**Lesson-1**

**1.Write a Java program, which will take your name as command line argument and will print**

**Hello <your name>**

class Tesla

{

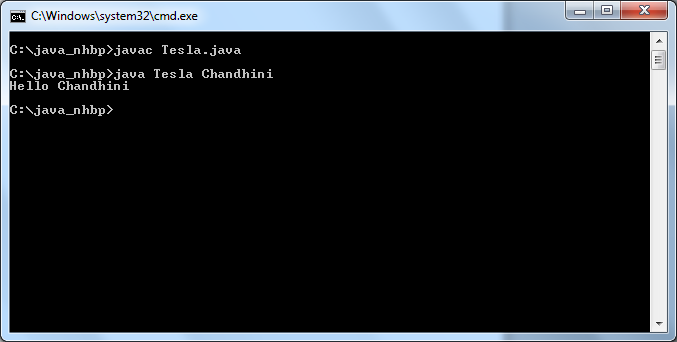
public static void main(String arg[])

{

System.out.println("Hello "+arg[0]);

}

}



**Lesson-2**

**1.Write a Java program to declare two integer variables, one float variable, and one string variable and assign 10, 12.5,and "Java programming" to them respectively. Then display their values on the screen.**

import java.io.\*;

class Print

{

public static void main(String arg[])

{

int a=10,b=12;

float f=12.5f;

String s="Java Programming";

System.out.println(a);

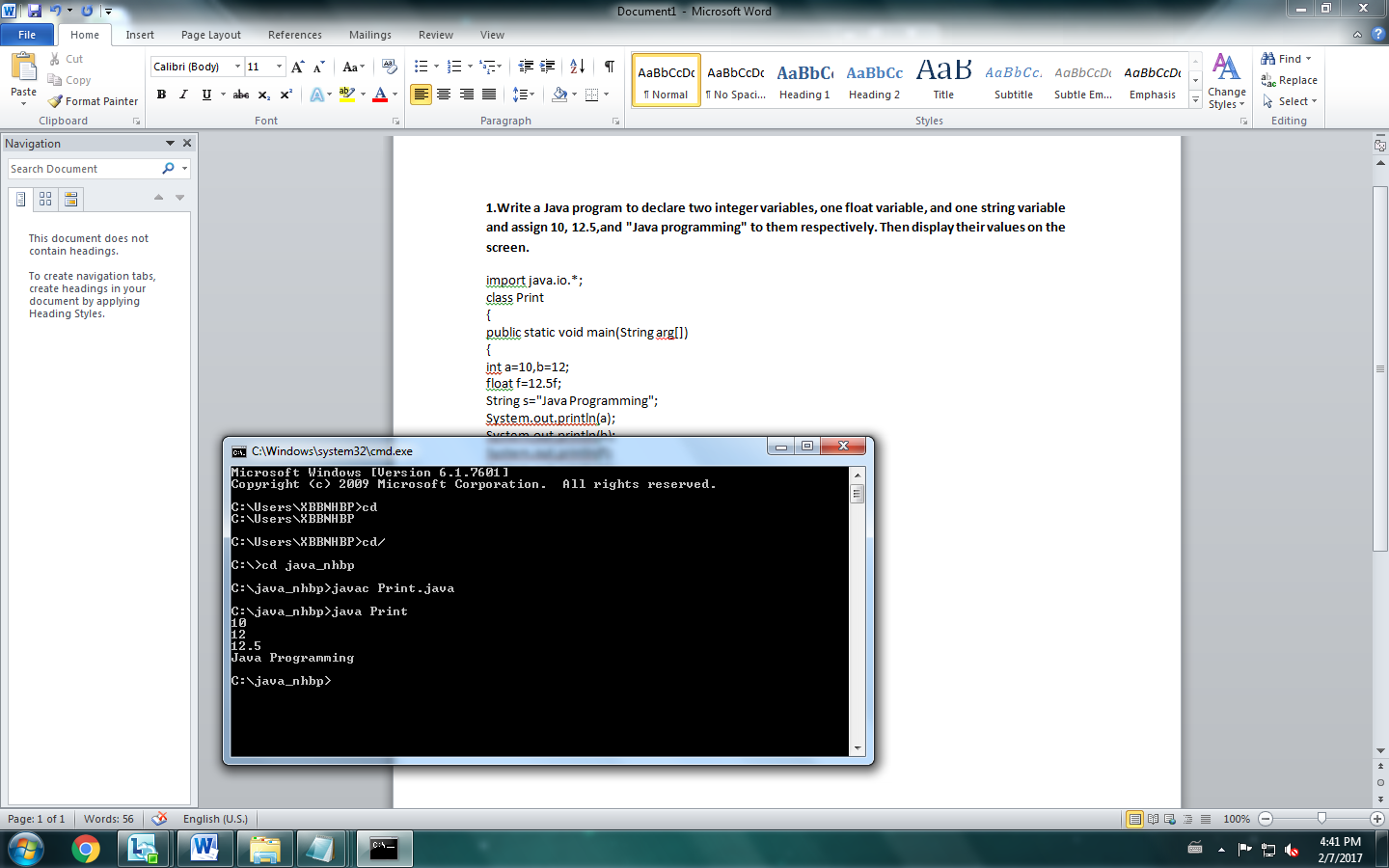
System.out.println(b);

System.out.println(f);

System.out.println(s);

}

}



**2.Write Java program to allow the user to input two integer values and then the program prints the results of adding,subtracting, multiplying, and dividing among the two values.**

import java.io.\*;

import java.util.\*;

class Opr

{

public static void main(String s[])

{

int a,b,c,d,e;

float f;

Scanner sc=new Scanner(System.in);

System.out.print("Enter numbers");

a=sc.nextInt();

b=sc.nextInt();

c=a+b;

e=a\*b;

if(a>b)

{

d=a-b;

}

else

{

d=b-a;

}

if(a>b)

{

f=a/b;

}

else

{

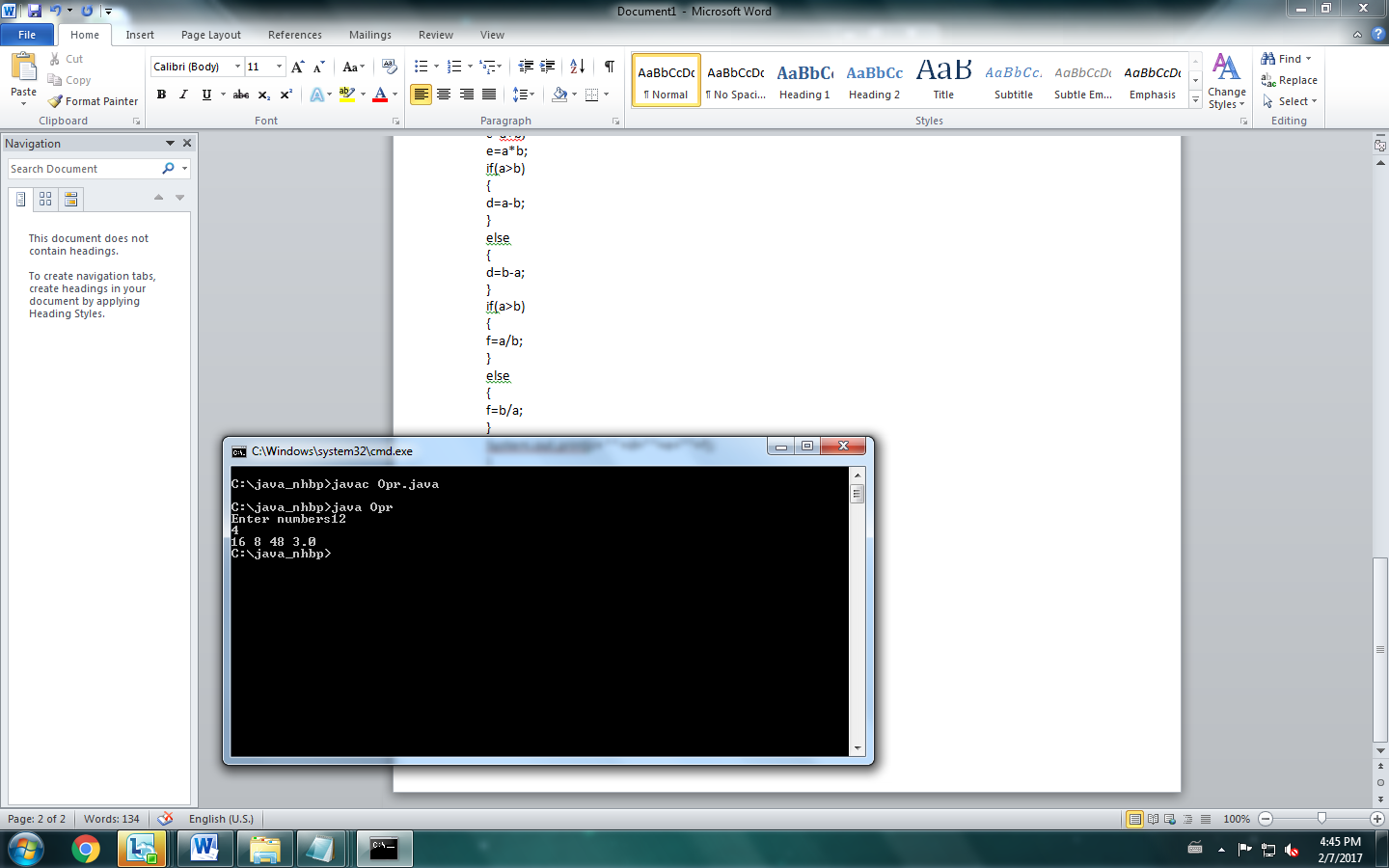
f=b/a;

}

System.out.print(c+ " " +d+" "+e+" "+f);

}

}



**3.Write Java program to allow the user to input two float values and then the program adds the two values together.The result will be assigned to the first variable.**

import java.io.\*;

class Flt

{

public static void main(String s[])

{

float a,b;

a=Float.parseFloat(s[0]);

