

# St. Francis Institute of Technology, Mumbai-40103

A.Y. 2023-24

Class: SE-ITA/ITB, Semester: IV

Subject: **Python Lab.**

## Experiment – 9: Python program to implement Polymorphism.

**1. Aim:** Write a Python program to create a class Vehicle having methods vehicle\_info() and mode\_transport() and derive two subclasses Bike and Train from it. Override the superclass methods inside the two subclasses.

**2. Prerequisite:** Knowledge of OOP concepts like classes, objects, inheritance and polymorphism in Python.

**3. Objectives:** Knowledge of OOP concepts like Polymorphism in Python.

**4. Requirements:** Personal Computer (PC), Windows /Linux Operating System, IDLE 3.6 for Python3.

### 5. Pre-Experiment Exercise:

#### Theory:

#### Polymorphism:

Polymorphism is based on the Greek words Poly (many) and morphism (forms). We will create a structure that can take or use many forms of objects. It's a feature of class definition in Python that is utilized when you have commonly named methods across classes or subclasses. Polymorphism can be carried out through [inheritance](#), with subclasses making use of base class methods or overriding them.

Example:

```
class Vehicle():
    def display(self):
        print("The Vehicle is Running.")
class Car():
    def display(self):
        print("The Car is Running.")
```

#### a) Method Overloading:

In Python you can define a method in such a way that there are multiple ways to call it. Given a single method or function, we can specify the number of parameters our self. Depending on the function definition, it can be called with zero, one, two or more parameters. This is known as *method overloading*. Python does not support method overloading by default. But there are different ways to achieve method overloading in Python. The problem with method overloading in Python is that we may overload the methods but can only use the latest defined method. If any other method is used to call, then program throws an error.

Two ways to achieve method overloading:

**Method 1 (Not The Most Efficient Method):**

We can use the variable length arguments to make the same function work differently i.e. as per the arguments

**Method 2 (Not the efficient one):**

We can achieve method overloading in python by user defined function using “None” keyword as default parameter.

**Method 3 (Efficient One):**

By Using Multiple Dispatch Decorator. The Multiple Dispatch Decorator Can be installed by:  
pip3 install multipledispatch

and then import it in the program

from multipledispatch import dispatch

**b) Method Overriding:**

Overriding is the ability of a class to change the implementation of a method provided by one of its ancestors. If there is any method in the superclass and a method with the same name in a subclass, then by executing the method, the method of the corresponding class will be executed.

The version of a method that is executed will be determined by the object that is used to invoke it. If an object of a parent class is used to invoke the method, then the version in the parent class will be executed, but if an object of the subclass is used to invoke the method, then the version in the child class will be executed. In other words, it is the type of the object being referred to (not the type of the reference variable) that determines which version of an overridden method will be executed.

**Method overriding with multiple and multilevel inheritance**

**Multiple Inheritance:** When a class is derived from more than one base class it is called multiple Inheritance.

**Multilevel Inheritance:** When we have a child and grandchild relationship

Python super() function provides us the facility to refer to the parent class explicitly. It is basically useful where we have to call superclass functions. It returns the proxy object that allows us to refer parent class by ‘super’

## 6. Laboratory Exercise

### A. Procedure

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

### B. Program code with comments:

Write and execute your program code to achieve the given aim and attach it **with your own comments with neat indentation.**

### Post-Experiments Exercise

#### a. Extended Theory:

Differentiate between Method Overloading and Method Overriding.

#### b. Questions/Programs:

Write a Python program to declare a class Calculate which calculates the Area of Circle, Triangle and Rectangle (Use Method Overloading).

#### c. Conclusion:

1. Write what was performed in the experiment/program.
2. What is the significance of experiment/program?

#### d. References

- [1] James Payne, "Beginning Python: Using Python 2.6 and Python 3.1", WroxPublication.
- [2] Dr.Nageswara Rao,"Core Python Programming",Wiley Publication.
- [3] <https://www.python.org/>
- [4] [www.pythonforbeginners.com](http://www.pythonforbeginners.com)
- [5] <https://www.w3schools.com/python>
- [6] <https://www.geeksforgeeks.org/python-method-overloading/>

## In-Lab Exercise

Q1) Write a Python program to create a class `Vehicle` having methods `vehicle_info()` and `mode_transport()` and derive two subclasses `Bike` and `Train` from it. Override the superclass methods inside the two subclasses.

