**Practical No.9**

**Name**: Anuj Rajendra Mane

**Div**: SY A

**Roll No**: 65 **Batch**: A3

**PRN** **No**: 2162701242054

**Title**: **Program on Polymorphism**

**Theory:**

**Polymorphism:**

1.It means one name many forms so we can say that in this type of programming same function is used to perform different kind of operation.

2.It is an important part of object oriented programming language.

**Example**

Let's take a simple example in which as you can see that class name is poly and there are four funation with same name a() with different parameter so execution of function is based on the value passing at the time of calling.

class Poly

{

void a()

{

System.out.println("No para");

}

void a(int i)

{

System.out.println("Integer para");

}

void a(char ch)

{

System.out.println("Character para");

}

void a(float f)

{

System.out.println("Float para");

}

void a(double d)

{

System.out.println("Double para");

}

void a(String s)

{

System.out.println("String para");

}

}

class Easy

{

public static void main(String[] args)

{

//Creating instance(object)

Poly obj=new Poly();

obj.a(12);

obj.a('e');

obj.a("Easy");

}

}

/\*

### Output ###

Integer para

Character para

String para

\*/

Here function is called three times first time with integer value and the second time with character value and third time with string value so the output is

Integer para

character para

String para

**Types of Polyphorphism**

**1.Compile time Polymorphism**

Function overloading and Operator overloading are the example of Compile time Polymorphism.

**2.Runtime Polymorphism**

Function overriding is the example of Runtime Polymorphism.

**1.Function overloading:**

1.The function with same name and different parameter is called function overloading.

class Geometry

{

void area(int height,int width)

{

int ar=height\*width;

System.out.println("Area of Rectangle="+ar);

}

void area(int side)

{

int ar=side\*side;

System.out.println("Area of Square="+ar);

}

void area(float r)

{

float ar=3.14f\*r\*r;

System.out.println("Area of Circle="+ar);

}

void area(float base,float height)

{

float ar=0.5f\*base\*height;

System.out.println("Area of Triangle="+ar);

}

}

class Easy

{

public static void main(String[] args)

{

//Creating instance(object)

Geometry obj=new Geometry();

//single para for square

obj.area(12);

//double int para for rectangle

obj.area(5,6);

//single float for circle

obj.area(2.2f);

//double float for trianlge

obj.area(2.5f,6.3f);

}

/\*

### Output ###

Area of Square=144

Area of Rectangle=30

Area of Circle=15.197601

Area of Triangle=7.875

\*/

Here Geometry is class name contains two function with same name area() with different parameter, first with two parameter height and width and second with single parameter side .Therefor when two integer parameter is passed at the time of calling Rectangle area will be calculated, when single integer parameter is passed Square area will be calculated.

**2.Function Overriding**

1.Function with same name and same parameter is called function overriding.

2.It is not possible to make two function with same name and same parameter in a single class therefor to implement function overriding derived class is used.

class Geometry1

{

void area(int height,int width)

{

int ar=height\*width;

System.out.println("Area of Rectangle1="+ar);

}

}

class Geometry2 extends Geometry1

{

void area(int height,int width)

{

int ar=height\*width;

System.out.println("Area of Rectangle2="+ar);

}

}

class Easy

{

public static void main(String[] args)

{

//Creating instance(object)

Geometry2 obj=new Geometry2();

//calling function

obj.area(12,13);

}

}

/\*

### Output ###

Area of Rectangle2=156

\*/

In the above example there are two classes Geometry1 and Geometry2 containing same function area with same parameter.We are extending Geometry1 into Geometry2 therefore both function exists in the class Geometry2 but on priority bases when we call the function area from the object of class Geomery2 then the function inside Geometry2 will execute.

**Conclusion:**

I have learned the types of polymorphism and the concept of polymorphism also implemented the all types of programs.

**Completion Date: Co-Ordinator Sign:**