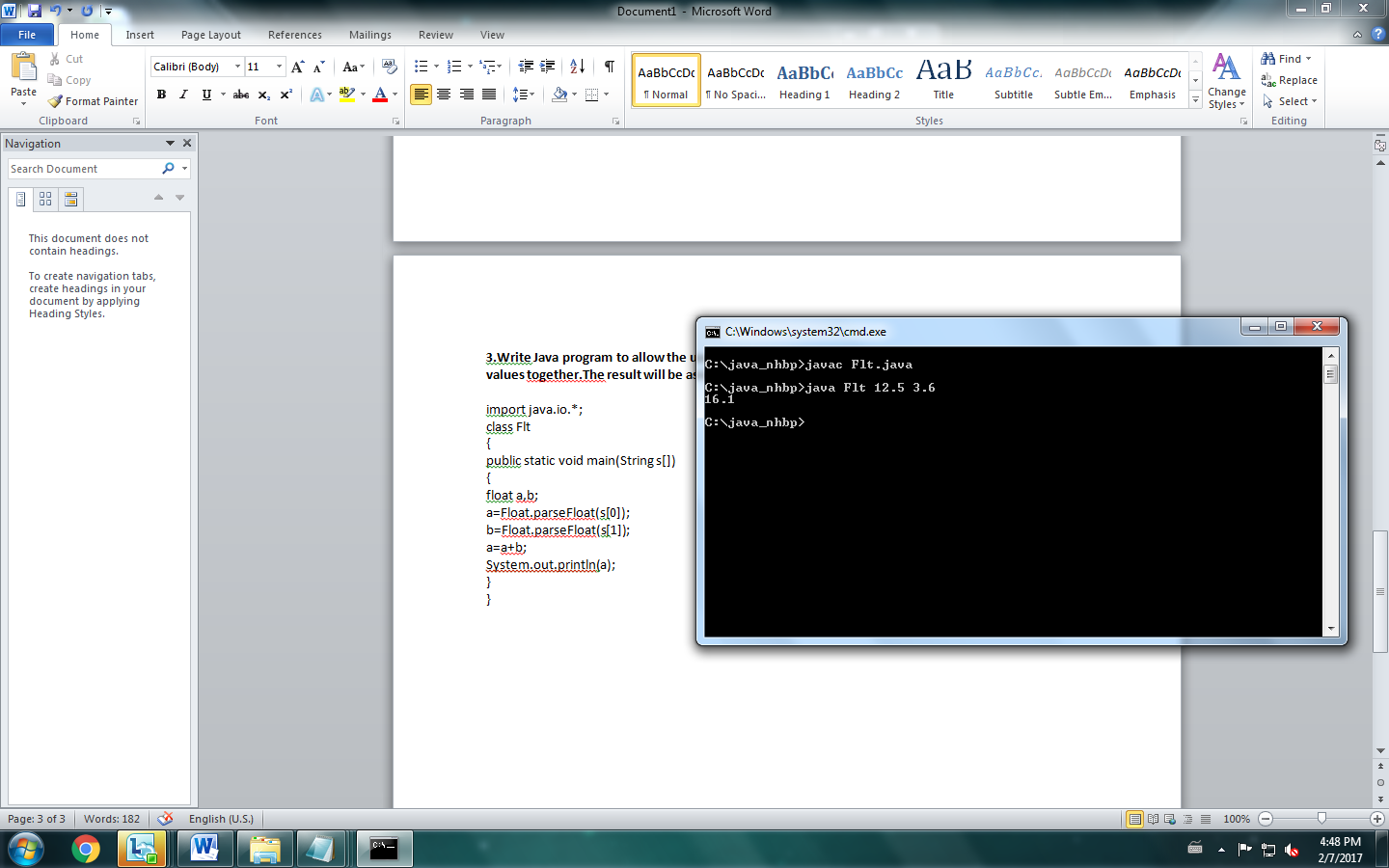
b=Float.parseFloat(s[1]);

a=a+b;

System.out.println(a);

}

}



**4.Write Java program to allow the user to input the amount of deposit, yearly interest rate (percentage), and income tax(percentage).Then the program will calculate the amount of interest that the person earns in the year.**

import java.io.\*;

class Prin

{

public static void main(String s[])

{

int a;

float b,c,si,amt,intr;

a=Integer.parseInt(s[0]);

b=Float.parseFloat(s[1]);

c=Float.parseFloat(s[2]);

si=((a\*b)/100);

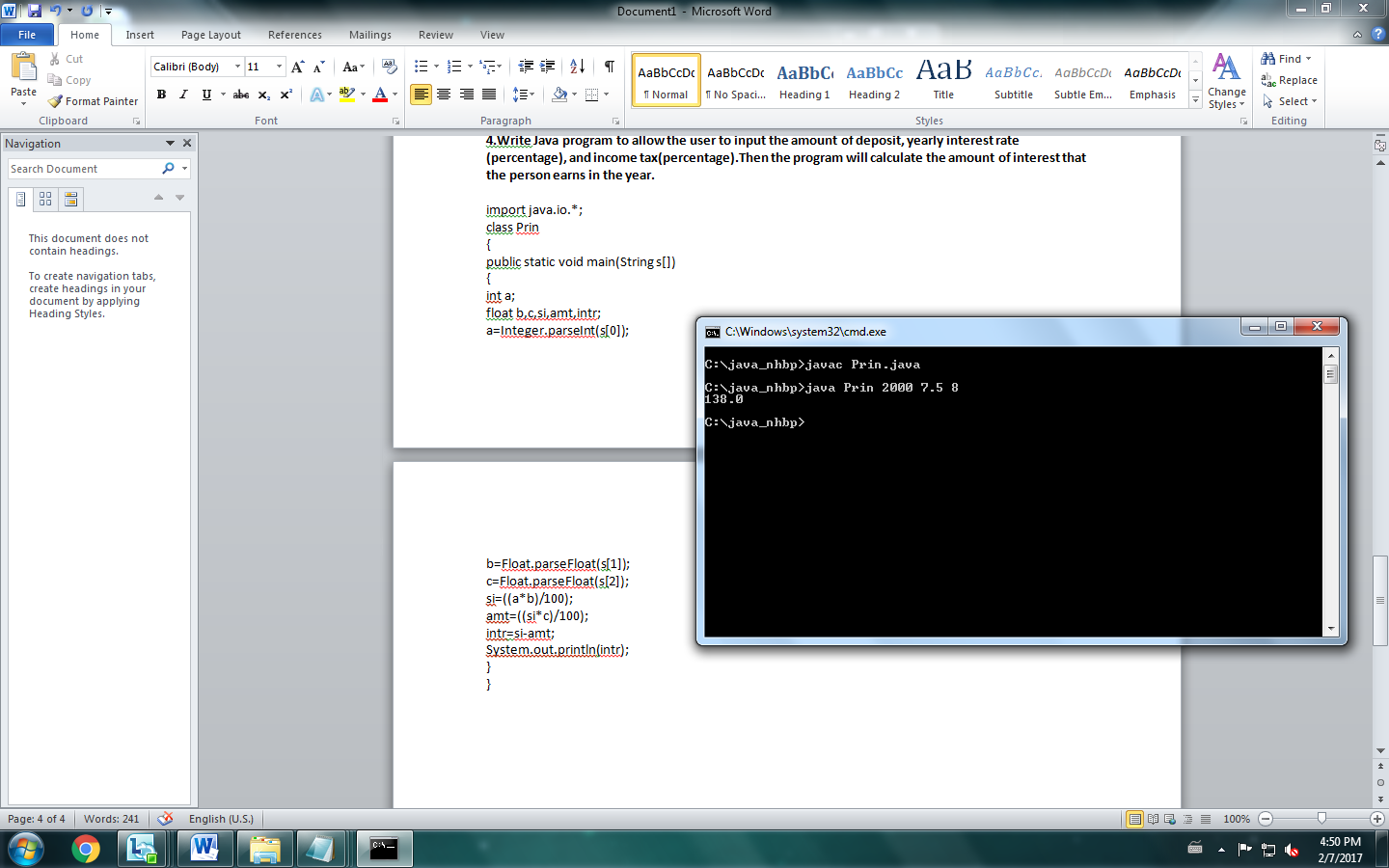
amt=((si\*c)/100);

intr=si-amt;

System.out.println(intr);

}

}



**5.Write Java program to allow the user to input his/her age. Then the program will show if the person is eligible to vote.A person who is eligible to vote must be older than or equal to 18 years old.**

import java.io.\*;

class Age

{

public static void main(String s[])

{

int a;

a=Integer.parseInt(s[0]);

if(a>=18)

{

System.out.println("Eligible to vote");

