ASSIGNMENTS

1.TYPE CASTING

```
package com.Assignments;

public class TypeCasting1
{
	public static void main(String[] args)
	{
		String a="959878799";
		int b=Integer.parseInt(a);
		System.out.println("conversion of string to int "+b);
		float c=Float.parseFloat(a);
		System.out.println("conversion of string to float "+c);
		double d=Double.parseDouble(a);
		System.out.println("conversion of string to double "+d);
		long e=Long.parseLong(a);
		System.out.println("conversion of string to long "+e);
		boolean f=Boolean.parseBoolean(a);
		System.out.println("conversion of string to boolean "+f);
	}

}
```

3.ARITHMETIC CALCULATOR

```
package com.Assignments;
import java.io.*;
import java.lang.Math;
import java.util.*;

public class ArithmeticCalculator3
{
	public static void main(String[] args)
	{
		double n1,n2;

		Scanner s=new Scanner(System.in);

		System.out.println("Enter the numbers");

		n1=s.nextDouble();
		n2=s.nextDouble();

		System.out.println("Enter the operator(+,-,*,/)");

		char op=s.next().charAt(0);
		double o=0;

		switch(op)
		{
		case '+':
			o=n1+n2;
			break;
		case '-':
			o=n1-n2;
			break;
```

```
                case '*':
                        o=n1*n2;
                        break;
                case '/':
                        o=n1/n2;
                        break;
                }
                System.out.println("The final output is");
                System.out.println();
                System.out.println(n1+" "+op+" "+n2+" = "+o);
        }

}
```

4.RETURN TYPES

```
package com.Assignments;
import java.util.*;

public class ReturnTypes4
{

        static void add()
        {
                int a,b;
                Scanner s=new Scanner(System.in);
                System.out.println("Enter the numbers");
                a=s.nextInt();
                b=s.nextInt();

                int c=a+b;
                System.out.println("Add method "+c);
        }
        static int addition()
        {
                int a,b;
                Scanner s=new Scanner(System.in);
                System.out.println("Enter the numbers");
                a=s.nextInt();
                b=s.nextInt();

                int c=a+b;
                return c;
        }
        static float adds()
        {
                float a,b;
                Scanner s=new Scanner(System.in);
                System.out.println("Enter the numbers");
                a=s.nextFloat();
                b=s.nextFloat();
                float c=a+b;
                return c;
        }
        static char returnchar()
        {
                return 'd';
        }
```

```
	public static void main(String[] args)
	{
		add();
		int addi=addition();
		System.out.println(addi);
		float addsmethod=adds();
		System.out.println(addsmethod);
		char d=returnchar();
		System.out.println(d);
	}
}
```

5.CONSTRUCTOR

```
package com.Assignments;
class calculate
{

	calculate()
	{
		System.out.println("zero argument constructor");
	}

	public static int calculate(int a,int b)
	{
	int  c=a+b;
	System.out.println("addition of two numbers"+ c);
	return c;
	}

	public static double calculate(int r)
	{
	 double circle=0.5*3.14*r*r;
	 System.out.println("area of circle"+ circle);
	 return circle;
	}

	 public static double calculate(double width,int length)
	 {
	 double rect=length*width;
	 System.out.println("area of rectangle"+rect);
	 return rect;
	 }
}

public class Constructor5
{
	public static void main(String[] args)
	{
	  calculate cal=new calculate();
	  cal.calculate(2,3);
	  cal.calculate(52);
	  cal.calculate(9.85,6);
	 }
}
```

6.COLLECTIONS

```
package com.Assignments;
import java.util.ArrayList;
import java.util.Collections;
class Collections6
{
	public static void main(String[] args){

		ArrayList<String> al=new ArrayList<String>();

		System.out.println("Initial size of Arraylist is "+ al.size());
// SIZE OF ARRAYLIST

		al.add("Hai");
		al.add("Hello");
		System.out.println("Arraylist is "+al);
		System.out.println("New size of Arraylist is "+al.size()); //
UPDATED SIZE OF ARRAYLIST



		al.add(1,"How are you");
		System.out.println("New Arraylist is "+al);
		System.out.println("New size of Arraylist is "+al.size());

		System.out.println("element at position 1 is  : "+ al.get(1));
// GET ELEMENT AT POSITION

		Collections.reverse(al);   //Reversing order of arraylist
		System.out.println("Reversed arraylist is : "+ al);    //
Printing arraylist after reversing

		al.set(2, "Bye"); //Setting Bye - Bye at position 6 (or)
Replacing Hello by Bye at position 2
		System.out.println("ArrayList after setting/replacing element at
position 2 is : " + al);

		al.remove(2);
		System.out.println("New Arraylist is "+al);

		al.remove("Hello");
		System.out.println("New Arraylist is "+al);

		System.out.println(al.contains("Hello"));



		}
	}
```

