**Practical No. 1**

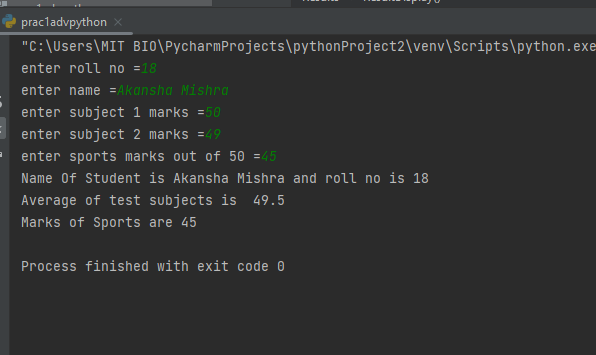
**Aim:** To demonstrate the concept of multipath inheritance and abstraction

1. **Inheritance:**

**Code:**

class Student:  
 def GetStudentData(self):  
 self.rollno=int(input('enter roll no ='))  
 self.name=input('enter name =')  
class Test(Student):  
 def MarksTest(self):  
 self.sub1 = int(input('enter subject 1 marks ='))  
 self.sub2 = int(input('enter subject 2 marks ='))  
class Sport(Student):  
 def SportsTest(self):  
 self.sport = int(input('enter sports marks out of 50 ='))  
class Results(Sport,Test):  
 def ResultDisplay(self):  
 print(f'Name Of Student is {self.name} and roll no is {self.rollno}')  
 avg=(self.sub1+self.sub2)/2.0  
 print('Average of test subjects is ',avg)  
 print(f'Marks of Sports are {self.sport}')  
p1=Results()  
a=p1.GetStudentData()  
b=p1.MarksTest()  
c=p1.SportsTest()  
d=p1.ResultDisplay()

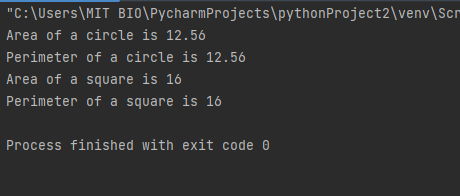
**Output:**



1. **Abstraction:**
2. **Code:**

from abc import ABC, abstractmethod  
class Shape(ABC):  
 @abstractmethod  
 def area(self):  
 pass  
 def perimeter(self):  
 pass  
  
class Circle(Shape):  
 def \_\_init\_\_(self,radius):  
 self.radius=radius  
 def area(self):  
 return 3.14\*self.radius\*\*2  
 def perimeter(self):  
 return 2\*3.14\*self.radius  
  
class Square(Shape):  
 def \_\_init\_\_(self,side):  
 self.side=side  
 def area(self):  
 return self.side\*\*2  
 def perimeter(self):  
 return self.side\*4  
  
circle = Circle(2)  
square = Square(4)  
print(f"Area of a circle is {circle.area()}")  
print(f"Perimeter of a circle is {circle.perimeter()}")  
print(f"Area of a square is {square.area()}")  
print(f"Perimeter of a square is {square.perimeter()}")

**Output:**



1. **Code (using for loop):**

from abc import ABC, abstractmethod

class Shape(ABC):

@abstractmethod

def area(self):

pass

def perimeter(self):

pass

class Circle(Shape):

def \_\_init\_\_(self, radius):

self.radius = radius

def area(self):

return 3.14 \* self.radius \*\* 2

def perimeter(self):

return 2 \* 3.14 \* self.radius

class Square(Shape):

def \_\_init\_\_(self, side):

self.side = side

def area(self):

return self.side \*\* 2

def perimeter(self):

return self.side \* 4

# Initializing the loop

for i in range(0, 100):

rad = int(input("Enter radius of circle: "))

s = int(input("Enter side of square: "))

circle = Circle(rad)

square = Square(s)

print("When radius is", rad)

print("--------------------")

print(f"Area of a circle is {circle.area()}")

print(f"Perimeter of a circle is {circle.perimeter()}")

print("--------------------")

print("When side is", s)

print("--------------------")

print(f"Area of a square is {square.area()}")

print(f"Perimeter of a square is {square.perimeter()}")

print("--------------------")

**Output:**