}else

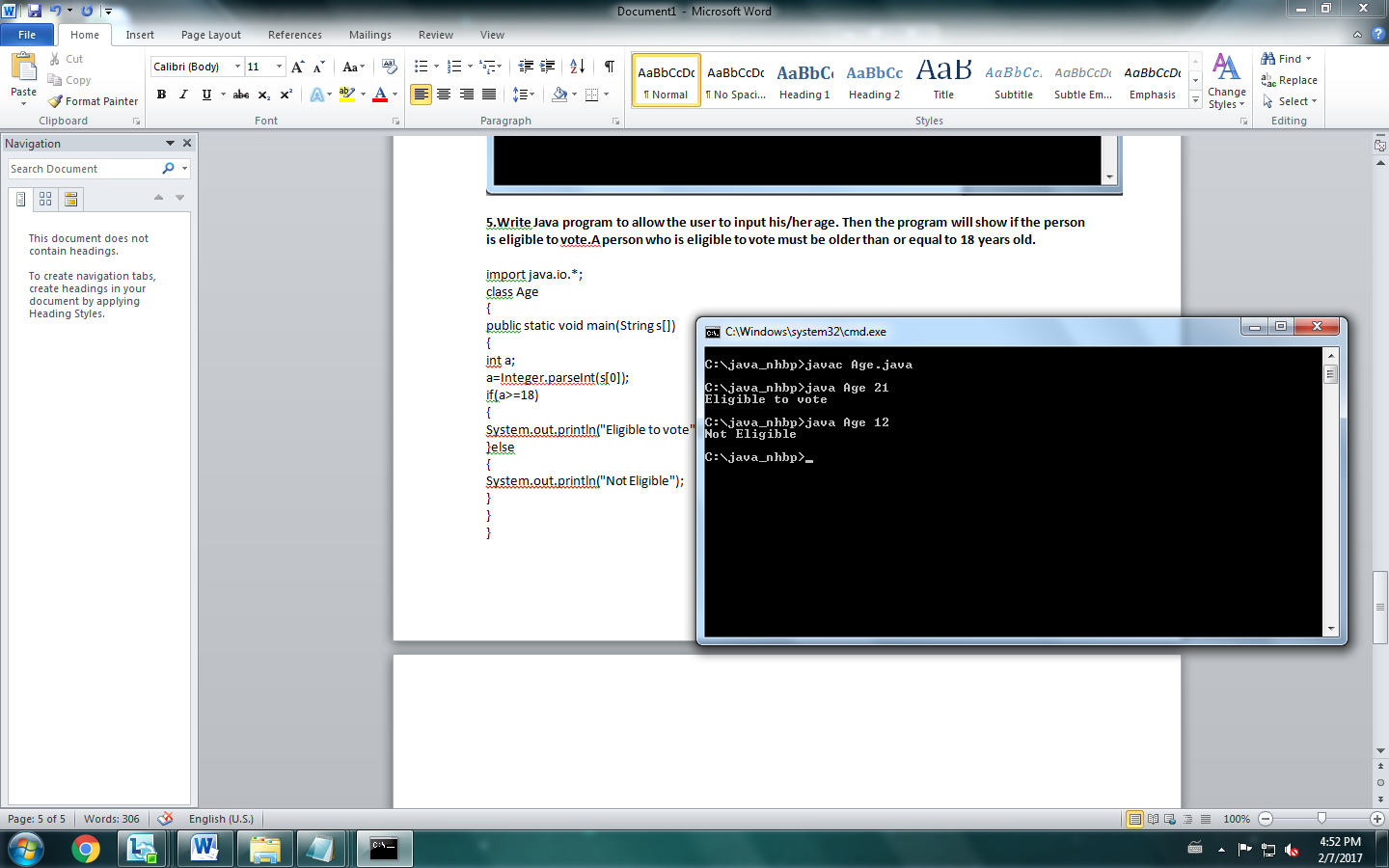
{

System.out.println("Not Eligible");

}

}

}



**6.Write a Java program to determine whether an input number is an even number also**

**9.Write a program called CheckOddEven which prints "Odd Number" if the int variable “number” is odd, or “Even Number” otherwise**

import java.io.\*;

class Even

{

public static void main(String s[])

{

int a;

a=Integer.parseInt(s[0]);

if(a%2==0)

{

System.out.println("Even");

}

else

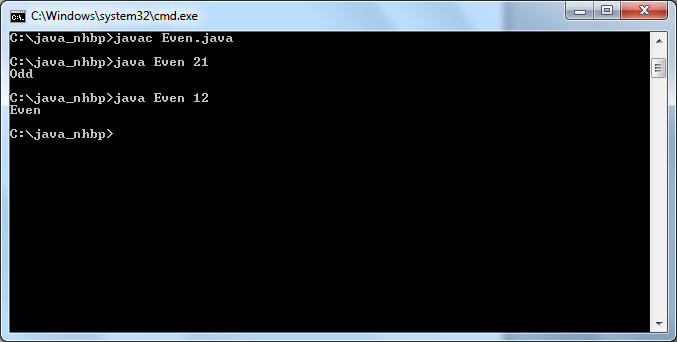
{

System.out.println("Odd");

}

}

}



**8.Write a program called CheckPassFail which prints "PASS" if the int variable "mark" is more than or equal to 50; or prints "FAIL" otherwise.**

import java.io.\*;

class Pass

{

public static void main(String s[])

{

int a;

a=Integer.parseInt(s[0]);

if(a>=50)

{

System.out.println("PASS");

}

else

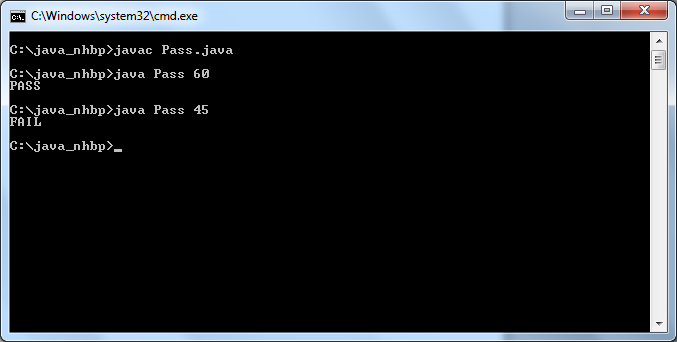
{

System.out.println("FAIL");

}

}

}



**10.Write a program called SumAndAverage to produce the sum of 1, 2, 3, ..., to an upperbound (e.g., 100).Also compute and display the average.Use for loop**

import java.io.\*;

class Sum

{

public static void main(String s[])

{

float avg;

int n=Integer.parseInt(s[0]);

int sum=0;

for(int i=1;i<=n;i++)

{

sum=sum+i;

}

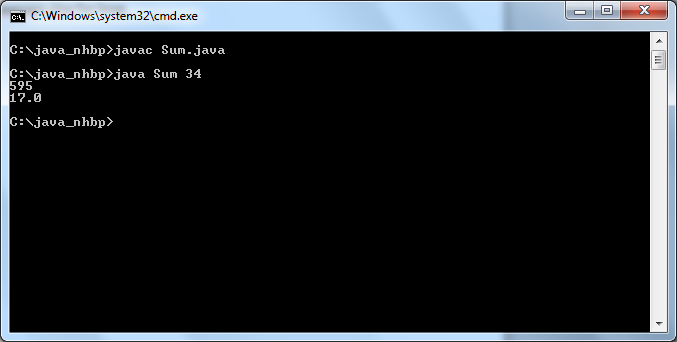
System.out.println(sum);

avg=sum/n;

System.out.println(avg);

}

}



**Lesson-3**

**1.Write a program to declare a Box class which has the variables width, height and depth**

import java.io.\*;

class Box

{

int width=10,height=20,depth=15;

}

class Imp

{

public static void main(String arg[])

{

Box b=new Box();

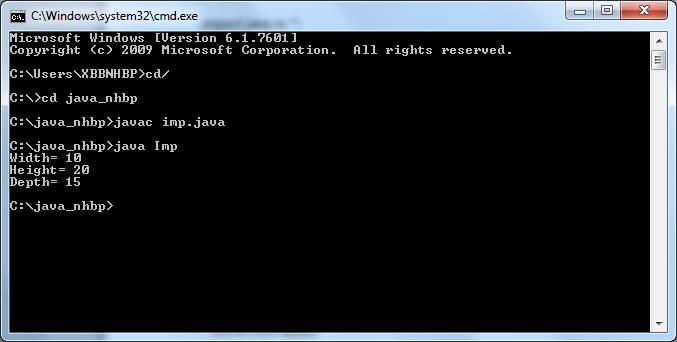
System.out.println("Width= " +b.width);

System.out.println("Height= " +b.height);

System.out.println("Depth= " +b.depth);

}

}



**2.Modify the above class to include a parameterized constructor for Box class**

import java.io.\*;

class Box

{

int width,height,depth;

Box(int w,int h,int d)

{

width=w;

height=h;

depth=d;

}

}

class Imp

{

public static void main(String arg[])

{

//Box b=new Box();

Box b=new Box(1,2,3);

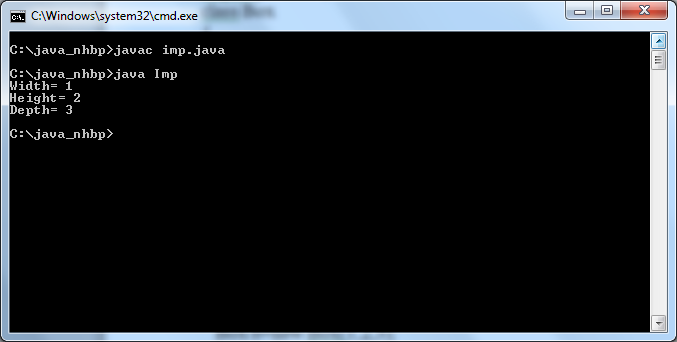
System.out.println("Width= " +b.width);

System.out.println("Height= " +b.height);

System.out.println("Depth= " +b.depth);

}

}



**3.Modify the above class to include volume method which prints the volume of the Box**

import java.io.\*;

class Box

{

int width,height,depth;

Box(int w,int h,int d)

{

width=w;

height=h;

depth=d;

}

public void volume()

{

int volume;

volume= width\*height\*depth;

System.out.println("Volume= " +volume);

}

}

class Imp

{

public static void main(String arg[])

{

//Box b=new Box();

Box b=new Box(1,2,3);

System.out.println("Width= " +b.width);

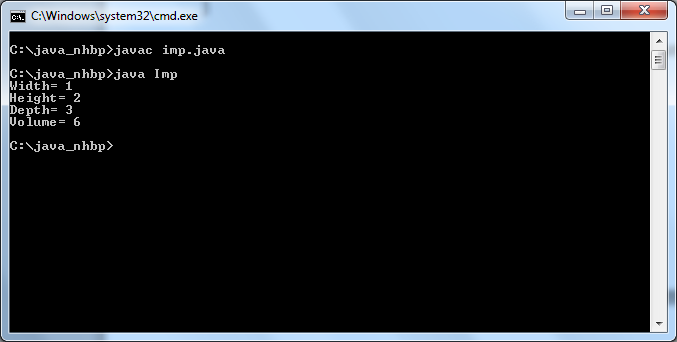
System.out.println("Height= " +b.height);

System.out.println("Depth= " +b.depth);

b.volume();