7.MAP

```
package com.Assignments;

import java.util.*;

public class Map7
{
	public static void main(String[] args) {
```

```
      //Hashmap
      HashMap hm = new HashMap();

    hm.put("1","1");
    hm.put("2","SECOND");
    hm.put("3","THIRD");
      hm.put("4",null);
      hm.put(1, 23);
      hm.put(null,"FIFTH");

    System.out.println("hashmap is "+ hm);
    System.out.println("Value of 3 key: "+ hm.get("3"));
    System.out.println("Is HashMap empty? "+hm.isEmpty());

    hm.remove("2");
    System.out.println("After removal process, the hashmap is "+hm);
    System.out.println("Size of the HashMap: "+hm.size());

    System.out.println(hm.containsValue("FIFTH"));


    //Linkedhashmap
    Map<Integer, String> linkedHashMap = new LinkedHashMap<Integer,
String>();

    linkedHashMap.put(1, new String("Samsung"));
    linkedHashMap.put(2, new String("Mi"));
    linkedHashMap.put(3, new String("Toshiba"));
    linkedHashMap.put(4, new String("HCL"));
    linkedHashMap.put(5, new String("Wipro"));


    System.out.println("Contents of LinkedHashMap : " + linkedHashMap);
    System.out.println("\nValues of map after iterating over it : ");

    for (Integer key : linkedHashMap.keySet()) {
        System.out.println(key + ":\t" + linkedHashMap.get(key));
    }

    System.out.println("\nThe size of the LinkedHashMap is : " +
linkedHashMap.size());
    System.out.println("\nIs LinkedHashMap empty? : " +
linkedHashMap.isEmpty());
    System.out.println("\nLinkedHashMap contains 2 as key? : " +
linkedHashMap.containsKey(2));
    System.out.println("LinkedHashMap contains HCL as value? : " +
linkedHashMap.containsValue("HCL"));
    System.out.println("\nRemove entry for key 3 : " +
linkedHashMap.remove(3));
    System.out.println("Content of LinkedHashMap after removing key 2: " +
linkedHashMap);

    linkedHashMap.clear();
    System.out.println("\nContent of LinkedHashMap after clearing: " +
linkedHashMap);


    //Treemap
    TreeMap<String, Integer> marks = new TreeMap<String, Integer>();
    marks.put("Student1", 120);
    marks.put("Student5", 99);
```

```
    marks.put("Student6", 130);
    marks.put("Student2", 190);
    marks.put("Student3", 89);
    marks.put("Student4", 142);

    for(String key: marks.keySet()){
            System.out.println(key  +" : "+ marks.get(key));
    }
}
}
```

7.INNER CLASS

```
package com.Assignments;



public class InnerClass7
{
private int data=30;

      void display(){
            System.out.println("i am inside the outer class method");
      }

      class Inner{
       private int data=20;

        void msg()
        {
              InnerClass7.this.display();
              System.out.println("data is "+data);
        }

        // calling the duplicate method of the outer class
        void display(){
              System.out.println("i am inside the inner class method");
        }// inner class accessing the outside private variable
        class Inner_2{

              void Inner()
              {
                    InnerClass7.this.display();
                    System.out.println("data is "+data);
              }
              void Inner2()
              {
                    System.out.println("2nd Inner class");
              }

        }
       }

      public static void main(String args[]){

        InnerClass7 obj=new InnerClass7();     // creating object of Outer
class

        InnerClass7.Inner in=obj.new Inner();
        Inner.Inner_2 i=in.new Inner_2();// creating object of Inner class
        in.msg();
        in.display();
```

```
        i.Inner();
        i.Inner2();
       }

}
```

8.STRING TO STRING BUFFER

```
package com.Assignments;
public class StringBuffer8{
      public static void main(String args[])
      {

            StringBuffer sb=new StringBuffer("Hello ");

            sb.append("Java");         //now original string is changed
            System.out.println(sb);    //prints Hello Java

            sb.insert(1,"Java");      //now original string is changed
            System.out.println(sb);

            sb.replace(1,3,"Java");
            System.out.println(sb);

            sb.delete(1,3);
            System.out.println(sb);

            sb.reverse();
            System.out.println(sb);

            System.out.println(sb.charAt(3));

            StringBuilder ss = new StringBuilder("Core");

            System.out.println(">>>>>>>"+ ss);
            ss.append("Java");
            System.out.println(">>>>>>>"+ ss);
            System.out.println(ss.length());
            System.out.println(ss.charAt(6));
            System.out.println(ss.reverse());
      }
}
```

9.ARRAYS

```
package com.Assignments;

public class Arrays9
{
      public static void main(String[] args)
    {
            //one dimension array
            int a[]= {7,5,32,5,41};
            for(int i=0;i<a.length;i++)
            {
                  System.out.print(a[i]);
                  System.out.print(" ");
```

```
            }
            System.out.println(" ");

            //Two dimensional array
        int[][] arr = { { 1, 2 }, { 3, 4 } };

        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {
                System.out.print(arr[i][j] + " ");
            }
            System.out.println();
        }
    }

}
```

10.STRINGS USING REGULAR EXPRESSIONS

```
package com.Assignments;
import java.util.regex.*;

public class Stringsusingregx10{

public static void main(String[] args) {

      String pattern = "[a-z]+";
      String check = "Regular Expressions";
      Pattern p = Pattern.compile(pattern);
      Matcher c = p.matcher(check);

      while (c.find())
            System.out.println( check.substring( c.start(), c.end() ) );
}
}
```

11.ARRAY OF STRINGS

```
package com.Assignments;

import java.util.*;

public class SearchaString11 {
      public static void main(String[] args)
      {
            String[] str= {"hema","mahesh","anu"};
            boolean found=false;
            int index=0;
            Scanner s=new Scanner(System.in);
            System.out.println("Enter the string");
            String a=s.nextLine();

            for(int i=0;i<str.length;i++)
            {
                  if(a.equals(str[i]))
                  {
                        index=i;
```

```
                        found=true;
                        break;
                    }
                }
                if(found)
                {
                    System.out.println(a+" found at the index "+index);
                }
                else
                {
                    System.out.println(a+" not found in the array");
                }
        }

}
```

12.THREAD CREATION MECHANISM

```
package com.Assignments;


public class Threads12 extends Thread
{
    public void run()
    {
        for(int i=1;i<=5;i++)
        {
           try
           {
                Thread.sleep(500);
           }
            catch(Exception e)
           {
                System.out.println(e);
           }
           System.out.println(i);
          }
       }

    public static void main(String args[])
    {

        Threads12 t1=new Threads12();
        Threads12 t2=new Threads12();
        Threads12 t3=new Threads12();
        t1.start();

        try
        {
            t1.join();
        }
        catch(Exception e)
        {
            System.out.println(e);
        }

        t2.start();
```

```
             t3.start();
             }
            }
```

```
package com.Assignments;

public class ThreadsRunnable12  implements Runnable
{

      public  void run()
      {
            System.out.println("thread is running...");
      }

      public static void main(String args[])
      {

            ThreadsRunnable12 m1=new ThreadsRunnable12();
            Thread t1 = new Thread(m1);
            t1.start();
      }
}
```

