**1. Write a program to convert a given Decimal number to Binary, Octal and Hexadecimal using recursive functions.**

import java.util.Scanner;

public class Convert

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

System.out.println("Enter a no:");

int n = sc.nextInt();

System.out.println("Decimal to Binary is: ");

DecimalToBinary(n);

System.out.println("\nDecimal to Octal:");

DecimalToOctal(n);

System.out.println("\nDecimal to Hexadecimal:");

DecimalToHexadecimal(n);

System.out.println("");

}

public static void DecimalToBinary(int n)

{

if (n > 0)

{

DecimalToBinary(n / 2);

System.out.printf("%d", n % 2);

}

}

public static void DecimalToOctal(int n)

{

if (n > 0)

{

DecimalToOctal(n / 8);

System.out.printf("%d", n % 8);

}

}

public static void DecimalToHexadecimal(int n)

{

if(n>0)

{

DecimalToHexadecimal(n/16);

Display(n%16);

}

}

public static void Display(int r)

{

if(r<=9)

{

System.out.printf("%d",r);

}

else

{

for(int i=10;i<16;i++)

{ if(r==i)

{ System.out.printf("%c",(i+55));

break;

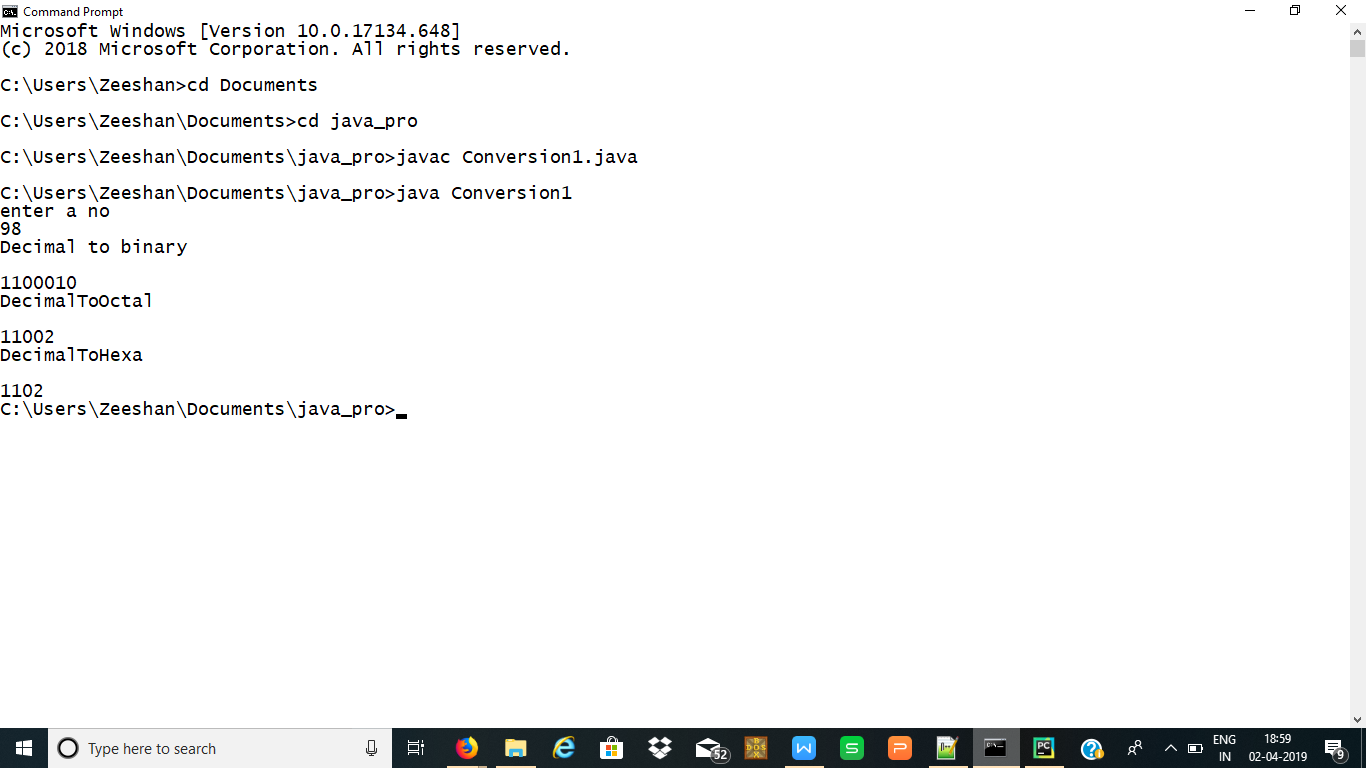
}

}

}

}

}



**2. Write a program to explain the concept of constructor overloading.**

import java.util.\*;

class Student

{

int Roll;

String Name;

double Marks;

Student(int R,String N,double M) // Constructor 1

{

Roll = R;

Name = N;

Marks = M;

}

Student(String N,double M,int R) // Constructor 2

{

Roll = R;

Name = N;

Marks = M;

}

void Display()

{

System.out.print("\n\t" + Roll+"\t" + Name+"\t" + Marks);

}

}

class ConstructorOverloadingDemo

{

public static void main(String[] args)

{

Student S1 = new Student(2,”faiz",75.);

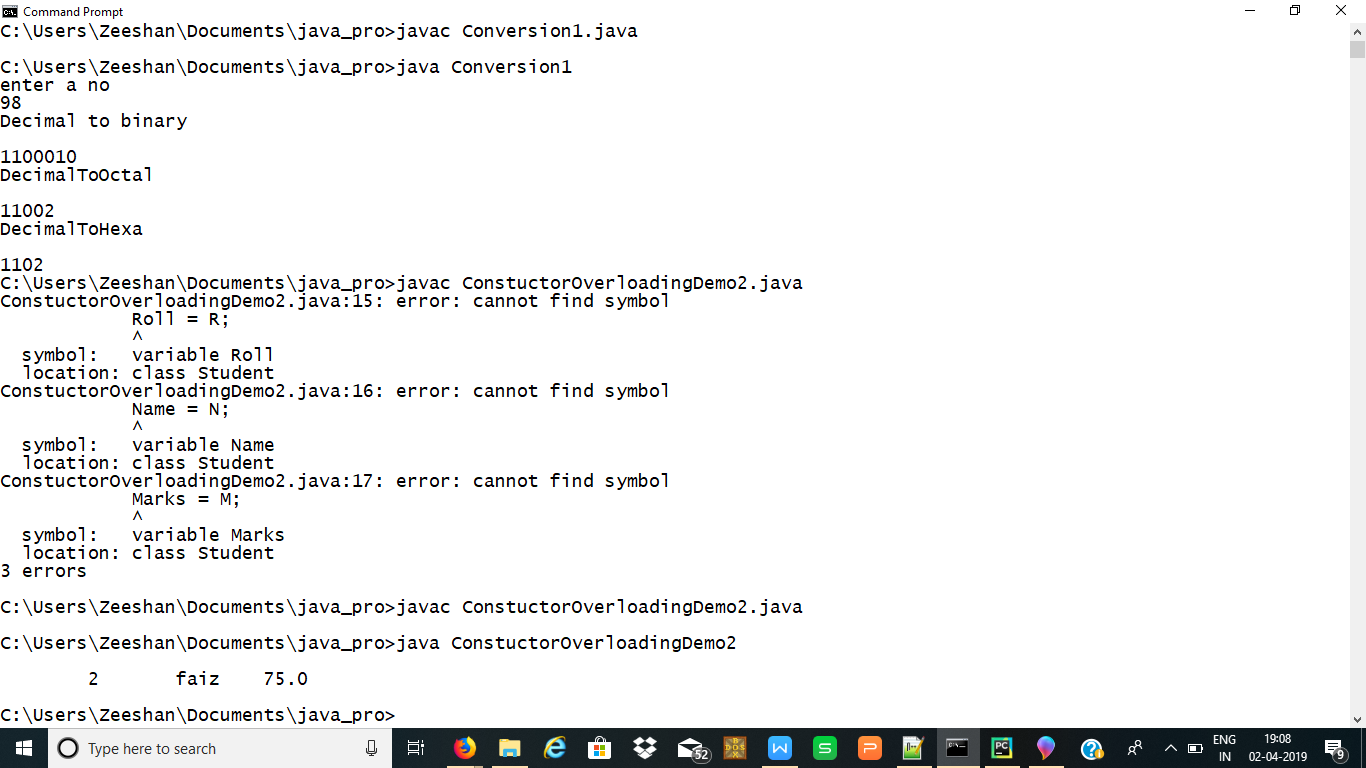
System.out.print("\n\tRoll\tName\tMarks\n");