}

}



**4.Modify the above class to change the volume method to return volume instead of printing it**

import java.io.\*;

class Box

{

int width,height,depth;

Box(int w,int h,int d)

{

width=w;

height=h;

depth=d;

}

public int volume()

{

int volume;

volume= width\*height\*depth;

return volume;

}

}

class Imp

{

public static void main(String arg[])

{

//Box b=new Box();

Box b=new Box(1,2,3);

System.out.println("Width= " +b.width);

System.out.println("Height= " +b.height);

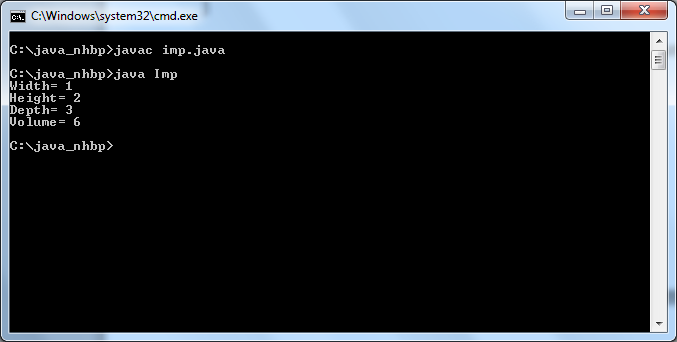
System.out.println("Depth= " +b.depth);

int v=b.volume();

System.out.println("Volume= "+v);

}

}



**5.Write a program to implement a stack**

public class MyStack {

private int maxSize;

private long[] stackArray;

private int top;

public MyStack(int s) {

maxSize = s;

stackArray = new long[maxSize];

top = -1;

}

public void push(long j) {

stackArray[++top] = j;

}

public long pop() {

return stackArray[top--];

}

public long peek() {

return stackArray[top];

}

public boolean isEmpty() {

return (top == -1);

}

public boolean isFull() {

return (top == maxSize - 1);

}

public static void main(String[] args) {

MyStack theStack = new MyStack(10);

theStack.push(10);

theStack.push(20);

theStack.push(30);

theStack.push(40);

theStack.push(50);

while (!theStack.isEmpty()) {

long value = theStack.pop();

System.out.print(value);

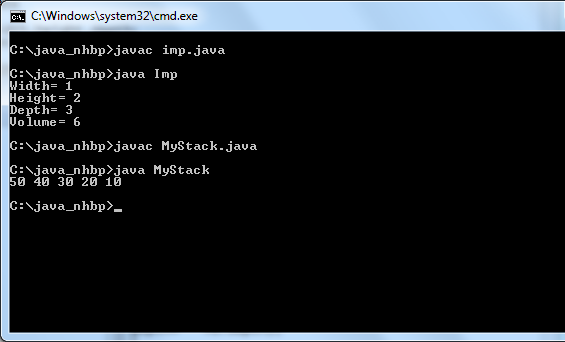
System.out.print(" ");

}

System.out.println("");

}

}



**6.Write an abstract class shape with an abstract method area .Declare classes Square,Circle which extends Shape and implement area method**

import java.util.\*;

abstract class Shape

{

abstract void area(int a);

}

class Square extends Shape

{

int area;

public void area(int a)

{

area=a\*a;

System.out.print("Area of square= "+area);

}

}

class Circle extends Shape

{

float area;

public void area(int a)

{

area=(3.14f\*a\*a);

System.out.print("\nArea of Circle= "+area);

}

}

class Abs

{

public static void main(String arg[])

{

Shape s;

s=new Square();

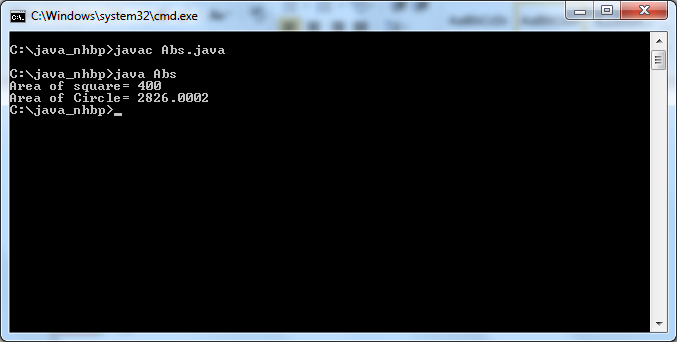
s.area(20);

s=new Circle();

s.area(30);

}

}



**Lesson-4**

**Simple programs:**

1. **Write a simple  Java program to print a Message “Hello world”**

class Xbbnhbq\_Hello

{

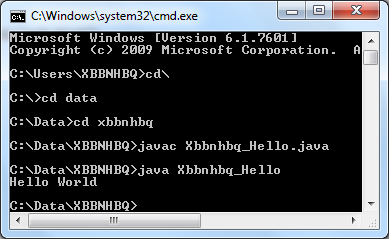
public static void main(String as[])

{

   System.out.println("Hello World");

}

}



**2.Write a Java program to find the sum of two numbers(Initialize two numbers in program)**

class Xbbnhbq\_Sum

{

public static void main(String as[])

{

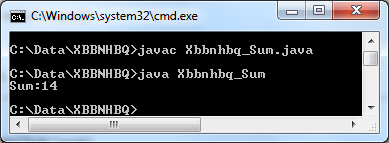
   int a=5,b=9,sum;

    sum=a+b;

   System.out.println("Sum:"+sum);

}

}



**3.Write a Java program to find the sum of two  numbers .Input must be given through command line arguments:**

class Xbbnhbq\_SumCommand

{

public static void main(String as[])

{

   int a,b,sum;

    a=Integer.parseInt(as[0]);

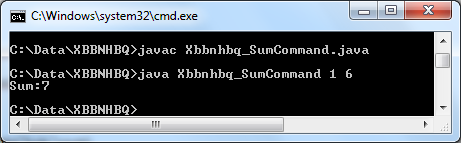
    b=Integer.parseInt(as[1]);

    sum=a+b;

   System.out.println("Sum:"+sum);

}

}



**4.Write a Java program to find sum of n numbers. Read the count n and input numbers from user:**

import java.util.\*;

class Xbbnhbq\_Sumn

{

public static void main(String as[])

{

   int n,sum=0;

System.out.println("Enter no.of elements:");

   Scanner s=new Scanner(System.in);

     n=s.nextInt();

    int[] a=new int[n];

System.out.println("Enter elements:");

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

{

    a[i]=s.nextInt();

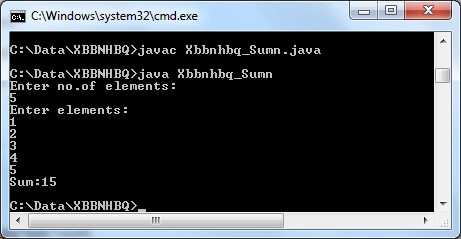
    sum+=a[i];

}

   System.out.println("Sum:"+sum);

}

}



**5.Write a Java program that uses all relational operators(=,<,>,<=,>=,!) and print  the result :**

import java.util.\*;

class Xbbnhbq\_Comp

{

public static void main(String as[])

{

   Scanner s=new Scanner(System.in);

     System.out.println("Enter 2 elements to compare:");

   int a=s.nextInt();

int b=s.nextInt();

  if(a==b)

   System.out.println(a+" is equal to "+b);

  if(a<b)

   System.out.println(a+" is lesser than "+b);

  if(a>b)

  System.out.println(a+" is greater than "+b);

  if(a<=b)

  System.out.println(a+" is lesser than or equal to "+b);

  if(a>=b)

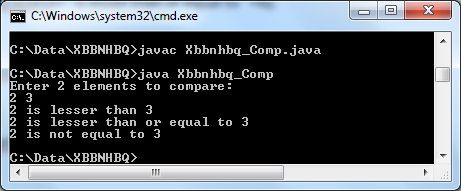
System.out.println(a+" is greater than or equal to "+b);

  if(a!=b)

System.out.println(a+" is not equal to "+b);

}

}



**6.Write a java program to find the largest of three numbers(Use Conditional statements):**

import java.util.\*;

class Xbbnhbq\_Comp3

{

public static void main(String as[])

{

   Scanner s=new Scanner(System.in);

     System.out.println("Enter 3 elements to compare:");

   int a=s.nextInt();

int b=s.nextInt();

int c=s.nextInt();

  if((a==b)&&(b==c))

   System.out.println("All are equal");

  else if((a>b)&&(a>c))

   System.out.println(a+" is greater than "+b+" and "+c);

   else if(b>c)

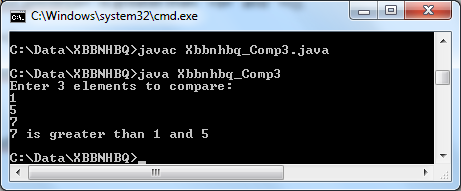
  System.out.println(b+" is greater than "+a+" and "+c);

  else

System.out.println(c+" is greater than "+a+" and "+b);

}

}



**7.  Write a Java program to Sort N numbers**

import java.util.\*;

class Xbbnhbq\_Sort

{

public static void main(String as[])

{

  Scanner s=new Scanner(System.in);

System.out.println("Enter no.of elements");

  int n=s.nextInt();

  int[] a=new int[n];

System.out.println("Enter elements:");

  for(int i=0;i<n;i++)

   a[i]=s.nextInt();

int temp;

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

{

  for(int j=i+1;j<n;j++)

{

   if(a[i]>a[j])

{

   temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

System.out.println("Sorted Elements in ascending order:");

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

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

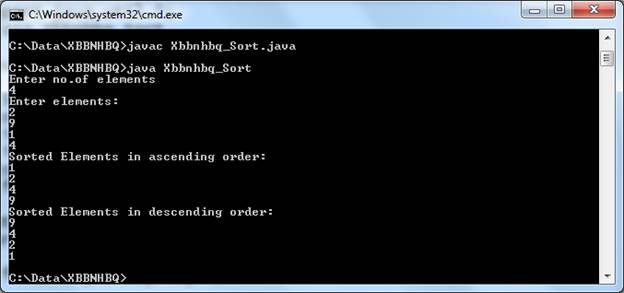
System.out.println("Sorted Elements in descending order:");

for(int i=n-1;i>=0;i--)

