| Laboratory Activity No.7 | |
|---|---|
| **Polymorphism** | |
| **Course Code:** CPE103 | **Program:** BSCPE |
| **Course Title:** Object-Oriented Programming | **Date Performed:** 03/01/25 |
| **Section:** 1-A | **Date Submitted:** 03/01/25 |
| **Name:** GABIJAN, RHOVIC M. | **Instructor:** ENGR. SAYO |

**1.Objective(s):**

This activity aims to familiarize students with the concepts of Polymorphism in Object-Oriented Programming

**2. Intended Learning Outcomes (ILOs):**

The students should be able to:
2.1 Identify the use of Polymorphism in Object-Oriented Programming
2.2 Implement an Object-Oriented Program that applies Polymorphism

**3. Discussion:**

Polymorphism is a core principle of Object-Oriented that is also called “method overriding”. Simply stated the principles says that a method can be redefined to have a different behavior in different derived classes.

For an example, consider a base file reader/writer class then three derived classes Text file reader/writer, CSV filer eader/writer, and JSON file reader/writer. The base file reader/writer class has the methods: read (filepath=””) , write (filepath=””).The three derived classes (classes that would inherit from the base class) should have behave differently when their read, write methods are invoked.

Operator Overloading:
Operator overloading is an important concept in object oriented programming. It is a type of polymorphism in which a user defined meaning can be given to an operator in addition to the predefined meaning for the operator.

Operator overloading allow us to redefine the way operator works for user-defined types such as objects. It cannot be used for built-in types such as int, float, char etc., For example, '+' operator can be over loaded to perform addition of two objects of distance class.

Python provides some special function or magic function that is automatically invoked when it is associated with that particular operator. For example, when we use + operator on objects, the magic method __add__() is automatically invoked in which the meaning/operation for + operator is defined for user defined objects.

**4. Materials and Equipment:**

Windows Operating System
Google Colab

**5. Procedure:**

**Creating the Classes**

1. Create a folder named oopfa1 <lastname>_lab 8
2. Open your IDE in that folder.
3. Create the base polymorphism_a.ipynb file and Class using the code below:

Coding:

```
#distance is a class. Distance is measured in terms of feet and inches
class distance:
 def __init__(self,f,i):
 self.feet=f
 self.inches=i

#overloading of binary operator > to compare two distances
 def __gt__(self,d):
 if(self.feet>d.feet):
 return(True)
 elif((self.feet==d.feet) and (self.inches>d.inches)):
 return(True)
 else:
 return(False)

#overloading of binary operator + to add two distances
 def __add__(self, d):
 i=self.inches+d.inches
 f=self.feet + d.feet
 if(i>=12):
 i=i-12
 f=f+1
 return distance(f,i)

#displaying the distance
 def show(self):
 print("Feet=",self.feet,"Inches=",self.inches)

a,b=(input("Enter feet and inches of distance 1:")).split()
a,b =[int(a),int(b)]
c,d=(input("Enter feet and inches of distance 2:")).split()
c,d =[int(c),int(d)]
d1=distance(a,b)
d2=distance(c,d)

if(d1>d2):
 print("Distance 1 is greater than Distance
2")  else:
 print("Distance 2 is greater or equal to Distance 1")
d3=d1+d2
print("Sum of the two Distance is:")
d3.show()
```

4.Screenshot of the program output:

```
Enter feet and inches of distance 1: 5 82
Enter feet and inches of distance 2: 6 64
Distance 2 is greater or equal to Distance 1
Sum of the two Distance is:
Feet= 12 Inches= 134
```

**Testing and Observing Polymorphism**

1. Create a code that displays the program below:

```
class RegularPolygon:
    def __init__ (self, side):
        self._side = side
class Square (RegularPolygon):
    def area (self):
        return self._side * self._side
class EquilateralTriangle (RegularPolygon):
    def area (self):
        return self._side * self._side * 0.433

obj1 = Square(4)
obj2 = EquilateralTriangle(3)

print (obj1.area())
print (obj2.area())
```

```
16
3.897
```

2. Save the program as polymorphism_b.ipynb and paste the screenshot below:

3. Run the program and observe the output.
4. Observation:

The code use a polymorphism which the child class inherits the attributes of a parent class while the child class can make their own attribute without affecting other child class. In this code we use self. side as the attribute of a parent class and also can be use by the child class.

**6. Supplementary Activity:**

In the above program of a Regular polygon, add three more shapes and solve for their area using each proper formula. Take a screenshot of each output and describe each by typing your proper labeling.

```
The area of Square:                16
The area of Equilateral Triangele: 3.897
The area of Hexagon:               127.30199999999999
The area of Circle:                50.26548245743669
The area of Rectangle:             54
```

The area of Square use the self._side attribute of the parent class (RegularPolygon). Next is the area of Equilateral Triangle which also uses the attribute of the parent class and multiply it by 0.433 which is the formula for triangle. Another is the hexagon, is also using the attribute from parent class and multiply by 2.598 which is the formula for area of hexagon. Additionally, the circle needs to import math to have the value for pi (3.14) and also used the attribute self._side from the parent class. Lastly, the area of rectangle have its own attribute which is the self._side2 to compute its area with the help of its own attribute and parent class attribute (self._side*self._side2).

## Questions

1. **Why is Polymorphism important?**
   It is important since in enhances code reusability and flexibility to allow the programmer work with different data types and classes, enabling overriding and dynamic method class as well as reducing redundancy and repeating of codes.

2. **Explain the advantages and disadvantages of using applying Polymorphism in an Object-Oriented Program.**
   The advantages of polymorphism is the code reusability, reducing redundancy, and repeating of codes. While the disadvantages of this is it introduce complexity especially in beginners.

3. **What may be the advantage and disadvantage of the program we wrote to read and write csv and json files?**
   Using CSV and JSON offers easy data storage, readability, and compatibility in different applications however, the both have lacks of data structure supports and strict data types enforcement leading to inconsistencies.

4. **What may be considered if Polymorphism is to be implemented in an Object-Oriented Program?**
   To implement polymorphism it is important to design a clear and consistent interface so multiple classes can implement or override to ensure its reusability and flexibility.

5. **How do you think Polymorphism is used in an actual programs that we use today?**
   Polymorphism is used in GUI or Graphical user interface like buttons, text, fields, all inherits from a common ui components, Also web development and game development.

## 7.Conclusion:

Polymorphism is an essential concept that enables code reusability and flexibility providing a common interface for different objects, while can each behave differently. The advantages of this is it reduce repetition of codes and redundancy making it more readable and flexible while its disadvantage is it introduce complexity in beginners.

## 8.Assessment Rubric: