**/\*6. Write a Java program that creates an object and initializes its data members using constructor. Use constructor overloading concept\*/**

class Rectangle

{

 int length ;

 int breadth ;

 Rectangle()

 {

 System.out.println("Constructor with Zero Parameter Called ");

 length = breadth = 0 ;

 }

 Rectangle(int side)

 {

 System.out.println("Constructor with One Parameter Called");

 length = breadth = side ;

 }

 Rectangle(int l,int b)

 {

 System.out.println("Constructor with Two Parameters Called");

 length = l ;

 breadth = b ;

 }

 int area()

 {

 return (length \* breadth) ;

 }

}

class ConstructorOverloading

{

 public static void main(String[] args)

 {

 Rectangle r1 = new Rectangle(); //const. with 0-parameter called

 Rectangle r2 = new Rectangle(5); //const with l parameter called

 Rectangle r3 = new Rectangle(7,8); //const.with2 parameter called

 System.out.println("Area of First Rectangle is : "+ r1.area( ));

 System.out. println("Area of Square is : "+ r2.area( ));

 System.out.println("Area of Second Rectangle is : "+ r3.area( ));

 }

}

**/\*7. Write a Java Program to implement Wrapper classes and their methods.\*/**

import java.io.\*;

import java.io.DataInputStream;

import java.lang.\*;

class wrapperdemo

{

public static void main(String args[])

{

Float P=new Float(0);

Float I=new Float(0);

int y=0;

try

{

DataInputStream ds=new DataInputStream(System.in);

System.out.println("ENTER THE PRINCIPAL AMOUNT");

System.out.flush();

String sp=ds.readLine();

P=Float.valueOf(sp);

System.out.println("ENTER THE INTEREST RATE");

System.out.flush();

String SI=ds.readLine();

I=Float.valueOf(SI);

System.out.println("ENTER THE NUMBER OF YEARS");

System.out.flush();

String sy=ds.readLine();

y=Integer.parseInt(sy);

}

catch(Exception e)

{

System.out.println("INPUT OUTPUT ERROR");

System.exit(1);

}

Float value=loan(P.floatValue(),I.floatValue(),y);

System.out.println("FINAL VALUE IS:"+value);

}

static float loan(float P,float I,int y)

{

int year=1;

float sum=P;

while(year<=y)

{sum=sum+(P\*I)/100;year++;}

return sum;}}

**/\*8. Write a Java Program to illustrate function overloading concept.\*/**

class OverloadTest8

{

 void sum (int a, int b)

 {

 System.out.println ("The sum of integer: "+(a+b));

 }

 void sum (double a, double b)

 {

 System.out.println ("The sum of double: "+(a+b));

 }

 void sum (int a, double b)

 {

 System.out.println ("The sum of int and double: "+(a+b));

 }

 void sum (String a, String b)

 {

 System.out.println ("The sum of String: "+(a+b));

 }

 public static void main(String args[])

 {

 OverloadTest8 over = new OverloadTest8();

 over.sum(20,35);

 over.sum(21.3,18.7);

 over.sum(17, 24.6);

 over.sum("Overloading", "Info");

 }

}

**/\* 9. Write a Java program implement basic queue operations using class concept.\*/**

import java.io.\*;

class QueueArr

{

 static int i,front,rear,item,max=5,ch;

 static int a[]=new int[5];

 QueueArr()

 {

 front=-1;

 rear=-1;

 }

 public static void main(String args[])throws IOException

 {

 while((boolean)true)

 {

 try

 {

 System.out.println("Select Option 1.insert 2.delete 3.display 4.Exit");

 BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

 ch=Integer.parseInt(br.readLine());

 }

 catch(Exception e)

 { }

 if(ch==4)

 break;

 else

 {

 switch(ch)

 {

 case 1:

 insert();

 break;

 case 2:

 delete();

 break;

 case 3:

 display();

 break;

 }

 }

 }

 }

 static void insert()

 {

 if(rear>=max)

 {

 System.out.println("Queue is Full");

 }

 else

 {

 try

 {

 BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

 System.out.println("Enter the Element: ");

 item=Integer.parseInt(br.readLine());

 }

 catch(Exception e)

 {}

 rear=rear+1;

 a[rear]=item;

 }

 }

 static void delete()

 {

 if(front==-1)

 {

 System.out.println("Queue is Empty");

 }

 else

 {

 front=front+1;

 item=a[front];

 System.out.println("Deleted Item: "+item);

 }

 }

 static void display()

 {

 System.out.println("Elements in the Queue are:");

 for(int i=front+1; i<=rear; i++)

 {

 System.out.println(a[i]);

 }

 }

}

**/\*10. Write a Java program implement basic stack operations using class concept.\*/**

import java.io.\*;

class StackArr

{

 static int max=10,i,top,ch,item;

 static int a[]=new int[10];

 StackArr()

 {

 top=-1;

 }

 public static void main(String args[])throws IOException

 {

 while((boolean)true)

 {

 System.out.println("enter 1.Push 2.Pop 3.Display 4.Exit");

 try

 {

 BufferedReader br=new BufferedReader(newInputStreamReader(System.in));

 ch=Integer.parseInt(br.readLine());

 }

 catch(Exception e) { }

 if(ch==4)

 break;

 else

 {

 switch(ch)

 {

 case 1:

 push();

 break;

 case 2:

 pop();

 break;

 case 3:

 display();

 break;

 }

 }

 }

 }

 static void push()

 {

 if(top==max)

 System.out.println("stack is full");

 else

 try

 {

 BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

 System.out.println("enter the element:");

 item=Integer.parseInt(br.readLine());

 a[++top]=item;

 }

 catch(Exception e) { }

 }

 static void pop()

 {

 if(top==-1)

 System.out.println("stack is empty");

 else

 top--;

 System.out.println("poped item:"+a[top]);

 }

 static void display()

 {

 System.out.println("elements in stack are:");

 for(i=top; i>0; i--)

 System.out.println(a[i]);

 }

}