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

}

}



## Classes and Objects

**/\*1 .Create a class called Student with attributes name , rollno , marks and method to print the details of the student with help of creating object for the class student:\*/**

class Student

{

int rno=101,marks=97;

String name="Tesla";

void print()

{

System.out.println("Roll no= "+rno+" Marks= "+marks+" Name= "+name);

}

}

class StudInfo

{

public static void main(String arg[])

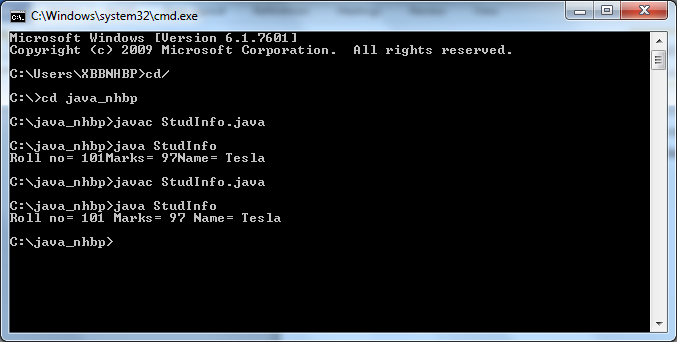
{

Student s=new Student();

s.print();

}

}



**2. Create a class called Employee with attributes employee name, empno, salary, designation. Create two objects for the class Employee and calculate the total salary of two employees and print the individual employee details with total salary of two employees :**

import java.util.\*;

class Employee

{

int empno,salary;

String name,desig;

Scanner sc=new Scanner(System.in);

void getDetails()

{

System.out.println("Enter details");

empno=sc.nextInt();

name=sc.next();

desig=sc.next();

salary=sc.nextInt();

}

void putDetails()

{

System.out.println("Empno= "+empno);

System.out.println("Name= "+name);

System.out.println("Desig= "+desig);

System.out.println("Salary= "+salary);

}

int sal;

sal+=salary;

System.out.println(“Total salary= ”+sal);

}

class Emp

{

public static void main(String arg[])

{

Employee e,e1;

e=new Employee();

e.getDetails();

e.putDetails();

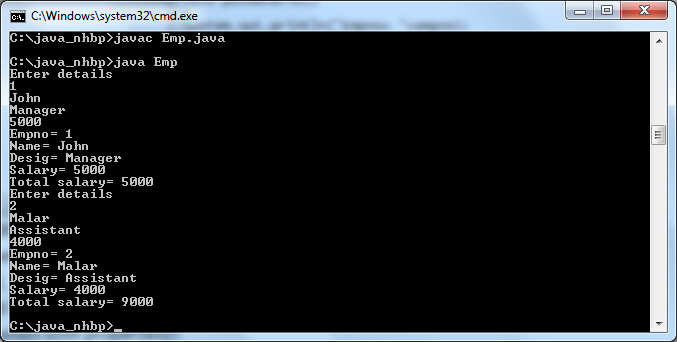
e1=new Employee();

e.getDetails();

e.putDetails();

}

}



**Parameterized Constructor & Method Passing :-**

1. Create a class Number with attributes fno, sno, tno. Initialize the attributes using constructor (using parameterized constructor) and a method to find the sum of three numbers and display method to print the total:

import java.util.\*;

class Number

{

int fno,sno,tno,sum;

Number(int f,int s,int t)

{

fno=f;

sno=s;

tno=t;

}

void cons()

{

sum=fno+tno+sno;

System.out.print(sum);

}

}

class Orig

{

public static void main(String arg[])

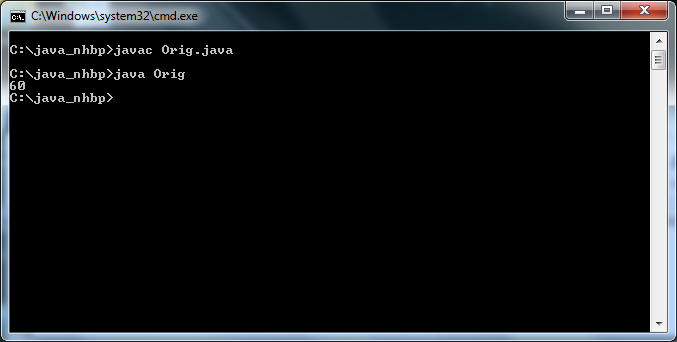
{

Number n=new Number(10,20,30);

n.cons();

}

}



**Create a class called Sort with attribute an array(integer) and an input method that takes count(n) from main method as argument which reads array numbers and another method which computes sorting of integer arrays:**

import java.io.\*;

import java.util.\*;

class Sort

{

Scanner sc=new Scanner(System.in);

int num,temp;

int a[]=new int[10];

void count(int n)

{

num=n;

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

{

a[i]=sc.nextInt();

}

}

void sorting()

{

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

{

for(int j=i+1;j<num;j++)

{

if(a[i]>a[j])

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

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

{

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

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

}

}

}

class Sort1

{

public static void main(String arg[])

{

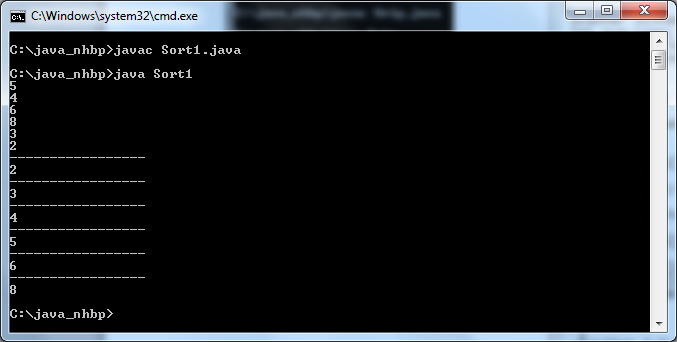
Sort s=new Sort();

s.count(6);

s.sorting();

}

}



**Create a Class A with attributes a, b. Create an object o1 and initialize attributes using constructor and create another object o2 and use copy constructor that copies the attributes of o1 to attributes of o2:-**

class A

{

int a,b;

A(int a,int b)

{

this.a=a;

this.b=b;

}

A(A o)

{

System.out.println(o.a);

System.out.println(o.b);

}

}

class Main1

{

public static void main(String s[])

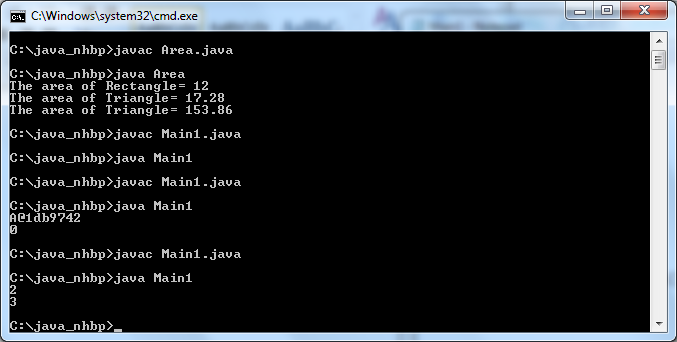
{

A o1=new A(2,3);

A o2=new A(o1);

}

}



**Method OverLoading:**

**Create a class called shape with attributes length, breadth, and radius. Create methods that differs with number of arguments, type of arguments, sequence of arguments to calculate area of rectangle , circle, square:**

import java.util.\*;

class Rect

{

int a;

double b,c;

int area(int len,int bre)

{

a=len\*bre;

return a;

}

double area(double len,double bre)

{

b=(len\*bre)/2;

return b;

}

double area(int len)

{

c=3.14\*(len\*len);

return c;

}

}

class Area

{

public static void main(String arg[])

{

int a1;

double a2,a3;

Rect r=new Rect();

a1=r.area(4,3);

a2=r.area(6.4,5.4);

a3=r.area(7);

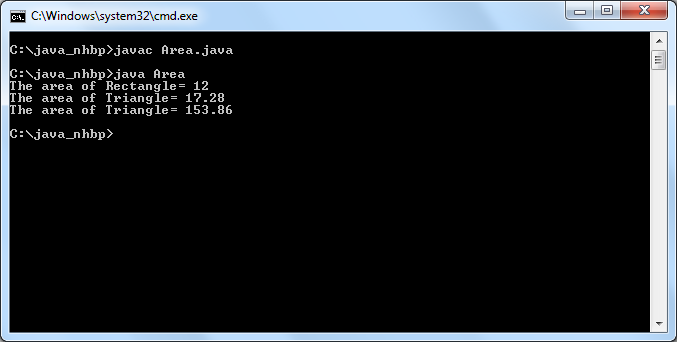
System.out.println("The area of Rectangle= "+a1);

System.out.println("The area of Triangle= "+a2);

System.out.println("The area of Triangle= "+a3);

}

}



**Inheritance:**

1. **Create a class called vehicle with attributes colour, vehicleno, modelno. Create another class called Bus which extends vehicle class with its own properties and a method which displays both properties (vehicle, Bus):- (Single inheritance).**

**Create a class called Bike which extends vehicle class with its properties and a method which displays both properties(vehicle, Bike)[Multiple Inheritance]:**

**Create a Class called cycle which extends from Bike with its own properties and a method which displays details of both cycle and Bike [Multilevel Inheritance]:**

class Vehicle

{

int veh\_no=1266,mod\_no=27;

String colour="Bisque";

int num;

String col;

Vehicle(int numb,String clr)

{

num=numb;

col=clr;

}

void display()

{

System.out.println(" "+veh\_no+" "+mod\_no+" "+colour+" "+num+" "+col);

}

}

class Bus extends Vehicle

{

Bus(int numb,String clr)

{

super(numb,clr);

}

}

class Bike extends Vehicle

{

Bike(int numb,String clr)

{

super(numb,clr);

}

}

class Cycle extends Bike

{

Cycle(int numb,String clr)