13.SLEEP_WAIT

```
package com.Assignments;

class Customer{
       int amount=10000;

            synchronized void withdraw(int amount){
            System.out.println("going to withdraw...");

                  if(this.amount<amount)
                  {
                  System.out.println("Less balance; waiting for
deposit...");
                  try{
                        wait();
                     }
                  catch(Exception e){}
                  }
                  this.amount-=amount;
                  System.out.println("withdraw completed...the lefft over
amount is"+ this.amount);
            }

            synchronized void deposit(int amount){
                  System.out.println("going to deposit...");
                  this.amount+=amount;

                  System.out.println("deposit completed... " +
this.amount);
                  notify();  //unlocking of thread
            }
       }
```

```
public class Sleep_Wait13 extends Thread
{
       public void run(){
              for(int i=1;i<5;i++){
                try{
                        Thread.sleep(2000);  // 2000 miliseconds = 2 secs
                  }
                catch(InterruptedException e)
                {
                  System.out.println(e);
                }
                System.out.println(Thread.currentThread().getName()+ "   :"+
i);
              }
             }



            public static void main(String args[]){
             Sleep_Wait13 t1=new  Sleep_Wait13 ();
             t1.setName("Java");
             t1.setPriority(MAX_PRIORITY);

             Sleep_Wait13  t2=new  Sleep_Wait13 ();
             t2.setName("Python");

             Sleep_Wait13  t3=new  Sleep_Wait13 ();
             t3.setName("Oracle");

             Sleep_Wait13 t4=new  Sleep_Wait13 ();
             t4.setName("C++");

             t1.start();
             t2.start();
             t3.start();
             t4.start();

             System.out.println(t1.getState());
             System.out.println(t3.getState());
             System.out.println(t2.getState());


             final Customer c=new Customer();

                              new Thread(){                  // anonymous
class
                                public void run()
                                {
                                    c.withdraw(5000);
                                }
                              }.start();
                              new Thread(){                  // anonymous
class
                                    public void run()
                                    {
                                        c.withdraw(4000);
                                    }
                                  }.start();
                              new Thread(){                  // anonymous
class
                                          public void run()
```

```
                                        {
                                            c.withdraw(10000);
                                        }
                                    }.start();

                                new Thread(){
                                  public void run()
                                   {
                                       c.deposit(20000);
                                   }
                                }.start();
                            }

                    }
```

14.MULTI THREADING

```
package com.Assignments;



public class Multithreading14 extends Thread
{

            public void run(){
                System.out.println("running thread name
is:"+Thread.currentThread().getName());  // name of the thread
                System.out.println("running thread priority is:"+
Thread.currentThread().getName()+ " :::" +
Thread.currentThread().getPriority());  //what is current prirotiy of
thread
                System.out.println("running thread state is:"+
Thread.currentThread().getName()+ " :::" +
Thread.currentThread().getState());  // it will print state of the thread
running
                System.out.println("The thread group is:" +
Thread.currentThread().getName()+ " :::" +
Thread.currentThread().getThreadGroup()); // a group in whihc thread is
assgined
                System.out.println("The thread id is:"+
Thread.currentThread().getName()+ " :::" + Thread.currentThread().getId());
// CPU scheduler has given a unique ID to each thread
                System.out.println("Is my thread alive or not? :  "+
Thread.currentThread().getName()+ " :::" +
Thread.currentThread().isAlive());  // it will heck if thread is alive or
dead
                System.out.println("Is my thread Daemon or not? :  "+
Thread.currentThread().getName()+ " :::" +
Thread.currentThread().isDaemon()); // this thread always run in background
            }

             public static void main(String args[]){

              Multithreading14 m1=new Multithreading14();
              Multithreading14 m2=new Multithreading14();
              Multithreading14 m3=new Multithreading14();

              m1.setName("Abc");
              m2.setName("XYZ");
              m3.setName("MNP");
```

```
            m1.setPriority(Thread.MIN_PRIORITY);
            m2.setPriority(Thread.MAX_PRIORITY);
            m3.setPriority(Thread.NORM_PRIORITY);

            m2.setDaemon(true);

            m1.start();
            m2.start();
            m3.start();

          }
        }
```

15.TRY CATCH

```
package com.Assignments;

public class Trycatch15 {
      public static void main(String[] args) {

        try{
             int a[]=new int[5];
             a[5]=30/0;
             a[6]=9;
            }
           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");
               }
        finally{
         System.out.println("rest of the code");
        }


  }

}
```

16.CUSTOM EXCEPTION

```
package com.Assignments;

public class CustomException16
{
      static void performance(int salary)throws PerformanceException
      {
```

```
	if (salary<2100)
		throw new PerformanceException("you need to work hard");
	else if(salary>2100 && salary<5000)
		throw new PerformanceException("your salary is somehow good");
	else if(salary>5100 && salary<9000)
		throw new PerformanceException("your salary is very good");
	else
		System.out.println("invalidamount");
}
public static void main(String[] args) {
	try {
		performance(5200);
	}
	catch(Exception e)
	{
		System.out.println("Exception occured:"+e.getMessage());
	}
	finally
	{
		System.out.println("finally block executed");
	}
}

}
class PerformanceException  extends Exception{
PerformanceException(String e){
	super(e);
}


}
```

17.EXCEPTION HANDLING

```
package com.Assignments;


public class ExceptionHandling17
{

	public static void main(String[] args){
		int num1,num2,num3;
		num1=20;
		num2=10;

		try{
		  num3 = num1/num2;
		  System.out.println("Result is "+num3);
		}
		catch(ArithmeticException ae){  // child
			System.out.println("Numbers cannot be divided by zero");
		}
		catch(Exception ae1) // parent
		{
			System.out.println("i am before the subclass exception");
		}
		finally
		{
			System.out.println(" this block will always executed");
		}
```

```
        num3=num1+num2;                 // normal flow after catching exception
        System.out.println("Result after addition is "+num3);
        }


}
```