S1.Display();

S2.Display();

}

}



1. **Explain the concept of passing objects as parameters by adding two distances given in feet and inches.**

**import java.util.\*;**

**class Distance**

**{**

**int feet,inches;**

**Distance(int f, int i)**

**{**

**feet=f;**

**inches=i;**

**}**

**void Calculate(Distance d)**

**{**

**d.feet= d.feet+(d.inches/12);**

**d.inches=d.inches%12;**

**System.out.println("feet" +d.feet +"inches" + d.inches);**

**}**

**}**

**class ObjectAsParameter**

**{**

**public static void main(String args[])**

**{**

**int feet,inches,feet1,feet2,inches1,inches2;**

**Scanner sc = new Scanner(System.in);**

**System.out.println("enter feet 1");**

**feet1=sc.nextInt();**

**System.out.println("enter feet 2");**

**feet2=sc.nextInt();**

**System.out.println("enter inch 1");**

**inches1=sc.nextInt();**

**System.out.println("enter inch 2");**

**inches2=sc.nextInt();**

**feet=feet1+ feet2;**

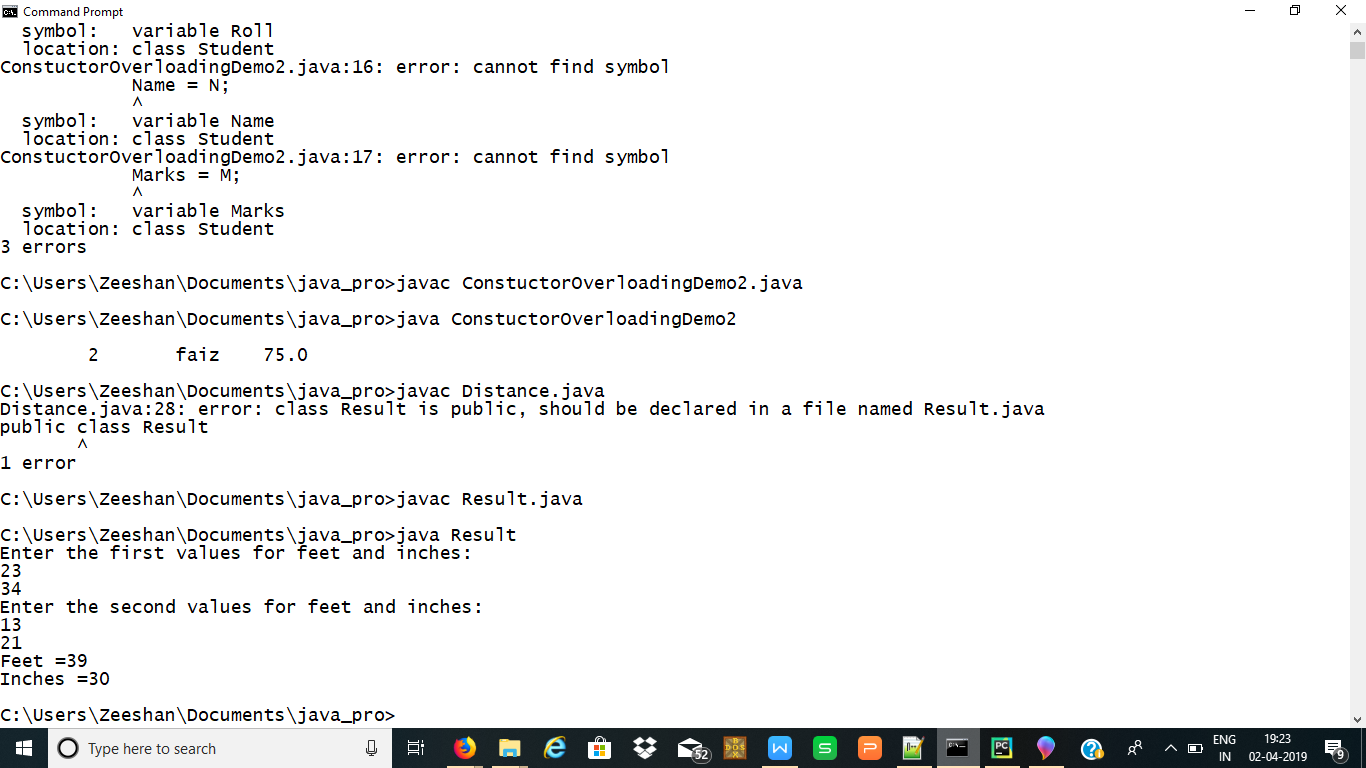
**inches=inches1+inches2;**

**Distance obj= new Distance(feet,inches);**

**obj.Calculate(obj);**

**}**

**}**



**4. Write a program to implement inheritance Concept in Java**

import java.util.Scanner;

class A

{

void methA()

{

System.out.println(" I belong to class A");

}

}

class B extends A

{

void methB()

{

methA();

System.out.println(" I belong to class B");

}

}

class C extends B

{

void methC()

{

methB();

System.out.println(" I belong to class C ");

}

}

class D extends A

{

void methD()

{

methA();

System.out.println(" I belong to class D");

}

}

public class Inherit

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

System.out.println("1.Single Inheritance");

System.out.println("2.Multilevel Inheritance");

System.out.println("3.Hierarical Inheritance");

System.out.println("Enter user choice:");

int ch=sc.nextInt();

switch(ch)

{

case 1:B obj= new C();

obj.methB();

break;

case 2:C obj1= new C();

obj1.methC();

break;

case 3:D obj2= new D();

obj2.methD();

break;

