# Assignment: 4

**Problem Statement**:- Design a base class shape with two double type values and member functions to input the data and compute\_area() for calculating area of figure. Derive two classes’ triangle and rectangle. Make compute\_area() as abstract function and redefine this function in the derived class to suit their requirements. Write a program that accepts dimensions of triangle/rectangle and display calculated area. Implement dynamic binding for given case study.

**Aim :-** To Study Polymorphism using Java

## Theory:- Polymorphism in Java

The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form.

**Real life example of polymorphism:** A person at the same time can have different characteristic. Like a man at the same time is a father, a husband, an employee. So the same person posses different behaviour in different situations. This is called polymorphism.

Polymorphism is considered as one of the important features of Object Oriented Programming. Polymorphism allows us to perform a single action in different ways. In other words, polymorphism allows you to define one interface and have multiple implementations. The word “poly” means many and “morphs” means forms, So it means many forms.

In Java polymorphism is mainly divided into two types:

* Compile time Polymorphism
* Runtime Polymorphism

1. Compile time polymorphism: It is also known as static polymorphism. This type of polymorphism is achieved by function overloading or operator overloading.

Method Overloading: When there are multiple functions with same name but different parameters then these functions are said to be overloaded. Functions can be overloaded by change in number of arguments or/and change in type of arguments.

Example: By using different types of arguments

// Java program for Method overloading class MultiplyFun {

// Method with 2 parameter

static int Multiply(int a, int b)

{

return a \* b;

}

// Method with the same name but 2 double parameter static double Multiply(double a, double b)

{

return a \* b;

}

}

class Main {

public static void main(String[] args)

{

System.out.println(MultiplyFun.Multiply(2, 4));

System.out.println(MultiplyFun.Multiply(5.5, 6.3));

}

}

Output:

8

34.65

**Operator Overloading**: Java also provide option to overload operators. For example, we can make the operator (‘+’) for string class to concatenate two strings. We know that this is the addition operator whose task is to add two operands. So a single operator ‘+’ when placed between integer operands, adds them and when placed between string operands, concatenates them.

In java, Only “+” operator can be overloaded:

To add integers

To concatenate strings

## Example:

/ Java program for Operator overloading class OperatorOVERDDN {

void operator(String str1, String str2)

{

String s = str1 + str2; System.out.println("Concatinated String - "

+ s);

}

void operator(int a, int b)

{

int c = a + b; System.out.println("Sum = " + c);

}

}

class Main {

public static void main(String[] args)

{

OperatorOVERDDN obj = new OperatorOVERDDN(); obj.operator(2, 3);

obj.operator("joe", "now");

}

}

Output:

Sum =5

Concatinated String -joenow

[Runtime polymorphism](https://www.geeksforgeeks.org/dynamic-method-dispatch-runtime-polymorphism-java/): It is also known as Dynamic Method Dispatch. It is a process in which a function call to the overridden method is resolved at Runtime. This type of polymorphism is achieved by Method Overriding.

[Method overriding](https://www.geeksforgeeks.org/overriding-in-java/), on the other hand, occurs when a derived class has a definition for one of the member functions of the base class. That base function is said to be overridden.

**Objectives** :- 1) To understand the concept of Polymorphism using Java

2) To implement run time polymorphism

## Input:

length and breadth of rectangle base and height of triangle

## Output:

area of rectangle area of circle

## Implementation :-

abstract class shape {

abstract public void compute\_area();

}

class rectangle extends shape { public void compute\_area() {

---

}

}

class triangle extends shape { public void compute\_area() {

---

}

}

public class Shapeclass {

public static void main(String[] args) {

---

}

## Test case or Validation:

Different values for length and breadth of rectangle and base and height of triangle.

**Result:-** Area of Circle and area of Rectangle

**Algorithm:-**

1. Start
2. Create an abstract class named shape that contains two double type numbers and an empty method named compute\_area().
3. Provide two classes named rectangle and triangle such that each one of the classes extends the class Shape.
4. Each of the inherited class from shape class should provide the implementation for the method compute\_area().
5. Get the input and calculate the area of rectangle and triangle.
6. In the fourth separate class, create the objects for the two inherited classes and invoke the methods and display the area values of the different shapes.
7. Stop.

**Code:**

package com.company;  
  
import java.util.Scanner;  
  
abstract class Shape{  
 double a,b;  
 Scanner in = new Scanner(System.*in*);  
 void input(){  
 System.*out*.print("Enter the first variable:");  
 a=in.nextDouble();  
 System.*out*.print("Enter the second variable:");  
 b=in.nextDouble();  
 }  
 abstract public void compute\_area();  
}  
  
  
class Triangle extends Shape{  
 public void compute\_area(){  
 double area;  
 area = (double)(a\*b\*0.5);  
 System.*out*.printf("Area of Triangle = %f",area);  
 }  
}  
class Rectangle extends Shape{  
 public void compute\_area(){  
 double area;  
 area = a\*b;  
 System.*out*.printf("Area of Rectangle = %f ",area);  
 }  
}  
  
public class Assignmnet4 {  
 public static void main(String[] args) {  
 Shape s;  
 Rectangle r = new Rectangle();  
 s=r;  
 System.*out*.print("Enter the dimensions for rectangle : ");  
 s.input();  
 s.compute\_area();  
  
  
 Triangle t = new Triangle();  
 s=t;  
 System.*out*.println("Enter the dimensions for triangle :");  
 s.input();  
 s.compute\_area();  
 }  
}

**Output:**

"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.1\lib\idea\_rt.jar=58237:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.2.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\Rishabh\IdeaProjects\CollegeJavaAssignments\out\production\Assignment2 com.company.Assignmnet4

Enter the dimensions for rectangle :

Enter the first variable:34

Enter the second variable:34

Area of Rectangle = 1156.000000

Enter the dimensions for triangle :

Enter the first variable:43

Enter the second variable:34

Area of Triangle = 731.000000

Process finished with exit code 0

**Conclusion :-** Thus studied the concept of Polymorphism using java.