18.OPERATIONS ON THE FILES

```
package com.Assignments;

import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class FileOperations18
{
      public static void main(String[] args)
      {
            //Create a file
             File file = new File("C:\\lms\\hello.txt");
              try
              {
                  if (file.createNewFile())
                  {
                      System.out.println("New File is created!");
                  }
                  else
                  {
                     if(file.exists())
                     {
                           System.out.println("File already exists.");
                           System.out.println("File path:" +
file.getAbsolutePath());
                           System.out.println("File name:   " +
file.getName());
                           System.out.println("File class:   " +
file.getClass());
                           System.out.println("File parent:   " +
file.getParent());
                           System.out.println("File space allocated:   " +
file.getUsableSpace());
                           System.out.println("File length: " +
file.length());
                           System.out.println("File list:    " +
file.list());

                     }
                  }
              } catch (IOException e) {
                  e.printStackTrace();
              }


               //Write operation
               String data = "This is the data in the output file";

                try {
```

```
                    FileWriter output = new FileWriter("C:\\lms\\hello.txt");


                    output.write(data);
                    System.out.println("Data is written to the file.");


                    output.close();
                  }
                  catch (Exception ex) {
                    ex.getStackTrace();
                  }

                  //Read operation
                  char[] array = new char[60];

                  try {

                    FileReader input = new FileReader("c:\\lms\\hello.txt");

                    input.read(array);

                    System.out.println("Data in the file:");
                    System.out.println(array);


                    input.close();
                  }
                  catch(Exception exc) {
                    exc.getStackTrace();
                  }
                  //Delete a file
                  boolean b = file.delete();
              if(b==true)
              {
                    System.out.println("File deleted !!");
              }
              else
              {
                    System.out.println("File not deleted");
              }

          }


}
```

19.OOPS

```
package com.Assignments;
public class Objects19
{
    String name;
    String breed;
    int age;
    String color;
    public Objects19(String name, String breed, int age, String color)
    {
        this.name = name;
        this.breed = breed;
```

```
        this.age = age;
        this.color = color;
    }
    public String getName()
    {
        return name;
    }
    public String getBreed()
    {
        return breed;
    }
    public int getAge()
    {
        return age;
    }
    public String getColor()
    {
        return color;
    }
    @Override
    public String toString()
    {
        return("Hi my name is "+ this.getName()+ ".\nMy breed,age and color
are " + this.getBreed()+", " + this.getAge()+ ", and"+ this.getColor() +
".");
    }
    public static void main(String[] args)
    {
        Objects19 scott = new Objects19("Scott","papillon", 5, "black");
        System.out.println(scott.toString());
    }
}
```

ENCAPSULATION

```
package com.Assignments;
class Encapsulate
{
    private String Name;
    private int Roll;
    private int Age;
    public int getAge()
    {
      return Age;
    }
    public String getName()
    {
      return Name;
    }
    public int getRoll()
    {
       return Roll;
    }
    public void setAge( int newAge)
    {
      Age = newAge;
    }
    public void setName(String newName)
    {
      Name = newName;
    }
```

```
    public void setRoll( int newRoll)
    {
      Roll = newRoll;
    }
}
public class Encapsulation19
{
    public static void main (String[] args)
    {
        Encapsulate obj = new Encapsulate();
        obj.setName("Harsh");
        obj.setAge(19);
        obj.setRoll(51);
        System.out.println("My name: " + obj.getName());
        System.out.println("My age: " + obj.getAge());
        System.out.println("My roll: " + obj.getRoll());
    }
}
```

ABSTRACTION

```
package com.Assignments;
abstract class Shape
{
    String color;
    abstract double area();
    public abstract String toString();
    public Shape(String color)
    {
        System.out.println("Shape constructor called");
        this.color = color;
    }
    public String getColor()
    {
        return color;
    }
}
class Circle extends Shape
{
    double radius;
    public Circle(String color,double radius)
    {
        super(color);
        System.out.println("Circle constructor called");
        this.radius = radius;
    }
    @Override
    double area()
    {
        return Math.PI * Math.pow(radius, 2);
    }
    @Override
    public String toString()
    {
        return "Circle color is " + super.color + "and area is : " +
area();
    }
}
class Rectangle extends Shape
{
    double length;
```

```
    double width;
    public Rectangle(String color,double length,double width)
    {
        super(color);
        System.out.println("Rectangle constructor called");
        this.length = length;
        this.width = width;
    }
    @Override
    double area()
    {
        return length*width;
    }
    @Override
    public String toString()
    {
        return "Rectangle color is " + super.color +
                            "and area is : " + area();
    }
}
public class Abstraction19
{
    public static void main(String[] args)
    {
        Shape s1 = new Circle("Red", 2.2);
        Shape s2 = new Rectangle("Yellow", 2, 4);
        System.out.println(s1.toString());
        System.out.println(s2.toString());
    }
}
```

INHERITANCE

```
package com.Assignments;
class Bicycle
{
    public int gear;
    public int speed;
    public Bicycle(int gear, int speed)
    {
        this.gear = gear;
        this.speed = speed;
    }
    public void applyBrake(int decrement)
    {
        speed -= decrement;
    }
    public void speedUp(int increment)
    {
        speed += increment;
    }
    public String toString()
    {
        return("No of gears are " + gear + "\n" + "speed of bicycle is " +
speed);
    }
}
class MountainBike extends Bicycle
{
    public int seatHeight;
```

```
    public MountainBike(int gear,int speed,int startHeight)
    {
        super(gear, speed);
        seatHeight = startHeight;
    }
    public void setHeight(int newValue)
    {
        seatHeight = newValue;
    }
    @Override
    public String toString()
    {
        return (super.toString()+
                "\nseat height is "+seatHeight);
    }
}
public class Inheritance19
{
    public static void main(String args[])
    {
        MountainBike mb = new MountainBike(3, 100, 25);
        System.out.println(mb.toString());
    }
}
```

POLYMORPHISM

```
package com.Assignments;
class Polymorphism19
{
    public int sum(int x, int y)
    {
        return (x + y);
    }
    public int sum(int x, int y, int z)
    {
        return (x + y + z);
    }
    public double sum(double x, double y)
    {
        return (x + y);
    }
    public static void main(String args[])
    {
        Polymorphism19 s = new Polymorphism19();
        System.out.println(s.sum(10, 20));
        System.out.println(s.sum(10, 20, 30));
        System.out.println(s.sum(10.5, 20.5));
    }
}
```