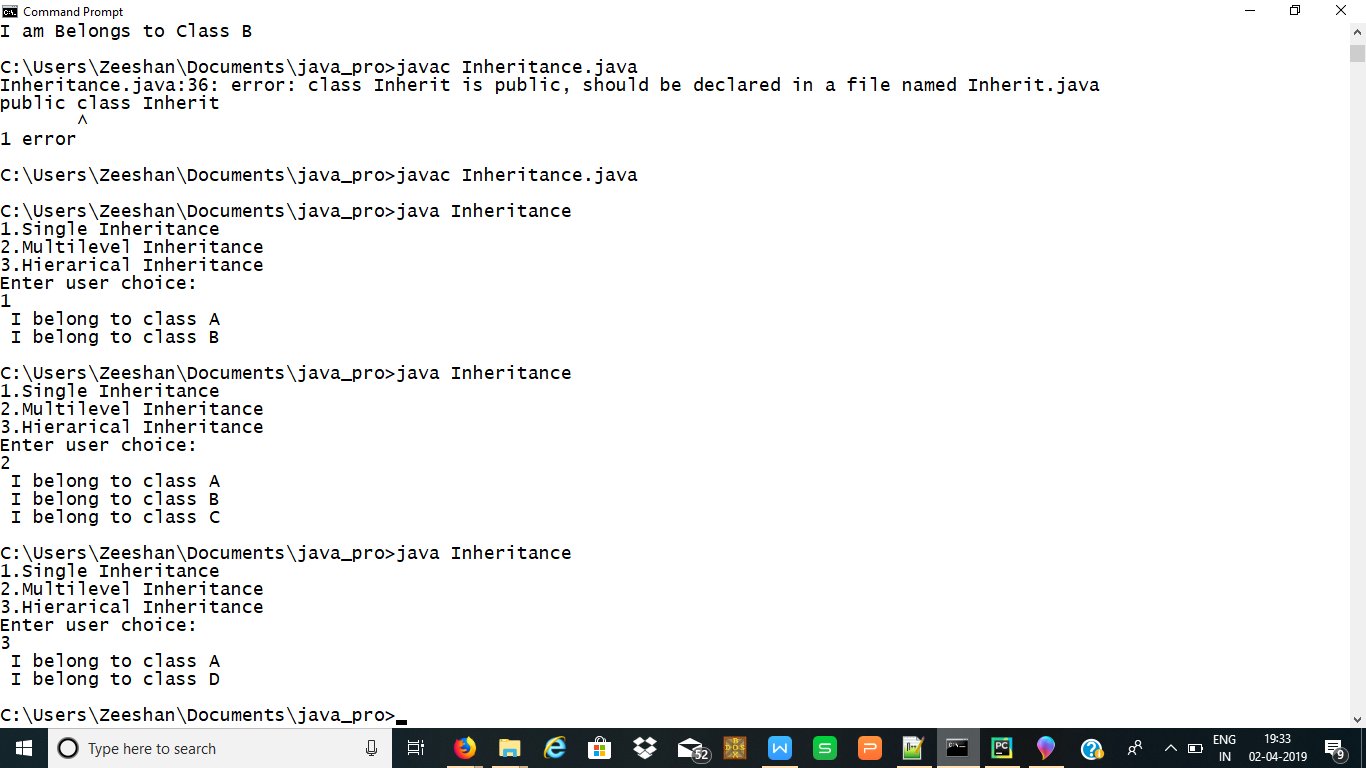
default:System.out.println("invalid");

break;

}

}

}



**5. Write a program to explain the concept of runtime polymorphism in java.**

import java.util.\*;

class Bank

{

void Transfer()

{

}

}

public class Customer extends Bank

{ //long[] longArray = new long[]{ 12345678910 };

int count;

static int balance=3000;

public static void main(String[]args)

{

Customer obj = new Customer();

obj.Transfer();

}

void Transfer()

{

System.out.println("Select the payment option");

System.out.println("1.RTGS\n2.NTF\n3.ABC\n");

Scanner in = new Scanner(System.in);

int a = in.nextInt();

switch(a)

{

case 1 :

case 2 :

case 3 : System.out.println("Enter the account number");

// Scanner in=new Scanner(System.in);

long n =in.nextLong();

while (n != 0)

{

n = n / 10;

++count;

}

System.out.println("Valid Account Number....Please enter the

further details");

if((count<10)||(count>17))

{

System.out.println("Invalid Account number");

break;

}

System.out.println("Enter the IFSC code:");

int b=in.nextInt();

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

int amt = in.nextInt();

if(amt>balance)

{

System.out.println("Insufficient Balance");

break;

}

else

balance=balance-amt;

System.out.println("Amount Transfered Succesfully");

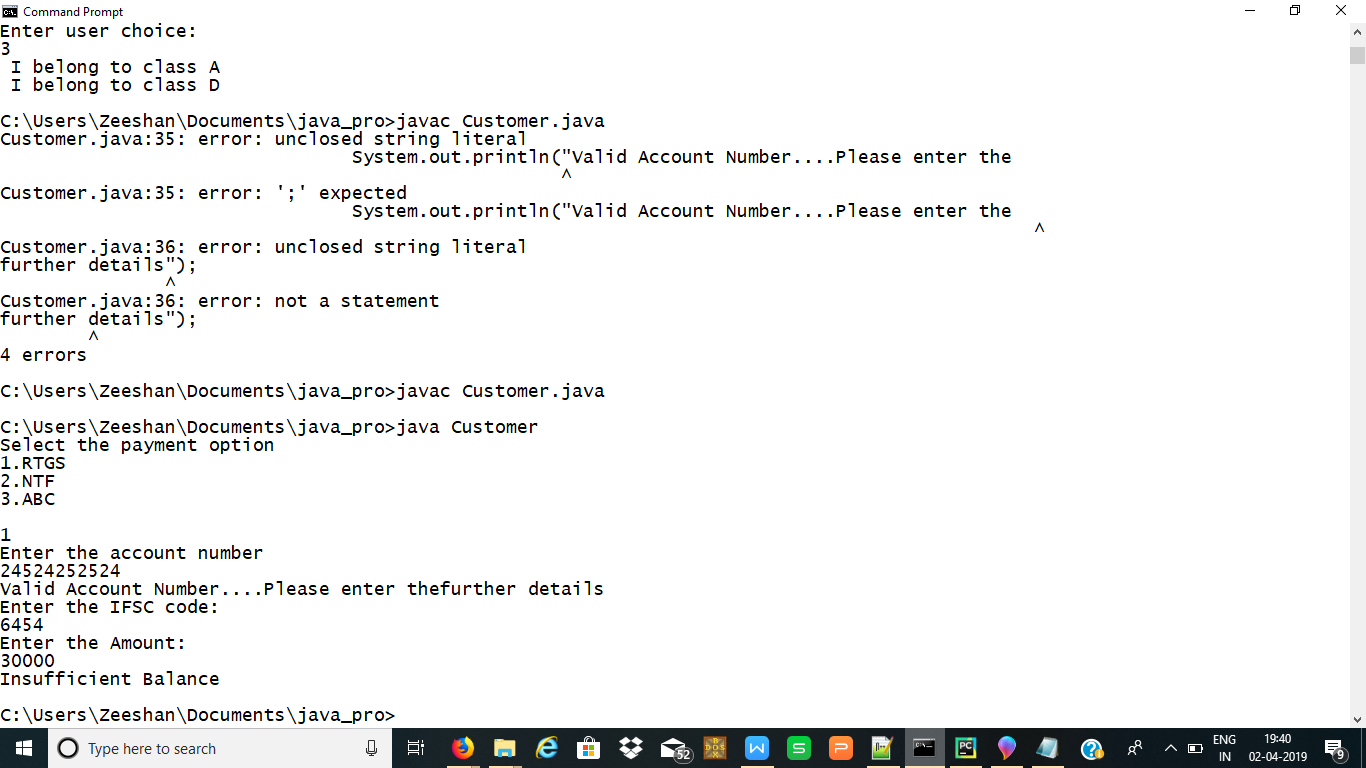
System.out.println("Your current balance is:"+balance);

break;

}

}

}



**6. Write a program to implement producer consumer problem using thread concept.**

import java.util.\*;

public class ProducerConsumerTest

{

public static void main(String[]args)

{

CubbyHole c = new CubbyHole();

Producer p1 = new Producer(c,1);

Consumer c1 = new Consumer(c,1);

p1.start();

c1.start();

}

}

class CubbyHole

{

private int contents;

private boolean available = false;

public synchronized int get()

{

while(available == false)

{

try

{

wait();

}

catch(InterruptedException e)

{

}

}

available = false;

notifyAll();

return contents;

}

public synchronized void put(int value)

{

while(available == true)

{

try

{

wait();

}

catch(InterruptedException e)

{

}

}

contents = value;

available = true;

notifyAll();

}

}

class Consumer extends Thread

{

private CubbyHole cubbyhole;

private int number;

public Consumer(CubbyHole c, int number)

{

cubbyhole = c;

this.number = number;

}

public void run()

{

int value = 0;

for(int i=0;i<10;i++)

{

value = cubbyhole.get();