{

super(numb,clr);

}

}

class Inherit

{

public static void main(String arg[])

{

Vehicle v;

v=new Bus(2134,"Brown");

v.display();

v=new Bike(2174,"Black");

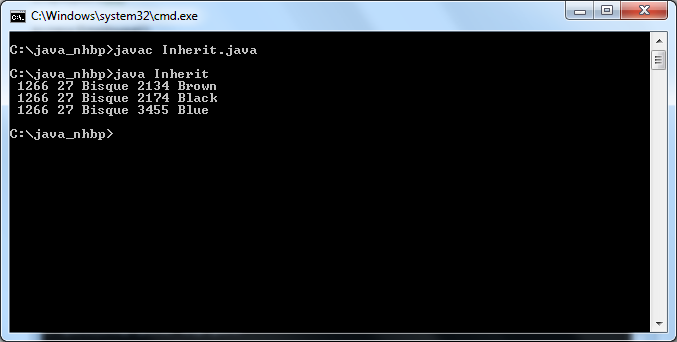
v.display();

v=new Cycle(3455,"Blue");

v.display();

}

}



**Method Overriding:**

**1.Create a class called Sample with method name “print” and another class Sample1 with same method name. Create an object for Sample1 that invoke the method in Sample1 which overrides the method in base class:**

class Sample

{

int a=45;

void print()

{

System.out.println(a);

}

}

class Sample1 extends Sample

{

int b=30;

void print()

{

System.out.println(b);

}

}

class Sam

{

public static void main(String arg[])

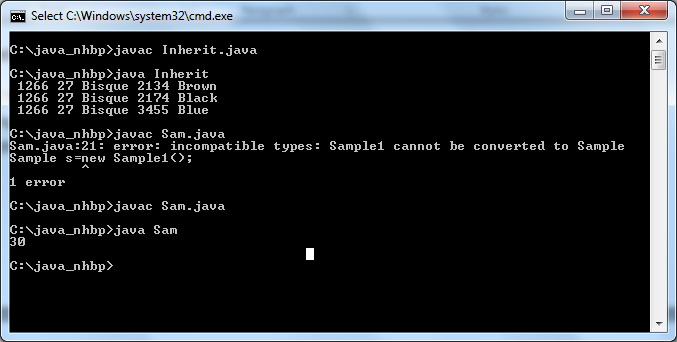
{

Sample s=new Sample1();

s.print();

}

}



**Interfaces:**

1. **Create an interface called shape that contains only method signature [method name – area , input-radius].Create a class called Rectangle that implements from shape interface and a method which contains the implementation for the method of interface shape-calculating area of rectangle:**

interface Shape

{

void area(int length,int breadth);

}

class Rectangle implements Shape

{

float area;

public void area(int length,int breadth)

{

area=length\*breadth;

System.out.println(area);

}

}

class InterDemo

{

public static void main(String s[])

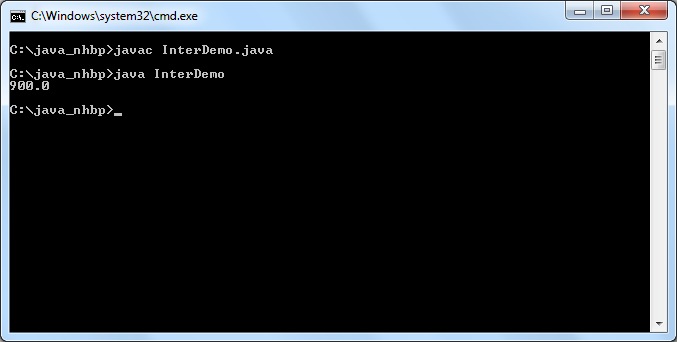
{

Rectangle r=new Rectangle();

r.area(45,20);

}

}



**1.Write the interface called GeometricObject, which declares two abstract methods: getParameter() and getArea()**

**Write the implementation class Circle, with a protected variable radius, which implements the interface GeometricObject**

**Write a test program called TestCircle to test the methods defined in Circle**

import java.util.\*;

interface GeometricObject

{

void getParameter();

void getArea();

}

class Circle implements GeometricObject

{

int rad;

float area;

Scanner sc=new Scanner(System.in);

public void getParameter()

{

System.out.println("Enter Radius");

rad=sc.nextInt();

}

public void getArea()

{

area=(3.14f\*rad\*rad);

System.out.println("Area= "+area);

}

}

class TestCircle

{

public static void main(String arg[])

{

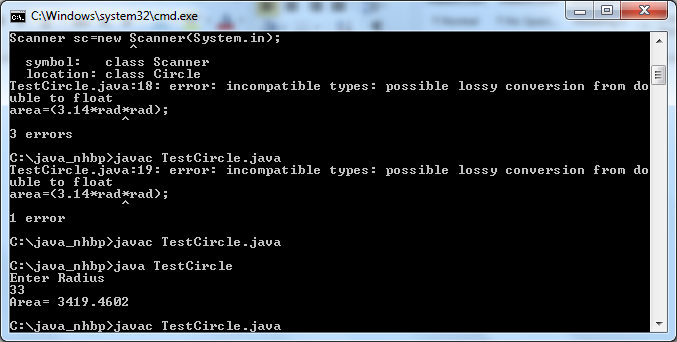
Circle c=new Circle();

c.getParameter();

c.getArea();

}

}



**2.The class ResizableCircle is defined as a subclass of the class Circle, which also implements an interface called Resizable. The interface Resizable declares an abstract method resize(), which modifies the dimension (such as radius) by the given percentage. Write the interface Resizable and the class ResizableCircle.**

**Write a test program called TestResizableCircle to test the methods defined in ResizableCircle.**

import java.util.\*;

interface GeometricObject

{

void getParameter();

void getArea();

}

interface Resizable

{

void resize(int perc);

}

class Circle implements GeometricObject

{

int rad;

float area;

Scanner sc=new Scanner(System.in);

public void getParameter()

{

System.out.println("Enter Radius");

rad=sc.nextInt();

}

public void getArea()

{

area=(3.14f\*rad\*rad);

System.out.println("Area= "+area);

}

}

class ResizableCircle extends Circle implements Resizable

{

public void resize(int perc)

{

rad+=(rad\*perc)/100;

System.out.println("Increased Radius= "+rad);

}

}

class TestResizable

{

public static void main(String arg[])

{

int per;

ResizableCircle r=new ResizableCircle();

r.getParameter();

r.getArea();

System.out.println("Enter percent");

Scanner sc=new Scanner(System.in);

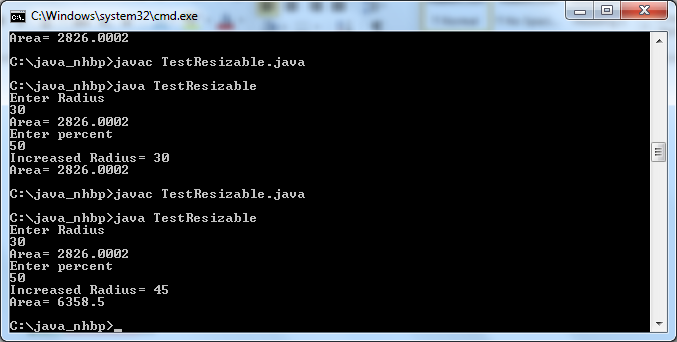
per=sc.nextInt();

r.resize(per);

r.getArea();

}

}



**Java Ex on Arrays**

**1) Given an array of ints, return the number of times that two 6's are next to each other in**

**the array. Also count instances where the second "6" is actually a 7.**

**array667([6, 6, 2]) ? 1**

**array667([6, 6, 2, 6]) ? 1**

**array667([6, 7, 2, 6]) ? 1**

import java.io.\*;

import java.util.\*;

class Array6

{

public static void main(String s[])

{

Scanner sc=new Scanner(System.in);

int[] a=new int[5];

int i;

int count=0;

System.out.println("Enter the array values");

for(i=0;i<5;i++)

{

a[i]=sc.nextInt();

}

count=array667(a);

System.out.println("The count is "+count);

}

public static int array667(int[] nums)

{

int count=0;

for(int i=0;i<nums.length-1;i++)

{

if (nums[i]==6 && (nums[i+1]==6 || nums[i+1]==7))

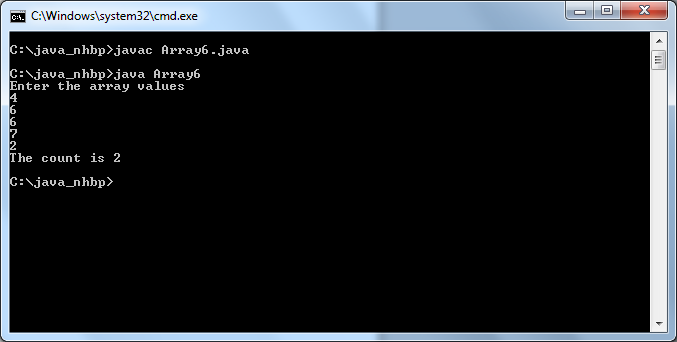
count++;

}

return count;

}

}



**2) Given an array of ints, we'll say that a triple is a value appearing 3 times in a row in the array. Return true if the array does not contain any triples.**

**noTriples([1, 1, 2, 2, 1]) ? true**

**noTriples([1, 1, 2, 2, 2, 1]) ? false**

**noTriples([1, 1, 1, 2, 2, 2, 1]) ? false**

import java.io.\*;

import java.util.\*;

class Triples

{

public static void main(String s[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

int[] a=new int[n];

int i,flag;

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

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

{

a[i]=sc.nextInt();

}

flag=triple(a);

if(flag==1)

{

System.out.println("False");

}

else

{

System.out.println("True");

}

}

public static int triple(int[] value)

{

int count=0;

for(int i=0;i<value.length-1;i++)

{

if((value[i]==value[i+1])&&(value[i+1]==value[i+2]))

{

count++;