20.INTERFACE

```
package com.Assignments;

interface Drawable{
      int salary = 100;
      void draw();  //by default abstarct method() // public abstract void
draw();
      void show();
}
```

```
interface Shape{
	int salary = 200;
	void shape();  //by default abstarct method()  // public abstract
void shape();
	void show();
}

interface Demo extends Shape{
}

	public class Interfaces20 implements Shape, Drawable{

		int salary = 900;

		public void show(){
			System.out.println("inside show()");
		}

		@Override
		public void draw() {
			// TODO Auto-generated method stub
			System.out.println("inside draw");
		}

		@Override
		public void shape() {
			// TODO Auto-generated method stub
			System.out.println("inside shape");
		}

		public static void main(String args[]){
			Interfaces20 obj = new Interfaces20();
			obj.shape();
			obj.show();
			obj.draw();

			Shape d = new Interfaces20();
			System.out.println(d.salary);

			System.out.println("salary  "+ obj.salary);
		}
	}
```

21.FILES

```
package com.Assignments;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;
import java.nio.charset.Charset;

public class Files21 {
   public static void main(String[] args) {

	  //initialize Path object
	Path path = Paths.get("F:/Demo.txt");
```

```
        //create file
        try {
           Path createdFilePath = Files.createFile(path);
           System.out.println("Created a file at : "+createdFilePath);
        }
        catch (IOException e) {
           e.printStackTrace();
        }

        //--------------------Write into the File---------------------//

        Path pathw = Paths.get("F:/Demo.txt");
        String question = "To be or not to be?";
        Charset charset = Charset.forName("ISO-8859-1");
        try {
           Files.write(pathw, question.getBytes());
           List<String> lines = Files.readAllLines(pathw, charset);
           for (String line : lines) {
              System.out.println(line);
           }
        }
        catch (IOException e) {
           System.out.println(e);
        }

    }
}
```

22.ARRAY ROTATION

```
package com.Assignments;

public class ArrayRotation22
{
      public static void main(String[] args)
      {
            int[] arr= {1,2,3,4,5};
            int n=3;
            System.out.println("original array:");

            for(int i=0;i<arr.length;i++)
            {
                  System.out.println(arr[i]+" ");
            }

            for (int i=0;i<n;i++)
            {
                  int j,last;
                  last=arr[arr.length-1];

                  for(j=arr.length-1;j>0;j--) {
                        arr[j]=arr[j-1];

                  }
                  arr[0]=last;
            }
            System.out.println( );

            System.out.println("array after rotation");
```

```
            for(int  i=0;i<arr.length;i++) {
                  System.out.println(arr[i]+" ");
            }
      }
}
```

23.ORDER STATISTICS

```
package com.Assignments;


      class KthSmallst
      {
      int kthSmallest(int arr[], int l, int r, int k)
            {
                        if (k > 0 && k <= r - l + 1)
                  {
                              int pos = randomPartition(arr, l, r);
                              if (pos-l == k-1)
                              return arr[pos];
                              if (pos-l > k-1)
                              return kthSmallest(arr, l, pos-1, k);
                              return kthSmallest(arr, pos+1, r, k-
pos+l-1);
                  }
               return Integer.MAX_VALUE;
          }
          void swap(int arr[], int i, int j)
          {
              int temp = arr[i];
              arr[i] = arr[j];
              arr[j] = temp;
          }
          int partition(int arr[], int l, int r)
          {
              int x = arr[r], i = l;
              for (int j = l; j <= r - 1; j++)
              {
                  if (arr[j] <= x)
                  {
                      swap(arr, i, j);
                      i++;
                  }
              }
              swap(arr, i, r);
              return i;
          }
          int randomPartition(int arr[], int l, int r)
          {
              int n = r-l+1;
              int pivot = (int)(Math.random()) * (n-1);
              swap(arr, l + pivot, r);
              return partition(arr, l, r);
          }
      }
      public class OrderStatistics23
      {
            public static void main(String[] args) {
                  KthSmallst ob = new KthSmallst();
              int arr[] = {12, 3, 5, 7, 4, 19, 26};
```

```
            int n = arr.length,k = 4;
            System.out.println("K'th smallest element is "+
ob.kthSmallest(arr, 0, n-1, k));
        }
    }
```

24.RANGE QUERIES

```
package com.Assignments;

public class RangeQueries24
{

        static int k = 16;
        static int N = 100000;
        static long table[][] = new long[N][k + 1];
        static void buildSparseTable(int arr[], int n)
        {
            for (int i = 0; i < n; i++)
                table[i][0] = arr[i];
            for (int j = 1; j <= k; j++)
                for (int i = 0; i <= n - (1 << j); i++)
                    table[i][j] = table[i][j - 1] + table[i + (1 << (j -
1))][j - 1];
        }
        static long query(int L, int R)
        {
            long answer = 0;
            for (int j = k; j >= 0; j--)
            {
                if (L + (1 << j) - 1 <= R)
                {
                    answer = answer + table[L][j];
                    L += 1 << j;
                }
            }
            return answer;
        }
        public static void main(String args[])
        {
            int arr[] = { 3, 7, 2, 5, 8, 9 };
            int n = arr.length;
            buildSparseTable(arr, n);
            System.out.println(query(0, 5));
            System.out.println(query(3, 5));
            System.out.println(query(2, 4));
        }
    }
```