System.out.println("Consumer#"+this.number + "get:" + value);

}

}

}

class Producer extends Thread

{

private CubbyHole cubbyhole;

private int number;

public Producer(CubbyHole c, int number)

{

cubbyhole = c;

this.number = number;

}

public void run()

{

for(int i=0;i<10;i++)

{

cubbyhole.put(i);

System.out.println("Producer#"+this.number + "put:" + i);

try

{

sleep((int)(Math.random()\*100));

}catch(InterruptedException e)

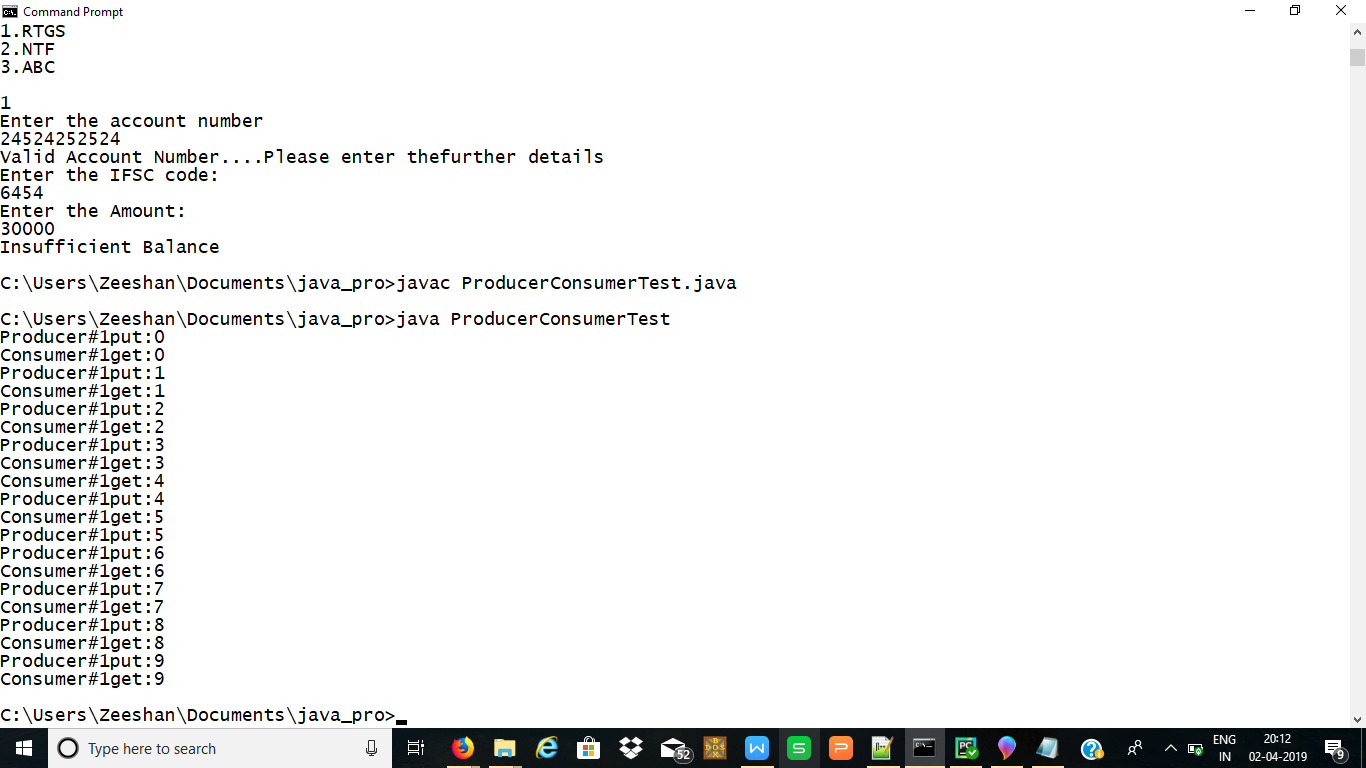
{

}

}

}

}



**7.(a) Write a program to create object for Stack and use all methods.**

import java.util.Scanner;

import java.util.Stack;

class Stack\_Ex4

{

Scanner scan;

Stack<String> stack;

int n;

void push()

{

scan = new Scanner(System.in);

stack = new Stack<String>();

System.out.println("Stack Push, Pop and Peek");

System.out.println("Enter the number of elements to be inserted into the stack :");

n = scan.nextInt();

System.out.println("Enter the data - PUSH:");

for(int i=0; i<n; i++)

{

stack.push(scan.next());

}

}

void peek()

{

System.out.println("Top Value of the Stack - PEEK:");

if(stack.isEmpty())

System.out.println("The Stack is Empty..");

else

System.out.println(stack.peek());

}

void pop()

{

System.out.println("The Stack - POP");

while(!stack.empty())

{

System.out.println(stack.pop());

}

peek();

}

}

public class MainClass

{

public static void main(String args[])

{

Stack\_Ex4 obj = new Stack\_Ex4();

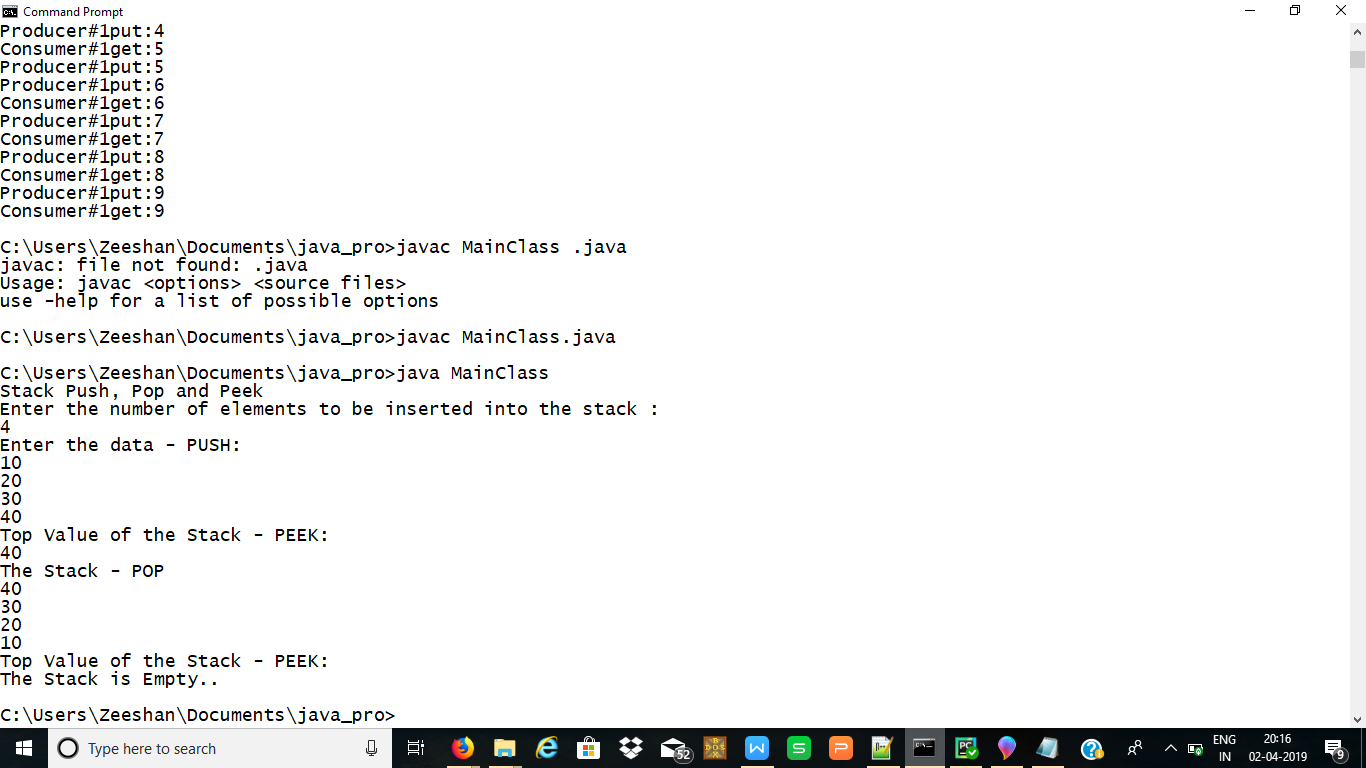
obj.push();

obj.peek();

obj.pop();