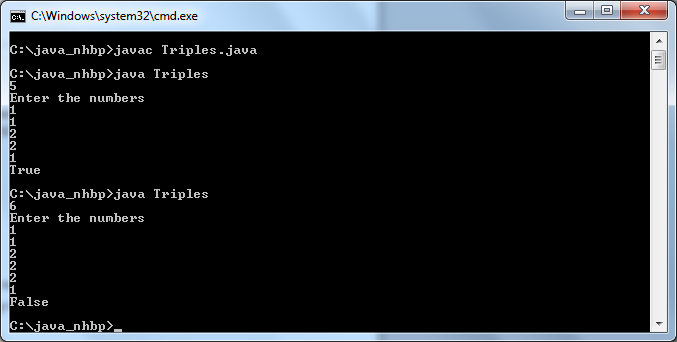
}

}

return count;

}

}



**3)Given an array of ints, return true if it contains a 2, 7, 1 pattern -- a value, followed by the value plus 5,**

**followed by the value minus 1. Additionally the 271 counts even if the "1" differs by 2 or less from the correct value.**

**has271([1, 2, 7, 1]) ? true**

**has271([1, 2, 8, 1]) ? false**

**has271([2, 7, 1]) ? true**

import java.io.\*;

import java.util.\*;

class Pattern

{

public static void main(String s[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

int[] a=new int[n];

int i,flag;

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

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

{

a[i]=sc.nextInt();

}

flag=patt(a);

if(flag==1)

{

System.out.println("True");

}

else

{

System.out.println("False");

}

}

public static int patt(int[] value)

{

int count=0;

for(int i=0;i<value.length-1;i++)

{

if((value[i]+5==value[i+1])&&(value[i]-1==value[i+2]))

{

count++;

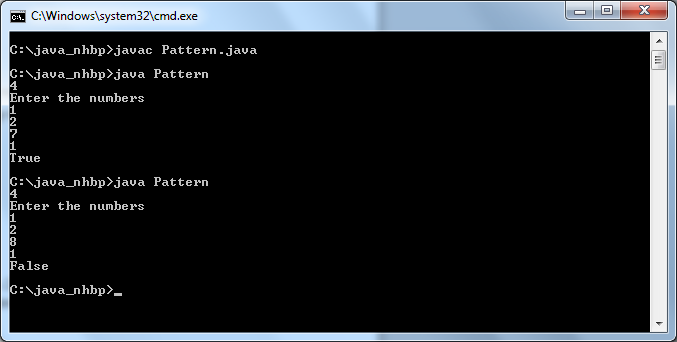
}

}

return count;

}

}



**4)Given three ints, a b c, return true if one of b or c is "close" (differing from a by at most 1), while the other is "far",**

**differing from both other values by 2 or more. Note: Math.abs(num) computes the absolute value of a number.**

**closeFar(1, 2, 10) ? true**

**closeFar(1, 2, 3) ? false**

**closeFar(4, 1, 3) ? true**

import java.io.\*;

import java.util.\*;

class Close

{

public static void main(String s[])

{

Scanner sc=new Scanner(System.in);

int[] a=new int[3];

int i;

int count=0;

System.out.println("Enter the array values");

for(i=0;i<3;i++)

{

a[i]=sc.nextInt();

}

count=closefar(a);

if(count==1)

{

System.out.println("False");

}

else

{

System.out.println("True");

}

}

public static int closefar(int[] nums)

{

int count=0;

int n=nums.length-1;

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

{

if (((nums[i])-(nums[i+1]))<=1)

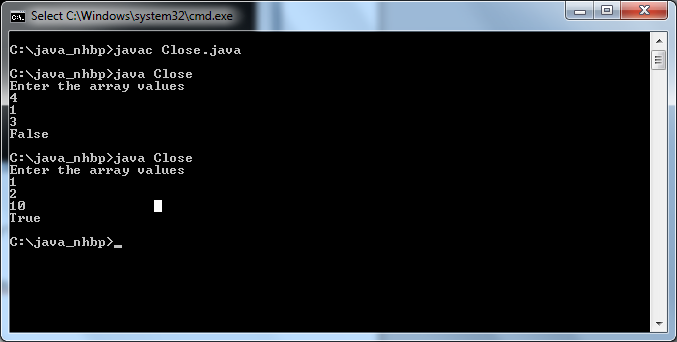
count++;

}

return count;

}

}



**5) For this problem, we'll round an int value up to the next multiple of 10 if its rightmost digit is 5 or more, so 15 rounds up to 20.**

**Alternately, round down to the previous multiple of 10 if its rightmost digit is less than 5, so 12 rounds down to 10. Given 3 ints, a b c,**

**return the sum of their rounded values. To avoid code repetition, write a separate helper "public int round10(int num) {" and call it 3 times.**

**Write the helper entirely below and at the same indent level as roundSum().**

**roundSum(16, 17, 18) ? 60**

**roundSum(12, 13, 14) ? 30**

**roundSum(6, 4, 4) ? 10**

import java.io.\*;

import java.util.\*;

class Round

{

public static void main(String s[])

{

Scanner sc=new Scanner(System.in);

int n,sum=0;

n=sc.nextInt();

int[] a=new int[n];

int[] c=new int[n];

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

{

a[i]=sc.nextInt();

}

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

{

c[i]=round10(a[i]);

}

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

{

sum=sum+c[i];

}

System.out.println("The sum is "+sum);

}

public static int round10(int a)

{

int n;

if((a%10)<5)

n=((a/10)\*10);

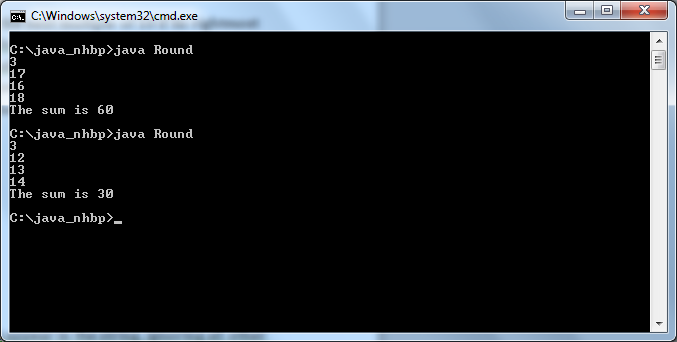
else

n=((a/10)\*10)+10;

return n;

}

}



**6)Given a string, return the sum of the digits 0-9 that appear in the string,**

**ignoring all other characters. Return 0 if there are no digits in the string.**

**(Note: Character.isDigit(char) tests if a char is one of the chars '0', '1', .. '9'.**

**Integer.parseInt(string) converts a string to an int.)**

**sumDigits("aa1bc2d3") ? 6**

**sumDigits("aa11b33") ? 8**

**sumDigits("Chocolate") ? 0**

import java.io.\*;

import java.util.\*;

class Digits

{

public static void main(String arg[])

{

Scanner sc=new Scanner(System.in);

String s;

int ad,num=0;

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

s=sc.next();

int n;

n=s.length();

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

{

char ch=s.charAt(i);

if(Character.isDigit(ch))

{

num=num+Character.getNumericValue(ch);

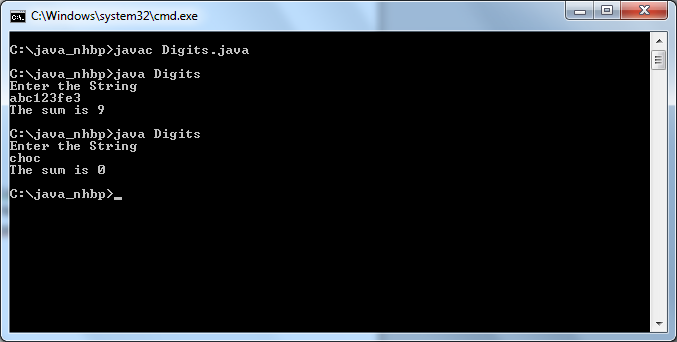
}

}

System.out.println("The sum is "+num);

}

}



**7) Given a string, look for a mirror image (backwards) string at both the beginning and end of**

**the given string. In other words, zero or more characters at the very begining of the given string,**

**and at the very end of the string in reverse order (possibly overlapping).**

**For example, the string "abXYZba" has the mirror end "ab".**

**mirrorEnds("abXYZba") ? "ab"**

**mirrorEnds("abca") ? "a"**

**mirrorEnds("aba") ? "aba"**

import java.io.\*;

import java.util.\*;

class Rev

{

public static void main(String ag[])

{

String s;

Scanner sc=new Scanner(System.in);

s=sc.next();

String a="";

int j=0;

char[] ch=s.toCharArray();

for(int i=ch.length-1;i>=0;i--)

{

a+=s.charAt(i);

}

char[] ch1=a.toCharArray();

int n=ch.length-1;

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

{

if(ch[i]==ch1[i])

{

j++;

}

else

{

break;

}

}

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

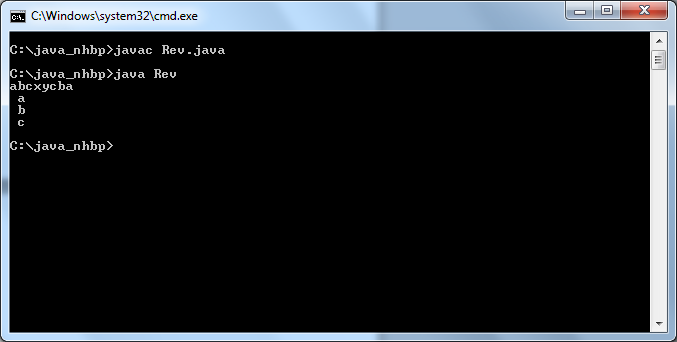
{

System.out.println(" "+ch[i]);

}

}

}



**8)Return an array that contains the exact same numbers as the given array, but rearranged so that all the zeros are grouped at the start of the array.**

**The order of the non-zero numbers does not matter. So {1, 0, 0, 1} becomes {0 ,0, 1, 1}. You may modify and return the given array or make a new array.**

