1.triangle

```
import java.util.Scanner;
class AreaTriangleDemo {
  public static void main(String args[]) {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter the width of the Triangle:");
    double base = scanner.nextDouble();

    System.out.println("Enter the height of the Triangle:");
    double height = scanner.nextDouble();

    double area = (base* height)/2;
    System.out.println("Area of Triangle is: " + area);
  }
}
```

2.switch case

```
import java.util.Scanner;
class Main {
  public static void main(String[] args) {
    char operator;
    Double number1, number2, result;
    // create an object of Scanner class
    Scanner input = new Scanner(System.in);
    // ask users to enter operator
    System.out.println("Choose an operator: +, -, *, or /");
    operator = input.next().charAt(0);
    // ask users to enter numbers
    System.out.println("Enter first number");
    number1 = input.nextDouble();
    System.out.println("Enter second number");
    number2 = input.nextDouble();
    switch (operator) {
      // performs addition between numbers
      case '+':
        result = number1 + number2;
        System.out.println(number1 + " + " + number2 + " = " + result);
        break;
      // performs subtraction between numbers
      case '-':
        result = number1 - number2;
        System.out.println(number1 + " - " + number2 + " = " + result);
        break;
      // performs multiplication between numbers
      case '*':
        result = number1 * number2;
        System.out.println(number1 + " * " + number2 + " = " + result);
        break;
      // performs division between numbers
      case '/':
        result = number1 / number2;
        System.out.println(number1 + " / " + number2 + " = " + result);
        break;
        default:
        System.out.println("Invalid operator!");
        break;
    }

    input.close();
  }
}
```

3 even or odd

```
import java.util.Scanner;

public class EvenOdd {

    public static void main(String[] args) {

        Scanner reader = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int num = reader.nextInt();

        if(num % 2 == 0)
            System.out.println(num + " is even");
```

```
        else
          System.out.println(num + " is odd");
      }
    }
    4.student
    public class Student
    {
      private int roll;
      private String name;
      //constructor to initialize roll number and name of the student
      Student(int rollNo, String sName)
      {
        roll = rollNo;
        name = sName;
      }
      //copy constructor
      Student(Student student)
      {
        System.out.println("\n---Copy Constructor Invoked---");
        roll = student.roll;
        name = student.name;
      }
      //method to return roll number
      int printRoll()
      {
        return roll;
      }
    //Method to return name of the student
      String printName()
      {
        return name;
      }
      //class to create student object and print roll number and name of the student
      public static void main(String[] args)
      {
        Student student1 = new Student(101, "Sneha");
        System.out.println("Roll number of the first student: "+ student1.printRoll());
        System.out.println("Name of the first student: "+ student1.printName());
        //passing the parameter to the copy constructor
        Student student2 = new Student(student1);
        System.out.println("\nRoll number of the second student: "+ student2.printRoll());
        System.out.println("Name of the second student: "+ student2.printName());
      }
    }
    5.adder
    class Adder{
    static int add(int a, int b){return a+b;}
    static double add(double a, double b){return a+b;}
    }
    class TestOverloading2{
    public static void main(String[] args){
    System.out.println(Adder.add(11,11));
    System.out.println(Adder.add(12.3,12.6));
    }}
    Compile by: javac TestOverloading2.java
    Run by: java TestOverloading2
    6. dog()
class Animal{
void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
void bark(){System.out.println("barking...");}
}
class BabyDog extends Dog{
void weep(){System.out.println("weeping...");}
}
class TestInheritance2{
public static void main(String args[]){
BabyDog d=new BabyDog();
d.weep();
d.bark();
d.eat();
}}
7.super
class Animal{
```

```
String color="white";
}
class Dog extends Animal{
String color="black";
void printColor(){
System.out.println(color);//prints color of Dog class
System.out.println(super.color);//prints color of Animal class
}
}
class TestSuper1{
public static void main(String args[]){
Dog d=new Dog();
d.printColor();
}}
```

Compile by: javac TestSuper1.java
Run by: java TestSuper1

8.bank()

```
class Bank{
float getRateOfInterest(){return 0;}
}
class SBI extends Bank{
float getRateOfInterest(){return 8.4f;}
}
class ICICI extends Bank{
float getRateOfInterest(){return 7.3f;}
}
class AXIS extends Bank{
float getRateOfInterest(){return 9.7f;}
}
class TestPolymorphism{
public static void main(String args[]){
Bank b;
b=new SBI();
System.out.println("SBI Rate of Interest: "+b.getRateOfInterest());
b=new ICICI();
System.out.println("ICICI Rate of Interest: "+b.getRateOfInterest());
b=new AXIS();
System.out.println("AXIS Rate of Interest: "+b.getRateOfInterest());
}
}
```