}

}



**7 . (b) Write a program to create object for TreeSet and use all methods.**

import java.util.TreeSet;

public class Treesetdemo

{

public static void main(String args[])

{

TreeSet<String> PlayerSet = new TreeSet<String>();

PlayerSet.add("Sachin");

PlayerSet.add("Virat");

PlayerSet.add("Dhoni");

PlayerSet.add("Umesh");

PlayerSet.add("Mahi");

PlayerSet.add("Zahir");

PlayerSet.add("Bhajji");

PlayerSet.add("Rahul");

PlayerSet.add("Pathan");

PlayerSet.add("Dhoni");

System.out.println("original set:" + PlayerSet);

System.out.println("First Name:" + PlayerSet.first());

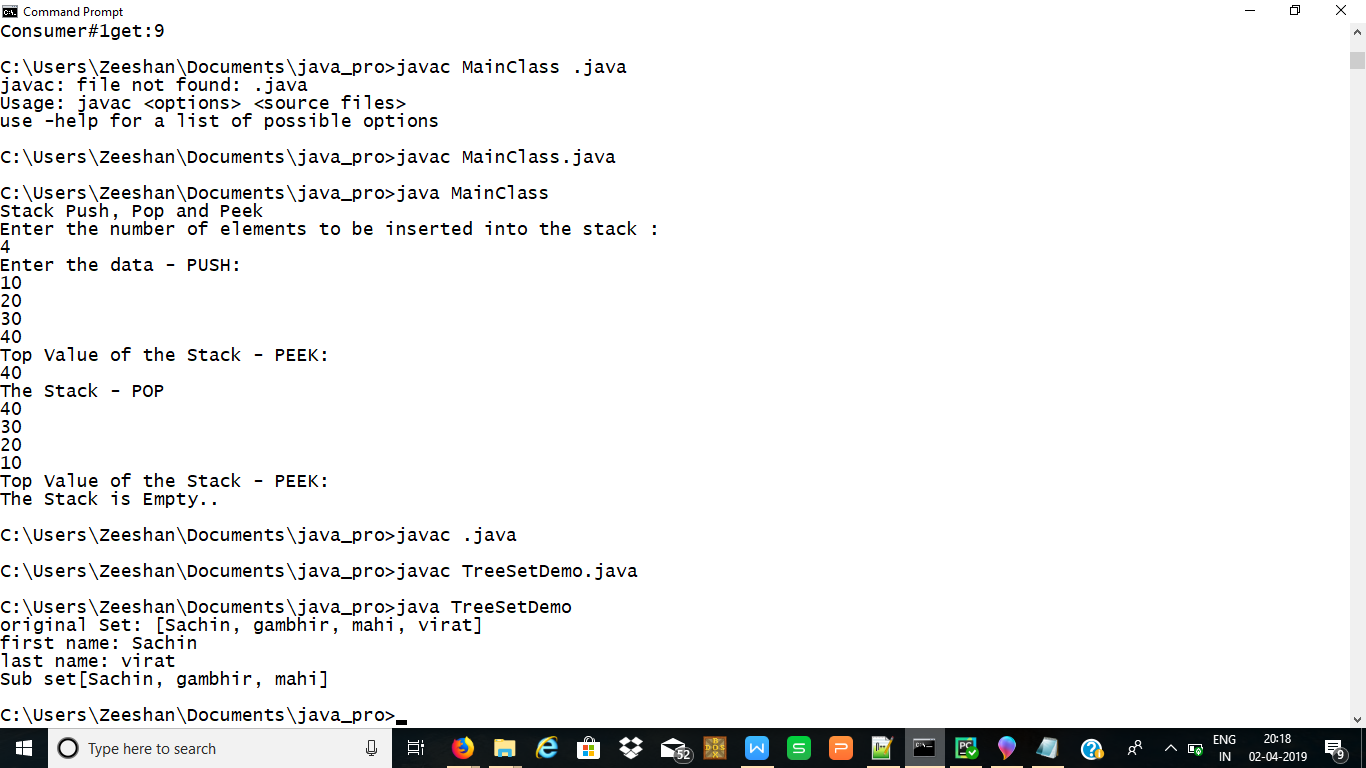
System.out.println("Last Name:" + PlayerSet.last());

TreeSet<String> newPlaySet = (TreeSet<String>) PlayerSet.subSet("Mahi", "Umesh");

System.out.println(" Sub set:"+ newPlaySet);

}

}



**8. Write a program to implement Exception handling in Java**

import java.io.\*;

public class CheckingAccount

{

private double balance;

private int number;

public CheckingAccount(int number)

{

this.number = number;

}

public void deposit(double amount)

{

balance += amount;

}

public void withdraw(double amount) throws InsufficientFundsException

{

if(amount <= balance)

{

balance -= amount;

}

else {

double needs = amount - balance;

throw new InsufficientFundsException(needs);

}

}

public double getBalance()

{

return balance;

}

public int getNumber()

{

return number;

}

}

import java.io.\*;

public class InsufficientFundsException extends Exception

{

private double amount;

public InsufficientFundsException(double amount)

{

this.amount = amount;

}

public double getAmount()

{

return amount;

}

}

public class BankDemo

{

public static void main(String [] args)

{

CheckingAccount c = new CheckingAccount(101);

System.out.println("Depositing $500...");

c.deposit(500.00);

try

{

System.out.println("\nWithdrawing $100...");

c.withdraw(100.00);

System.out.println("\nWithdrawing $600...");

c.withdraw(600.00);

}

catch (InsufficientFundsException e)