25.MATRICES

```
package com.Assignments;

public class Matrices25
{


    public static int[][] multiplyMatrices(int[][] firstMatrix, int[][]
secondMatrix, int r1, int c1, int c2)
```

```
        {
             int[][] product = new int[r1][c2];
             for(int i = 0; i < r1; i++)
             {
                for (int j = 0; j < c2; j++)
                {
                  for (int k = 0; k < c1; k++)
                  {
                     product[i][j] += firstMatrix[i][k] *
secondMatrix[k][j];
                  }
                }
             }
             return product;
        }
         public static void displayProduct(int[][] product)
         {
             System.out.println("Product of two matrices is: ");
             for(int[] row : product)
             {
               for (int column : row)
               {
                    System.out.print(column + "    ");
                }
                System.out.println();
              }
          }
         public static void main(String[] args)
             {
                   int r1 = 2, c1 = 3;
                   int r2 = 3, c2 = 2;
                   int[][] firstMatrix = { {3, -2, 5}, {3, 0, 4} };
                   int[][] secondMatrix = { {2, 3}, {-9, 0}, {0, 4} };
                   int[][] product = multiplyMatrices(firstMatrix,
secondMatrix, r1, c1, c2);
                   displayProduct(product);
              }
       }
```

26.SINGLY LINKED LIST

```
package com.Assignments;


public class SinglyLinkedList26
{

         class Node{
               int data;
               Node next;

               public Node(int data) {
                   this.data = data;
                   this.next = null;
               }
           }


           public Node head = null;
           public Node tail = null;
```

```
    public void addNode(int data) {

      Node newNode = new Node(data);


        if(head == null) {

            head = newNode;
            tail = newNode;
        }
        else {

            tail.next = newNode;

            tail = newNode;
        }
    }


    public void display() {

        Node current = head;

        if(head == null) {
            System.out.println("List is empty");
            return;
        }
        System.out.println("Nodes of singly linked list: ");
        while(current != null) {

            System.out.print(current.data + " ");
            current = current.next;
        }
        System.out.println();
    }

    public static void main(String[] args) {

        SinglyLinkedList26 sList = new SinglyLinkedList26();

        //Add nodes to the list
        sList.addNode(1);
        sList.addNode(2);
        sList.addNode(3);
        sList.addNode(4);

        //Displays the nodes present in the list
        sList.display();
    }
}
```

27.CIRCULAR LINKED LIST

```
package com.Assignments;
```

```
public class CircularLinkedList27
{
      static class Node
      {
          int data;
          Node next;
      };

      static Node addToEmpty(Node last, int data)
      {

          if (last != null)
          return last;
          Node temp = new Node();

          temp.data = data;
          last = temp;

       last.next = last;

          return last;
      }

      static Node addBegin(Node last, int data)
      {
          if (last == null)
          return addToEmpty(last, data);

          Node temp = new Node();

          temp.data = data;
          temp.next = last.next;
          last.next = temp;

          return last;
      }

      static Node addEnd(Node last, int data)
      {
          if (last == null)
              return addToEmpty(last, data);

          Node temp = new Node();

          temp.data = data;
          temp.next = last.next;
          last.next = temp;
          last = temp;

          return last;
      }

      static Node addAfter(Node last, int data, int item)
      {
          if (last == null)
              return null;

          Node temp, p;
          p = last.next;
          do
```

```
    {
        if (p.data == item)
        {
            temp = new Node();
            temp.data = data;
            temp.next = p.next;
            p.next = temp;

            if (p == last)
                last = temp;
            return last;
        }
        p = p.next;
    } while(p != last.next);

    System.out.println(item + " not present in the list.");
    return last;

}

static void traverse(Node last)
{
    Node p;


    if (last == null)
    {
        System.out.println("List is empty.");
        return;
    }


    p = last.next;

    do
    {
        System.out.print(p.data + " ");
        p = p.next;

    }
    while(p != last.next);

}

public static void main(String[] args)
{
    Node last = null;

    last = addToEmpty(last, 6);
    last = addBegin(last, 4);
    last = addBegin(last, 2);
    last = addEnd(last, 8);
    last = addEnd(last, 12);
    last = addAfter(last, 10, 8);

    traverse(last);
}
}
```

28.DOUBLYLINKEDLIST

```
package com.Assignments;


public class DoublyLinkedList28
{
      class Node{
        int data;
        Node previous;
        Node next;

        public Node(int data) {
            this.data = data;
        }
    }


    Node head, tail = null;

        public void addNode(int data) {

         Node newNode = new Node(data);


        if(head == null) {

            head = tail = newNode;

            head.previous = null;

            tail.next = null;
        }
        else {

            tail.next = newNode;

            newNode.previous = tail;

            tail = newNode;

            tail.next = null;
        }
    }


    public void display() {

        Node current = head;
        if(head == null) {
            System.out.println("List is empty");
            return;
        }
        System.out.println("Nodes of doubly linked list: ");
        while(current != null) {


            System.out.print(current.data + " ");
            current = current.next;
        }
    }
```

```
    public static void main(String[] args) {

        DoublyLinkedList28 dList = new DoublyLinkedList28();

        dList.addNode(1);
        dList.addNode(2);
        dList.addNode(3);
        dList.addNode(4);
        dList.addNode(5);


        dList.display();
    }
}
```

29.STACK

```
package com.Assignments;

public class Stack29
{
	static final int MAX = 1000;
	int top;
	int a[] = new int[MAX];
	boolean isEmpty()
	{
		return (top < 0);
	}
	Stack29()
	{
		top = -1;
	}
	boolean push(int x)
	{
		if (top >= (MAX-1))
		{
				System.out.println("Stack Overflow");
				return false;
		}
		else
		{
				a[++top] = x;
				System.out.println(x + " pushed into stack");
				return true;
		}
	}
	int pop()
	{
		if (top < 0)
		{
				System.out.println("Stack Underflow");
				return 0;
		}
		else
		{
				int x = a[top--];
				return x;
		}
	}
```

```
	public static void main(String args[])
	{
		Stack29 s = new Stack29();
		s.push(10);
		s.push(20);
		s.push(30);
		System.out.println(s.pop() + " Popped from stack");
	}
}
```

30.QUEUE

```
package com.Assignments;

import java.util.LinkedList;
import java.util.Queue;

public class Queue30
{
	public static void main(String[] args)
	{
		Queue <String> locationsQueue = new LinkedList<>();
		locationsQueue.add("Kolkata");
	    locationsQueue.add("Patna");
	    locationsQueue.add("Delhi");
	    locationsQueue.add("Gurgaon");
	    locationsQueue.add("Noida");
	    System.out.println("Queue is : " + locationsQueue);
	    System.out.println("Head of Queue : " + locationsQueue.peek());
	    locationsQueue.remove();
	    System.out.println("After removing Head of Queue : " +
locationsQueue);
	    System.out.println("Size of Queue : " + locationsQueue.size());
	 }


}
```