Code:

```
class Vehicle:
    def vehicle_info(self):
        print("This is a vehicle")

    def mode_transport(self):
        print("This is a mode of transport")

class Bike(Vehicle):
    def vehicle_info(self):
        print("This is a bike")

    def mode_transport(self):
        print("This is a mode of road transport")

class Train(Vehicle):
    def vehicle_info(self):
        print("This is a train")

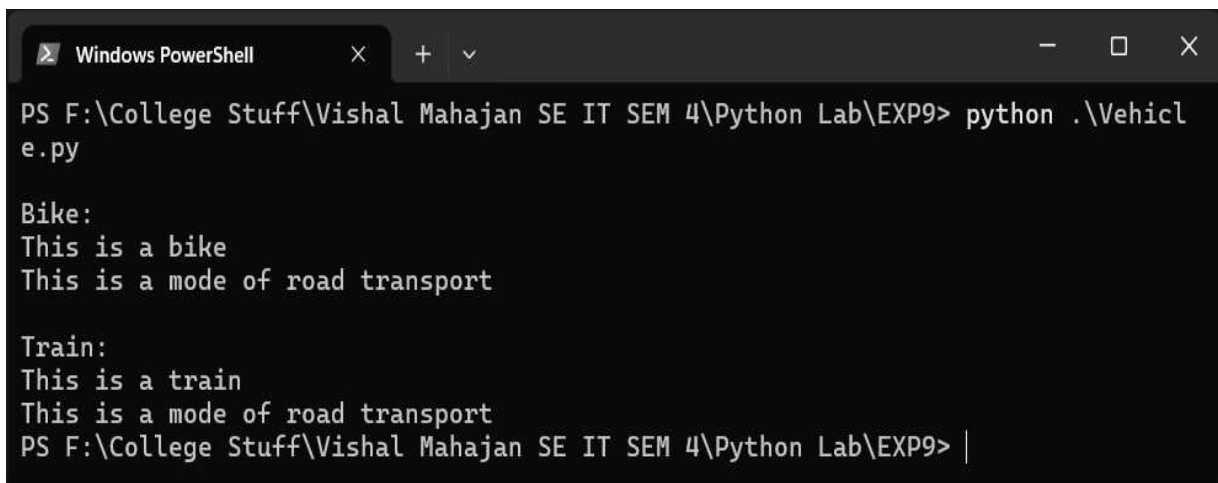
    def mode_transport(self):
        print("This is a mode of road transport")

bike = Bike()
train = Train()
```

```
print("\nBike:")
bike.vehicle_info()
bike.mode_transport()

print("\nTrain:")
train.vehicle_info()
train.mode_transport()
```

### Output:



The screenshot shows a Windows PowerShell window with the following content:

```
Windows PowerShell
PS F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP9> python .\Vehicle.py

Bike:
This is a bike
This is a mode of road transport

Train:
This is a train
This is a mode of road transport
PS F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP9> |
```

### Post Experiment:

Write a Python program to declare a class Calculate which calculates the Area of Circle, Triangle and Rectangle (Use Method Overloading).

### Code:

```
class Calculate:
    def area(self, **kwargs):
        if "radius" in kwargs:
            return self.area_circle(**kwargs)

        elif "length" in kwargs and "breadth" in kwargs:
            return self.area_rectangle(**kwargs)

        elif "base" in kwargs and "height" in kwargs:
            return self.area_triangle(**kwargs)

        else:
            return "Invalid arguments"

    def area_circle(self, **kwargs):
        radius = kwargs["radius"]
        return 3.14 * radius * radius

    def area_rectangle(self, **kwargs):
        length = kwargs["length"]
        breadth = kwargs["breadth"]
        return length * breadth
```

```
def area_triangle(self, **kwargs):
    base = kwargs["base"]
    height = kwargs["height"]
    return 0.5 * base * height

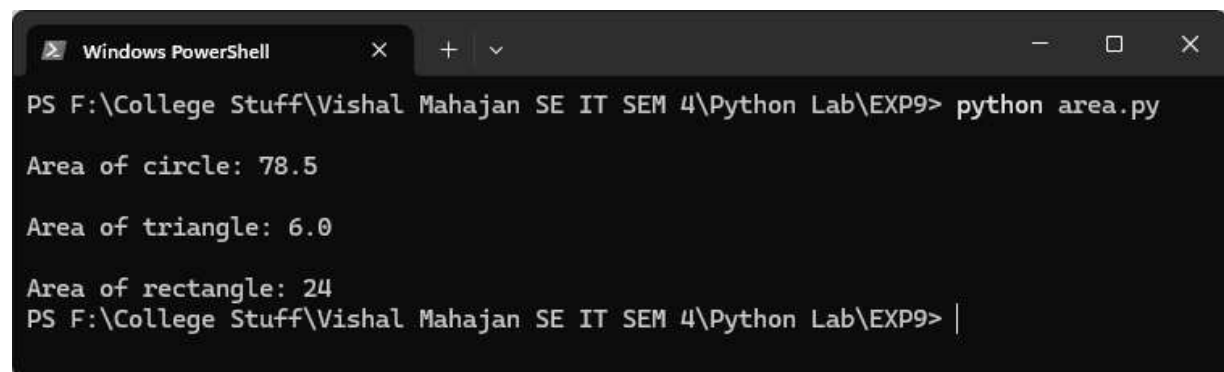
calculator = Calculate()

circle_area = calculator.area(radius=5)
print("\nArea of circle:", circle_area)

triangle_area = calculator.area(base=3, height=4)
print("\nArea of triangle:", triangle_area)

rectangle_area = calculator.area(length=4, breadth=6)
print("\nArea of rectangle:", rectangle_area)
```

**Output:**



```
Windows PowerShell
PS F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP9> python area.py

Area of circle: 78.5

Area of triangle: 6.0

Area of rectangle: 24
PS F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP9> |
```