{

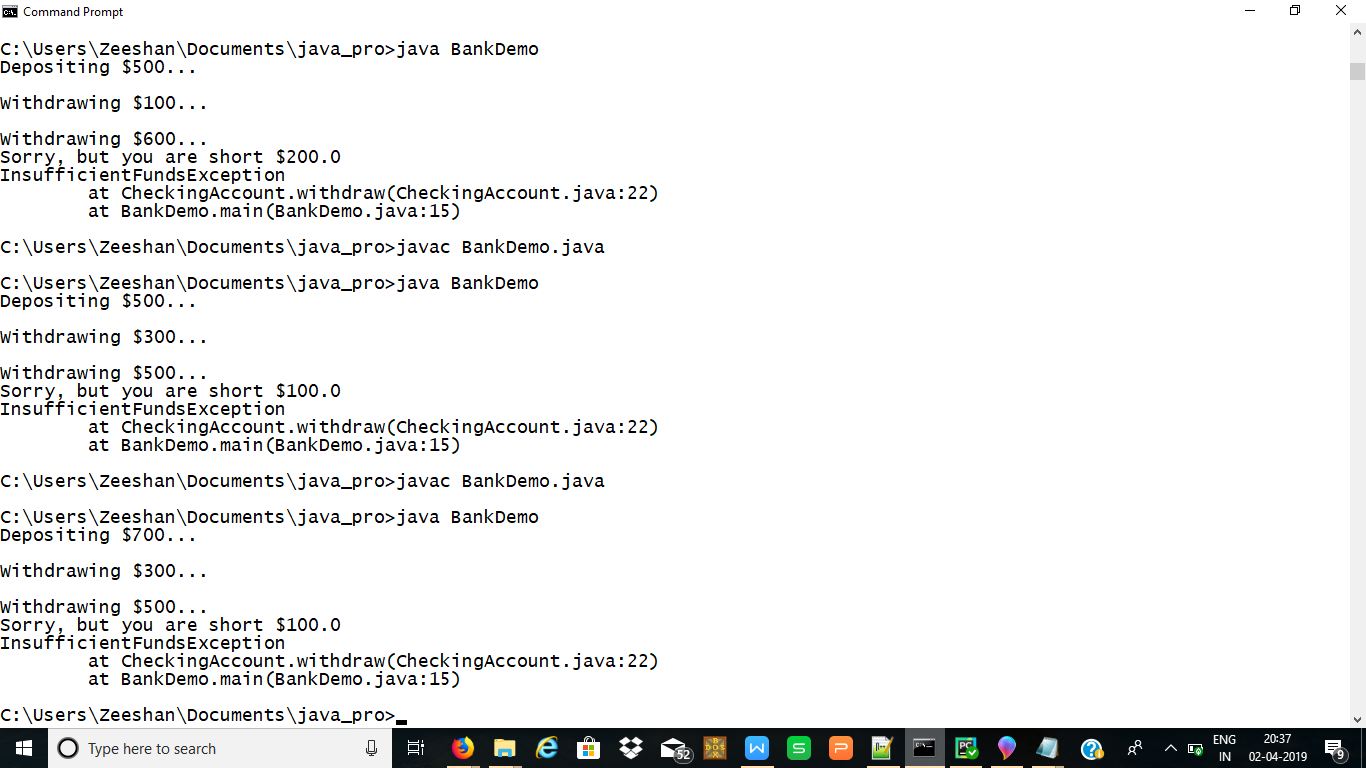
System.out.println("Sorry, but you are short $" + e.getAmount());

e.printStackTrace();

}

}

}



**9. Write a program to implement the Concept of Interface in Java**

import java.util.\*;

interface Arithmetic

{

public void add(int a,int b);

public void sub(int a,int b);

public void mul(int a,int b);

public void div(int a,int b);

public void mod(int a,int b);

};

public class InterfaceDemo implements Arithmetic

{

int res;

public void add(int a,int b)

{

res=a+b;

System.out.println("The sum of two values is: " + res);

}

public void sub(int a,int b)

{

res=a-b;

System.out.println("The diference of two values is: " + res);

}

public void mul(int a,int b)

{

res=a\*b;

System.out.println("The product of two values is: " + res);

}

public void div(int a,int b)

{

res=a/b;

System.out.println("The result is: " + res);

}

public void mod(int a,int b)

{

res=a%b;

System.out.println("The modulus of two values is: " + res);

}

public static void main(String args[])

{

InterfaceDemo op=new InterfaceDemo();

op.add(10,20);

op.sub(10,20);

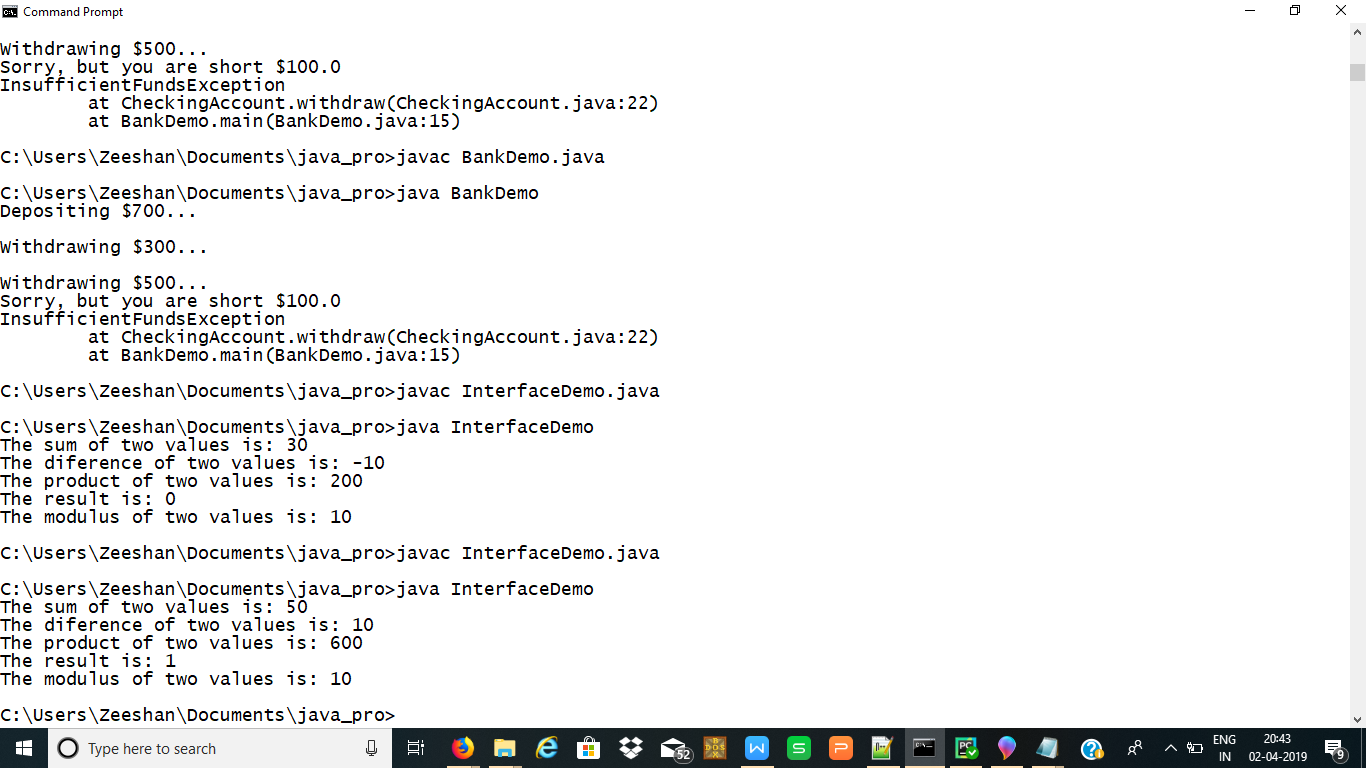
op.mul(10,20);

op.div(10,20);

op.mod(10,20);

}

}



.**10. Write a program to get file name at runtime and display number of lines and words in that file.**

import java.util.\*;

import java.io.\*;

class Filecontents

{

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

{

int l=0,w=0;

int c;

String x = args[0];

File f=new File(x);

InputStreamReader in = new InputStreamReader(new FileInputStream(f));

while((c=in.read())!=-1)

{

if((c==' ') || (c=='\n'))

w++;

if(c=='\n')

l++;