31.LONGEST INCREASE SEQUENCE

```
package com.Assignments;
class LIS {
	static int max_ref; // stores the LIS

	static int _lis(int arr[], int n)
	{

		if (n == 1)
			return 1;

		int res, max_ending_here = 1;


		for (int i = 1; i < n; i++) {
			res = _lis(arr, i);
```

```
                if (arr[i - 1] < arr[n - 1]
                      && res + 1 > max_ending_here)
                      max_ending_here = res + 1;
          }


          if (max_ref < max_ending_here)
                max_ref = max_ending_here;


          return max_ending_here;
    }


    static int lis(int arr[], int n)
    {

          max_ref = 1;


          _lis(arr, n);


          return max_ref;
    }


    public static void main(String args[])
    {
          int arr[] = { 10, 22, 9, 33, 21, 50, 41, 60 };
          int n = arr.length;
          System.out.println("Length of lis is " + lis(arr, n)
                                      + "\n");
    }
}
```

32.LINEAR SEARCH

```
package com.Assignments;

import java.util.*;

public class LinearSearch32
{

     public static void main(String[] args){

          int[] arr = {10,20,30,40,50};

          Scanner sc = new Scanner(System.in);
          System.out.println("Enter the element to be searched");
          int searchValue = sc.nextInt();

          int result = linearing(arr,searchValue);
          if(result==-1){

                System.out.println("Element not in the array");
            } else {
```

```
                        System.out.println("Element found at "+result+" and
the search key is "+arr[result]);
                    }
                }

              public static int linearing(int arr[], int x) {

            int arrlength = arr.length;
            for (int i = 0; i < arrlength - 1; i++) {

              if (arr[i] == x) {
                    return i;
                }
             }
                    return -1;
           }
      }
```

33.BINARY SEARCH

```
package com.Assignments;

public class BinarySearch33
{
       public static  void main(String[] args){

              int[] arr = {3,9,23,45,56,59,67,78,81,90,93,95};
              int key = 78;
              int arrlength = (arr.length-1);
              binarySearch(arr,0,key,arrlength);
          }

             public static void binarySearch(int[] arr, int start, int key,
int length){

              int midValue = (start+length)/2;
              while(start<=length){

                  if(arr[midValue]<key){
                      start = midValue + 1;
                  } else if(arr[midValue]==key){
                      System.out.println("Element is found at index
:"+midValue);
                      break;
                  }else {

                      length=midValue-1;
                  }
                  midValue = (start+length)/2;
              }
                  if(start>length){

                      System.out.println("Element is not found");
                  }
          }
      }
```

34.EXPONENTIAL SEARCH

```
package com.Assignments;

import java.util.Arrays;

public class ExponentialSearch34
{
	public static  void main(String[] args){

		int[] arr = {6,12,18,24,32};
		int length= (arr.length-1);
		int value = 18;
		int outcome = exponentialSearch(arr,length,value);

		if(outcome<0){
		   System.out.println( "Element is not present in the array");
		}else {
		    System.out.println( "Element is  present in the array at
index :"+outcome);
		    }
		}

	public static int exponentialSearch(int[] arr ,int length, int value
){

		    if(arr[0]==value){
		        return 0;
		        }
		    int i=1;
		    while(i<length && arr[i]<=value){

		        i=i*2;
		    }
		    return Arrays.binarySearch(arr,i/2,Math.min(i,length),value);
		    }
	}
```

35.SELECTION SORT

```
package com.Assignments;

public class SelectionSort35
{
	public static void main(String[] args) {

		int[] arr = {9,6,3,1,2,4,5};
		selectionSort(arr);
		System.out.println("The sorted elements are:");
		for(int i:arr){
		    System.out.println(i);
		     }
		 }

		public static void selectionSort(int[] arr){

		    for(int i=0;i<arr.length;i++){
```

```
                int index =i;
                for(int j=i+1;j<arr.length;j++){
                    if(arr[j]<arr[index]){

                        index =j;
                    }
                }
                int smallNumber = arr[index];
                arr[index]= arr[i];
                arr[i]= smallNumber;
            }
        }
    }
```

36.BUBBLE SORT

```
package com.Assignments;

public class BubbleSort36
{
    public static void main(String[] args){

        int[] arr= {25,20,15,5,10};
        bubbleSort(arr);
        for(int i=0;i<arr.length;i++){

            System.out.println(arr[i]);
            }
        }

       public static void bubbleSort(int[] arr){
            int len = arr.length;
            int temp = 0;
            for(int i=0;i<len;i++){
                for (int j=1;j<(len);j++){
                    if(arr[j-1]>arr[j]){
                    temp = arr[j-1];
                    arr[j-1]= arr[j];
                    arr[j]= temp;
                    }
                }
            }
        }
    }
```

37.INSERTION SORT

```
package com.Assignments;

public class InsertionSort37
{
    public static  void main(String[] args){

      int[] arr = {9,12,3,21,44,5,1,10,25};
      insertionSort(arr);
      for(int i=0;i<arr.length;i++){
          System.out.println(arr[i]);
      }
```

```
     }

    public static void insertionSort(int[] arr){

    int len = arr.length;
    for(int j=1;j<len;j++){
    int key = arr[j];
    int i=j-1;
    while ((i>-1) && (arr[i]>key)){

        arr[i+1]=arr[i];
        i--;
    }
    arr[i+1]=key;
         }
    }
}
```