Compile by: javac TestPolymorphism.java
Run by: java TestPolymorphism

9 MultipleCatchBlock1:

```
public class MultipleCatchBlock1 {
        public static void main(String[] args) {
                        try{
                                        int a[]=new int[5];
                                        a[5]=30/0;
                                        }
                                        catch(ArithmeticException e)
                            {
                                                System.out.println("Arithmetic Exception occurs");
                                                }
                                        catch(ArrayIndexOutOfBoundsException e)
                            {
                                                System.out.println("ArrayIndexOutOfBounds Exception occurs");
                                                }
                                       catch(Exception e)
                            {
                                                System.out.println("Parent Exception occurs");
                                                }
                                        System.out.println("rest of the code");
        }
}
```

Compile by: javac MultipleCatchBlock1.java
Run by: java MultipleCatchBlock

10.awt

```
import java.awt.*;
import java.awt.event.*;
public class ButtonExample3 {
public static void main(String[] args) {
   // create instance of frame with the label
   Frame f = new Frame("Button Example");
   final TextField tf=new TextField();
   tf.setBounds(50,50, 150,20);
```

```
    // create instance of button with label
    Button b=new Button("Click Here");
    // set the position for the button in frame
    b.setBounds(50,100,60,30);
    b.addActionListener(new ActionListener() {
    public void actionPerformed (ActionEvent e) {
            tf.setText("Welcome to Javatpoint.");
        }
    });
// adding button the frame
    f.add(b);
// adding textfield the frame
    f.add(tf);
// setting size, layout and visibility
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```

11.printArea()

```
import java.util.*;
abstract class shape
{
 int a,b;
 abstract public void printarea();
}
class rectangle extends shape
{
public int area_rect;
public void printarea()
{
  Scanner s=new Scanner(System.in);
  System.out.println("enter the length and breadth of rectangle");
  a=s.nextInt();
  b=s.nextInt();
  area_rect=a*b;
  System.out.println("Length of rectangle "+a +"breadth of rectangle "+b);
  System.out.println("The area ofrectangle is:"+area_rect);
 }
 }
class triangle extends shape
{
 double area_tri;
 public void printarea()
{
Scanner s=new Scanner(System.in);
System.out.println("enter the base and height of triangle");
a=s.nextInt();
b=s.nextInt();
System.out.println("Base of triangle "+a +"height of triangle  "+b);
area_tri=(0.5*a*b);
System.out.println("The area of triangle is:"+area_tri);
}
}
class circle extends shape
{
double area_circle;
public void printarea()
{
Scanner s=new Scanner(System.in);
System.out.println("enter the radius of circle");
a=s.nextInt();
 area_circle=(3.14*a*a);
 System.out.println("Radius of circle"+a);
 System.out.println("The area of circle is:"+area_circle);
 }
}
public class shapeclass
{
 public static void main(String[] args)
{
rectangle r=new rectangle();
r.printarea();
 triangle t=new triangle();
 t.printarea();
circle r1=new circle();
r1.printarea();
```

```
}
}
```

12.ranable

```
class Mythread implements Runnable {
    public void run() {
        System.out.print("Thread has been created using Runnable interface...!");
    }
}
public class Threads1 {
    public static void main(String[] args) {
        Thread t = new Thread(new Mythread());
        t.start();
    }
}
```

13.static keyword

```
class Student{
  int rollno;
  String name;
  static String college ="ITS";
  Student(int r, String n){
  rollno = r;
  name = n;
  }
  void display (){System.out.println(rollno+" "+name+" "+college);}
}
public class TestStaticVariable1{
 public static void main(String args[]){
 Student s1 = new Student(111,"Karan");
 Student s2 = new Student(222,"Aryan");
 s1.display();
 s2.display();
 }
}
```

14._Constructor

```
class Box{
double width;
double height;
double depth;
    Box(){
System.out.println("Constructing Box");
width = 100;
height = 100;
depth = 100;
    }
    double volume(){
        return width * height * depth;
}
}
class BoxDemo{
    public static void main(String args[]){
        Box mybox1 = new Box();
Box mybox2 = new Box();
        double vol;
        vol = mybox1.volume();
        System.out.println("Volume of the first box is " + vol);
        vol = mybox2.volume();
        System.out.println("Volume of the second box is " + vol);
    }
}
```