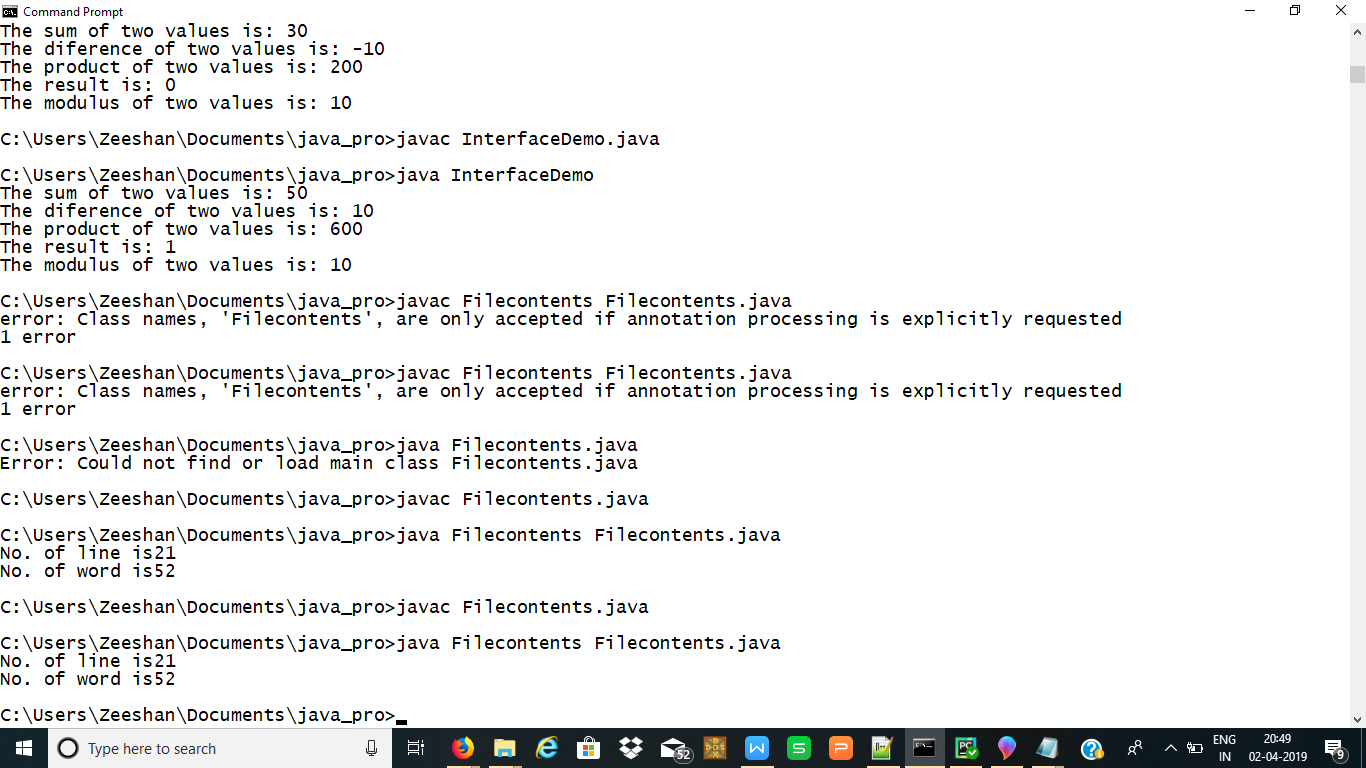
}

System.out.println("No. of line is"+l);

System.out.println("No. of word is"+w);

}

}



**11. Write a program to list files in the current working directory depending upon a given pattern**.

import java.io.\*;

import java.util.Scanner;

class Directory

{

public static void main(String args[])

{

System.out.println("Enter the directory name");

Scanner s= new Scanner(System.in);

String dir=s.nextLine();

File f=new File(dir);

if(f.exists())

{

if(f.isDirectory())

{

String f1[]=f.list();

System.out.println(f.list());

for(int i=0;i<f1.length;i++)

{

String tp=f1[i];

if(tp.endsWith("java"))

System.out.println("\t\t"+tp);

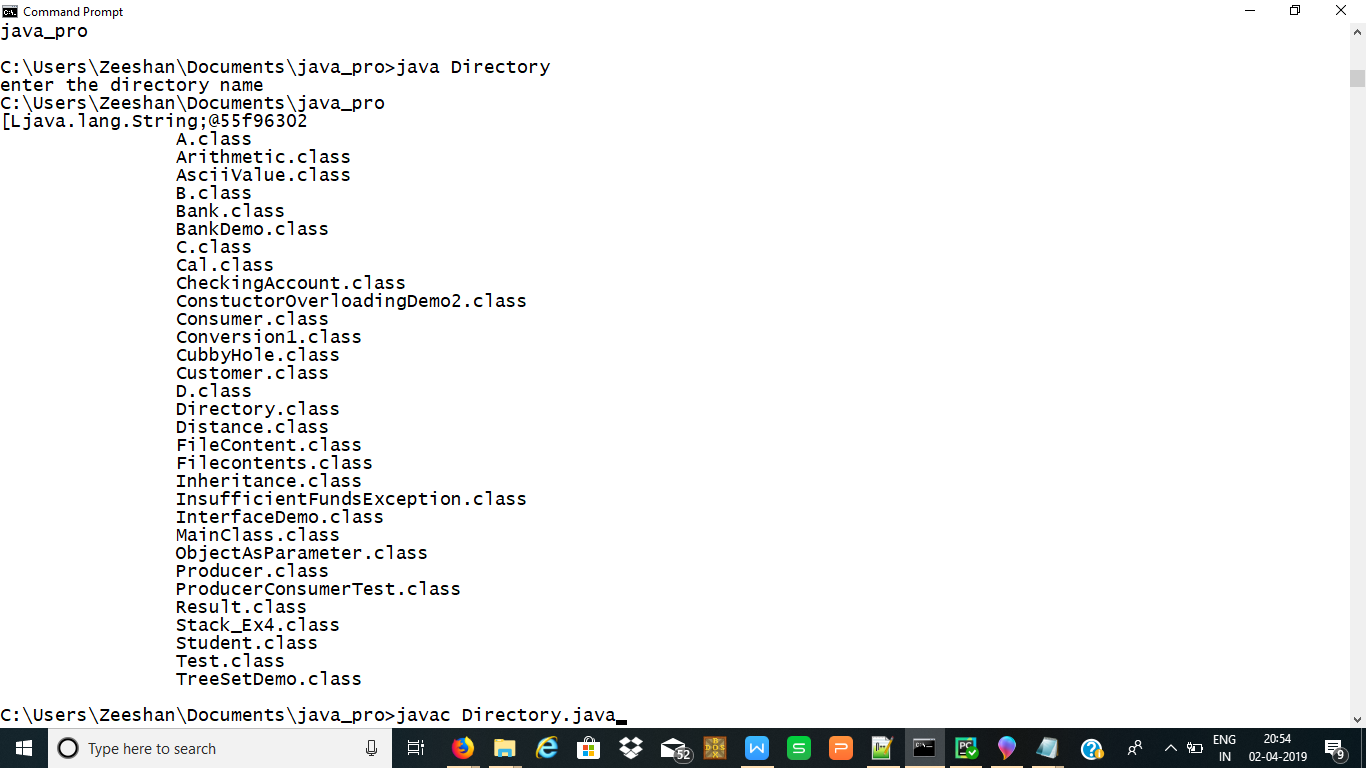
}

}

}

}

}



**12. Create a Frame for Student Registration containing all the fields Name, Age, Contact, Father’s Name, Annual Income and a submit button. Perform field validations.**

**13. Write a applet program to create a calculator**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

/\*

<applet code="Cal" width=300 height=300>

</applet>

\*/

public class Cal extends Applet

implements ActionListener

{

String msg=" ";

int v1,v2,result;

TextField t1;

Button b[]=new Button[10];

Button add,sub,mul,div,clear,mod,EQ;

char OP;

public void init()

{

Color k=new Color(120,89,90);

setBackground(k);

t1=new TextField(10);

GridLayout gl=new GridLayout(4,5);

setLayout(gl);

for(int i=0;i<10;i++)

{

b[i]=new Button(""+i);

}

add=new Button("add");

sub=new Button("sub");

mul=new Button("mul");

div=new Button("div");

mod=new Button("mod");

clear=new Button("clear");

EQ=new Button("EQ");

t1.addActionListener(this);

add(t1);

for(int i=0;i<10;i++)

{

add(b[i]);

}

add(add);

add(sub);

add(mul);

add(div);

add(mod);

add(clear);

add(EQ);

for(int i=0;i<10;i++)

{

b[i].addActionListener(this);

}

add.addActionListener(this);

sub.addActionListener(this);

mul.addActionListener(this);

div.addActionListener(this);

mod.addActionListener(this);

clear.addActionListener(this);

