amount)
        print("Depositing...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("Rs {} deposited 
successfully.".format(amount))
        acc.balanceCheck()

    elif choice == 3:
        print("\nChecking balance...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        print("...")
        time.sleep(0.5)
        acc.balanceCheck()

    elif choice == 4:
        time.sleep(1.5)
        print("\n\n***THANKYOU FOR USING OUR APP!***")
        time.sleep(1)
        break

    time.sleep(1)
```

	Output : 

```
***WELCOME TO TRANSACTIONS DAILY***

1. Withdrawal
2. Deposit
3. Check balance
4. Exit

Enter your choice : 3

Checking balance...
...
...
...
Balance :  0

***WELCOME TO TRANSACTIONS DAILY***

1. Withdrawal
2. Deposit
3. Check balance
4. Exit

Enter your choice : 2

Enter amount to be deposited : 2000
Depositing...
...
...
...
Rs 2000 deposited successfully.
```

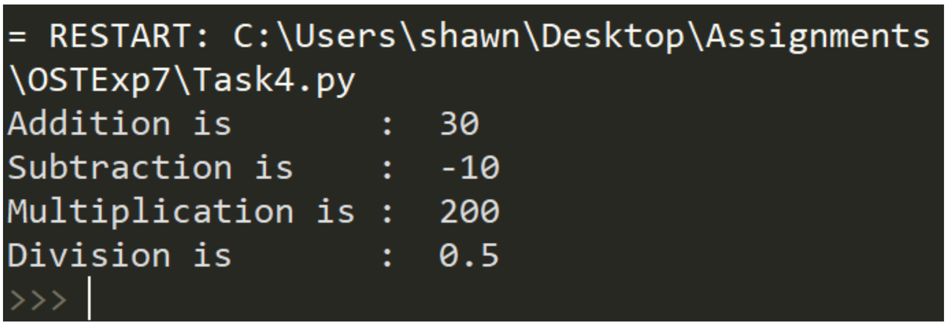


```
Balance :  2000

***WELCOME TO TRANSACTIONS DAILY***

1. Withdrawal
2. Deposit
3. Check balance
4. Exit

Enter your choice : 1

Enter amount to be withdrawn : 400
Withdrawing...
...
...
...
Please collect your money Rs 400
Balance :  1600

***WELCOME TO TRANSACTIONS DAILY***

1. Withdrawal
2. Deposit
3. Check balance
4. Exit

Enter your choice : 4


***THANKYOU FOR USING OUR APP!***
```

4. Implement Inheritance concept in Python. (For any Realtime Application).

```
class Calculation1:
    def add(self,a,b):
        print("Addition is       : ",(a+b))

class Calculation2:
    def subtract(self,a,b):
        print("Subtraction is    : ",(a-b))

class Calculation3:
    def multiply(self,a,b):
        print("Multiplication is : ",(a*b))

class Calculator(Calculation1,Calculation2,
Calculation3):
    def divide(self,a,b):
        print("Division is       : ",(a/b))

d = Calculator()
d.add(10,20)
d.subtract(10,20)
d.multiply(10,20)
d.divide(10,20)
```

Output :



```
= RESTART: C:\Users\shawn\Desktop\Assignments
\OSTExp7\Task4.py
Addition is       :  30
Subtraction is    :  -10
Multiplication is :  200
Division is       :  0.5
>>> 
```

Conclusion: Thus we have successfully able to write program using classes and objects in python.

References:

https://realpython.com/python3-object-oriented-programming/#classes-in-python

https://www.tutorialspoint.com/python/python_classes_objects.htm

https://docs.python.org/3/tutorial/classes.html

Don Bosco Institute of Technology
Department of Computer Engineering

Academic year – 2019-20

Open Source Technology Lab

Assessment Rubric for Experiment No.: 7

Performance Date :
Submission Date :

Title of Experiment : Classes and Objects in Python

Year and Semester : 2nd Year and IVth Semester

Batch : Computer

Name of Student : Shawn Louis

Roll No. : 31

Performance	**Poor**	**Satisfactory**	**Good**	**Excellent**	**Total**
	2 points	3 points	4 points	5 points	
Results and Documentations	**Poor**	**Satisfactory**	**Good**	**Excellent**	
	2 points	3 points	4 points	5 points	
Timely Submission	**Submission beyond 14 days of the deadline**	**Late submission till 14 days**	**Late submission till 7 days**	**Submission on time**	
	2 points	3 points	4 points	5 points	

Signature

(Sana Shaikh)