38.MERGE SORT

```
package com.Assignments;



public class MergeSort38
{
      void merge(int arr[], int l, int m, int r)
    {
        int n1 = m - l + 1;
        int n2 = r - m;


        int L[] = new int [n1];
        int R[] = new int [n2];


        for (int i=0; i<n1; ++i)
            L[i] = arr[l + i];

        for (int j=0; j<n2; ++j)
            R[j] = arr[m + 1+ j];

        int i = 0, j = 0;

        int k = l;
        while (i < n1 && j < n2)
        {
            if (L[i] <= R[j])
            {
                arr[k] = L[i];
                i++;
            }
            else
            {
                arr[k] = R[j];
                j++;
            }
            k++;
        }
        while (i < n1)
```

```
        {
            arr[k] = L[i];
            i++;
            k++;
        }
        while (j < n2)
        {
            arr[k] = R[j];
            j++;
            k++;
        }
    }

    void sort(int arr[], int l, int r)
    {
        if (l < r)
        {
            int m = (l+r)/2;

            sort(arr, l, m);
            sort(arr , m+1, r);
            merge(arr, l, m, r);
        }
    }

    static void printArray(int arr[])
    {
        int n = arr.length;
        for (int i=0; i<n; ++i)
            System.out.print(arr[i] + " ");
        System.out.println();
    }

    public static void main(String args[])
    {
        int arr[] = {12, 11, 13, 5, 6, 7};

        System.out.println("Given Array");
        printArray(arr);

        MergeSort38 ob = new MergeSort38();
        ob.sort(arr, 0, arr.length-1);

        System.out.println("\nSorted array");
        printArray(arr);
    }
}
```

39.QUICK SORT

```
package com.Assignments;



public class QuickSort39
{
      int partition(int arr[], int low, int high)
      {
            int pivot = arr[high];
```

```
            int i = (low-1);
            for (int j=low; j<high; j++)
            {
                  if (arr[j] <= pivot)
                  {
                        i++;


                        int temp = arr[i];
                        arr[i] = arr[j];
                        arr[j] = temp;
                  }
            }


            int temp = arr[i+1];
            arr[i+1] = arr[high];
            arr[high] = temp;

            return i+1;
      }


      void sort(int arr[], int low, int high)
      {
            if (low < high)
            {

                  int pi = partition(arr, low, high);


                  sort(arr, low, pi-1);
                  sort(arr, pi+1, high);
            }
      }


      static void printArray(int arr[])
      {
            int n = arr.length;
            for (int i=0; i<n; ++i)
                  System.out.print(arr[i]+" ");
            System.out.println();
      }


      public static void main(String args[])
      {
            int arr[] = {10, 7, 8, 9, 1, 5};
            int n = arr.length;

            QuickSort39 ob = new QuickSort39();
            ob.sort(arr, 0, n-1);

            System.out.println("sorted array");
            printArray(arr);
      }
}
```

40.FIX BUGS

```
package com.Assignments;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.Scanner;

public class BugFix40{
      public static void main(String[] args) {
        //System.out.println("Hello World!");/
        System.out.println("\n**************\n");
        System.out.println("\tWelcome to TheDesk \n");
        System.out.println("**************");
        optionsSelection();

    }
    private static void optionsSelection() {
        String[] arr = {"1. I wish to review my expenditure",
                "2. I wish to add my expenditure",
                "3. I wish to delete my expenditure",
                "4. I wish to sort the expenditures",
                "5. I wish to search for a particular expenditure",
                "6. Close the application"
        };
        int[] arr1 = {1,2,3,4,5,6};
        int  slen = arr1.length;
        for(int i=0; i<slen;i++){
            System.out.println(arr[i]);
            // display the all the Strings mentioned in the String array
        }
        ArrayList<Integer> arrlist = new ArrayList<Integer>();
        ArrayList<Integer> expenses = new ArrayList<Integer>();
        expenses.add(1000);
        expenses.add(2300);
        expenses.add(45000);
        expenses.add(32000);
        expenses.add(110);
        expenses.addAll(arrlist);
        System.out.println("\nEnter your choice:\t");
        Scanner sc = new Scanner(System.in);
        int  options =  sc.nextInt();
        for(int j=1;j<=slen;j++){
            if(options==j){
                switch (options){
                    case 1:
                        System.out.println("Your saved expenses are listed
below: \n");
                        System.out.println(expenses+"\n");
                        optionsSelection();
                        break;
                    case 2:
                        System.out.println("Enter the value to add your
Expense: \n");
                        int value = sc.nextInt();
                        expenses.add(value);
                        System.out.println("Your value is updated\n");
                        expenses.addAll(arrlist);
                        System.out.println(expenses+"\n");
                        optionsSelection();

                        break;
                    case 3:
```

```
                        System.out.println("You are about the delete all
your expenses! \nConfirm again by selecting the same option...\n");
                        int con_choice = sc.nextInt();
                        if(con_choice==options){
                               expenses.clear();
                            System.out.println(expenses+"\n");
                            System.out.println("All your expenses are
erased!\n");
                        } else {
                            System.out.println("Oops... try again!");
                        }
                        optionsSelection();
                        break;
                    case 4:
                        sortExpenses(expenses);
                        optionsSelection();
                        break;
                    case 5:
                        searchExpenses(expenses);
                        optionsSelection();
                        break;
                    case 6:
                        closeApp();
                        break;
                    default:
                        System.out.println("You have made an invalid
choice!");
                        break;
                }
            }
        }

    }
    private static void closeApp() {
        System.out.println("Closing your application... \nThank you!");
            }
    private static void searchExpenses(ArrayList<Integer> arrayList) {
        int leng = arrayList.size();
        System.out.println("Enter the expense you need to search:\t");
        //
        Scanner sc = new Scanner(System.in);
        int input = sc.nextInt();
        //Linear Search
        for(int i=0;i<leng;i++) {
            if(arrayList.get(i)==input) {
                  System.out.println("Found the expense " + input + " at "
+ i + " position");
            }
        }
    }
    private static void sortExpenses(ArrayList<Integer> arrayList) {
        int arrlength =  arrayList.size();
       //Complete the method. The expenses should be sorted in ascending
order.

        Collections.sort(arrayList);
        System.out.println("Sorted expenses: ");
        for(Integer i: arrayList) {
            System.out.print(i + " ");
        }
```

```
            System.out.println("\n");

        }
    }
```