EQ.addActionListener(this);

}

public void actionPerformed(ActionEvent ae)

{

String str=ae.getActionCommand();

char ch=str.charAt(0);

if ( Character.isDigit(ch))

t1.setText(t1.getText()+str);

else

if(str.equals("add"))

{

v1=Integer.parseInt(t1.getText());

OP='+';

t1.setText("");

}

else if(str.equals("sub"))

{

v1=Integer.parseInt(t1.getText());

OP='-';

t1.setText("");

}

else if(str.equals("mul"))

{

v1=Integer.parseInt(t1.getText());

OP='\*';

t1.setText("");

}

else if(str.equals("div"))

{

v1=Integer.parseInt(t1.getText());

OP='/';

t1.setText("");

}

else if(str.equals("mod"))

{

v1=Integer.parseInt(t1.getText());

OP='%';

t1.setText("");

}

if(str.equals("EQ"))

{

v2=Integer.parseInt(t1.getText());

if(OP=='+')

result=v1+v2;

else if(OP=='-')

result=v1-v2;

else if(OP=='\*')

result=v1\*v2;

else if(OP=='/')

result=v1/v2;

else if(OP=='%')

result=v1%v2;

t1.setText(""+result);

}

if(str.equals("clear"))

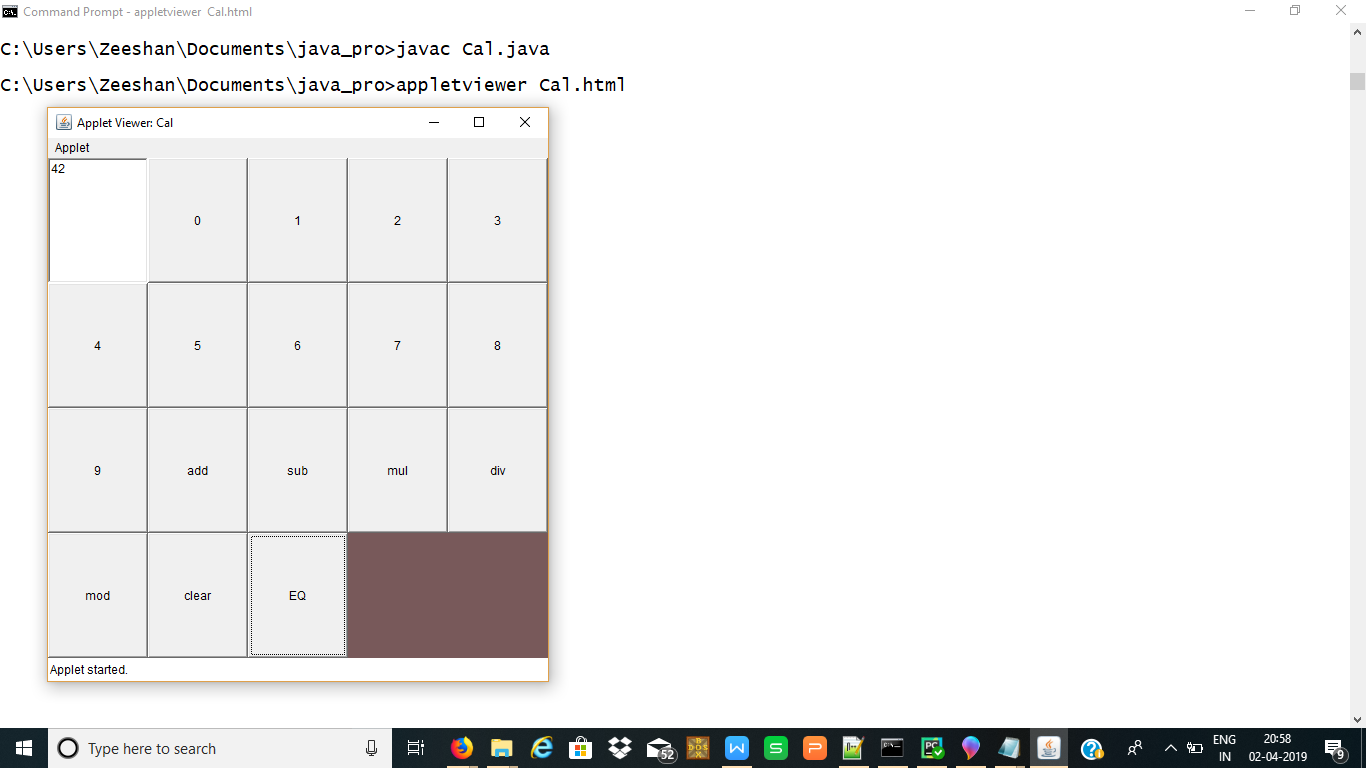
{

t1.setText("");

}

}

}



**14. Write a program to perform the following operation using JDBC: Insert, Update, Delete and Select Data.**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

class ODBCTestNew

{

void createTable(Statement st)

{

try

{

st.executeUpdate("CREATE TABLE EMP12( EMP\_ID int,EMP\_NAME VARCHAR(10),EMP\_SAL int);");

System.out.println("TABLE CREATED-----");

} catch(Exception e)

{

System.out.println(e.getMessage());

}

}

void insertTable(Statement st)

{

try

{

st.executeUpdate("INSERT INTO EMP12 VALUES (123,'RAJU',44567)");

st.executeUpdate("INSERT INTO EMP12 VALUES (124,'JESWIN',54569)");

st.executeUpdate("INSERT INTO EMP12 VALUES (125,'NATHAN',34567)");

st.executeUpdate("INSERT INTO EMP12 VALUES (126,'REUBEN',35567)");

System.out.println("4 ROWS CREATED");

} catch(Exception e)

{

System.out.println(e.getMessage());

}

}

void displayTable()

{

try

{

Connection con = DriverManager.getConnection("jdbc:odbc:Test4", " "," ");

Statement st = con.createStatement();

ResultSet rs = st.executeQuery("SELECT \* FROM EMP12");

while(rs.next())

{

System.out.println("The Employee ID is: " +rs.getInt(1));

System.out.println("The Employee Name is: " +rs.getString(2));

System.out.println("The Employee Salary is: " +rs.getInt(3));

}

st.close();

}catch(Exception e)

{

System.out.println(e.getMessage());

}

}

void updateTable(Statement st)

{

try

{

st.executeUpdate("UPDATE EMP12 SET EMP\_NAME='ATHUL', EMP\_SAL=10000 WHERE EMP\_ID=123");

st.executeUpdate("UPDATE EMP12 SET EMP\_NAME='SUNIL', EMP\_SAL=20000 WHERE EMP\_ID=124");

displayTable();

System.out.println("THE EMPLOYEE 123 AND 124 ARE UPDATED");

} catch(Exception e)

{

System.out.println(e.getMessage());

}

}

void deleteRecord(Statement st)

{

try

{

st.executeUpdate("DELETE FROM EMP12 WHERE EMP\_ID=123 ");

st.executeUpdate("DELETE FROM EMP12 WHERE EMP\_ID=124 ");

System.out.println("The EMPLOYEE ID 123 AND 124 IS DELETED-");

displayTable();

} catch(Exception e)

{

System.out.println(e.getMessage());

}

}

void dropTable(Statement st)

{

try

{

st.executeUpdate("DROP TABLE EMP12");

System.out.println("EMPLOYEE TABLE IS DELETED----");

} catch(Exception e)

{

System.out.println(e.getMessage());

}

}

}

public class Output

{

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

{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con = DriverManager.getConnection("jdbc:odbc:Test4", " "," ");

Statement st = con.createStatement();

ODBCTestNew oo=new ODBCTestNew();

oo.createTable(st);

oo.insertTable(st);

oo.displayTable();

oo.updateTable(st);

oo.deleteRecord(st);

oo.dropTable(st);

}

}