**zeroFront([1, 0, 0, 1]) ? [0, 0, 1, 1]**

**zeroFront([0, 1, 1, 0, 1]) ? [0, 0, 1, 1, 1]**

**zeroFront([1, 0]) ? [0, 1]**

import java.io.\*;

import java.util.\*;

class Zero

{

public static void main(String arg[])

{

int n,f;

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

int[] a=new int[n];

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

{

a[i]=sc.nextInt();

}

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

{

for (int j = i + 1; j < n; j++)

{

if (a[i] > a[j])

{

f = a[i];

a[i] = a[j];

a[j] = f;

}

}

}

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

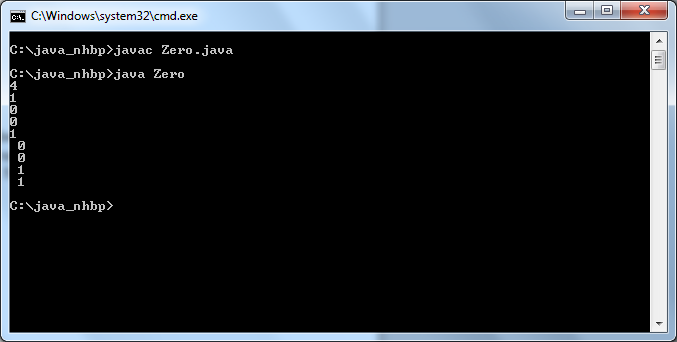
{

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

}

}

}



**9)Given an array of ints, return true if the number of 1's is greater than the number of 4's**

**more14([1, 4, 1]) → true**

**more14([1, 4, 1, 4]) → false**

**more14([1, 1]) → true**

import java.io.\*;

import java.util.\*;

class More14

{

public static void main(String arg[])

{

int n,count=0,count1=0;

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

int[] a=new int[n];

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

{

a[i]=sc.nextInt();

}

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

{

if(a[i]==1)

{

count++;

}

else if(a[i]==4)

{

count1++;

}

}

if(count>count1)

{

System.out.println("TRUE");

}

else

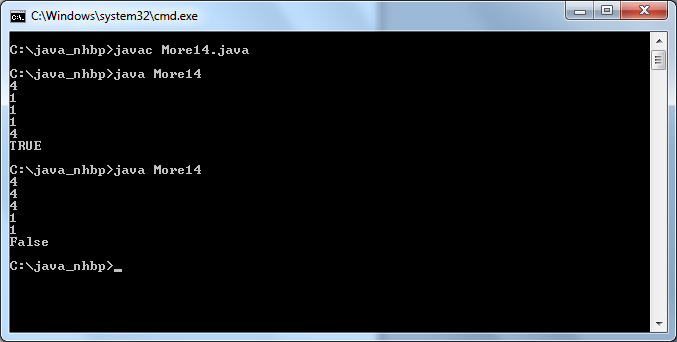
{

System.out.println("False");

}

}

}



**10)Return the "centered" average of an array of ints, which we'll say is the mean average of the values, except ignoring the largest and smallest values in the array.**

**If there are multiple copies of the smallest value,ignore just one copy, and likewise for the largest value. Use int division to produce the final average.**

**You may assume that the array is length 3 or more.**

**centeredAverage([1, 2, 3, 4, 100]) → 3**

**centeredAverage([1, 1, 5, 5, 10, 8, 7]) → 5**

**centeredAverage([-10, -4, -2, -4, -2, 0]) → -3**

import java.io.\*;

import java.util.\*;

class CenAvg

{

public static void main(String arg[])

{

int n,max,min,sum=0,avg=0;

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

int[] a=new int[n];

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

{

a[i]=sc.nextInt();

}

max=a[0];

min=a[0];

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

{

if(max>a[i])

{

max=a[i];

}

if(min<a[i])

{

min=a[i];

}

}

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

{

if(a[i]==max)

a[i]=0;

}

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

{

if(a[i]==min)

a[i]=0;

}

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

{

sum+=a[i];

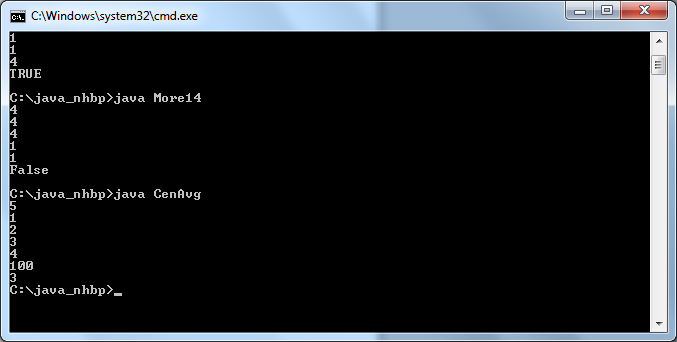
}

avg=sum/(n-2);

System.out.print(avg);

}

}



**11) Given a string, return recursively a "cleaned" string where adjacent chars that are the same have been reduced to a single char. So "yyzzza" yields "yza".**

**stringClean("yyzzza") → "yza"**

**stringClean("abbbcdd") → "abcd"**

**stringClean("Hello") → "Helo"**

import java.io.\*;

import java.util.\*;

class Duplic

{

public static void main(String arg[])

{

String s;

Scanner sc=new Scanner(System.in);

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

String ans="";

s=sc.next();

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

{

if(i==0)

{

ans+=s.charAt(i);

}

else if(s.charAt(i-1)!=s.charAt(i))

{

ans+=s.charAt(i);

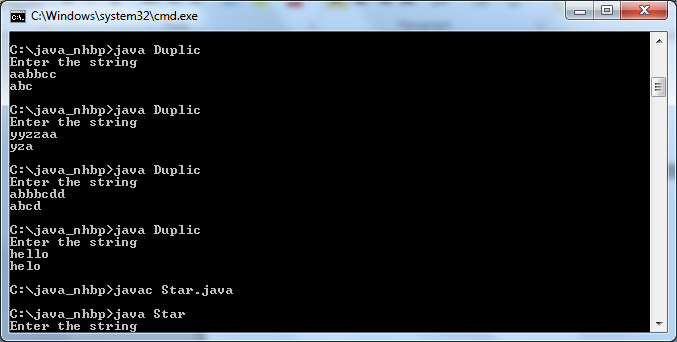
}

}

System.out.println(ans);

}

}



**12)Given a string, compute recursively a new string where identical chars that are adjacent in the original string are separated from each other by a "\*".**

**pairStar("hello") → "hel\*lo"**

**pairStar("xxyy") → "x\*xy\*y"**

**pairStar("aaaa") → "a\*a\*a\*a"**

import java.io.\*;

import java.util.\*;

class Star

{

public static void main(String arg[])

{

String s;

Scanner sc=new Scanner(System.in);

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

String ans="";

s=sc.next();

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

{

if(i==0)

{

ans+=s.charAt(i);

}

else if(s.charAt(i-1)!=s.charAt(i))

{

ans+=s.charAt(i);

}

else if(s.charAt(i-1)==s.charAt(i))

{

ans+="\*"+s.charAt(i);

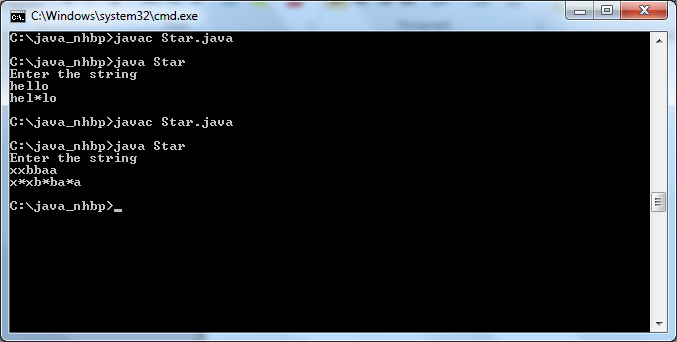
}

}

System.out.println(ans);

}

}



**13)Given a non-empty array, return true if there is a place to split the array so that the sum of the numbers on one side is equal to the sum of the numbers on the other side.**

**canBalance([1, 1, 1, 2, 1]) → true**

**canBalance([2, 1, 1, 2, 1]) → false**

**canBalance([10, 10]) → true**

import java.io.\*;

import java.util.\*;

class CanBal

{

public static void main(String arg[])

{

int n,sum=0,sum1=0;

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

int[] a=new int[n];

int[] b=new int[n];

int[] c=new int[n];

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

{

a[i]=sc.nextInt();

}

if(n%2!=0)

{

for(int i=0;i<=n/2;i++)

{

b[i]=a[i];

}

for(int i=(n/2)+1;i<n;i++)

{

c[i]=a[i];

}

for(int i=0;i<=n/2;i++)

{

sum+=b[i];

}

for(int i=(n/2)+1;i<n;i++)

{

sum1+=c[i];

}

if(sum==sum1)

{

System.out.println("True");

}

else

{

System.out.println("False");

}

}

if(n%2==0)

{

for(int i=0;i<n/2;i++)

{

b[i]=a[i];

}

for(int i=(n/2);i<n;i++)

{

c[i]=a[i];

}

for(int i=0;i<n/2;i++)

{

sum+=b[i];

}

for(int i=(n/2);i<n;i++)

{

sum1+=c[i];

}

if(sum==sum1)

{

System.out.println("True");

}

else

{

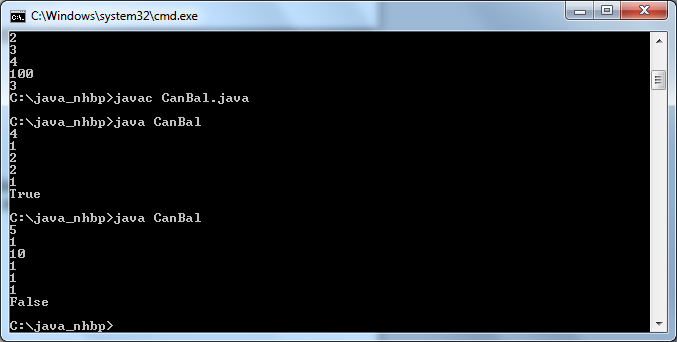
System.out.println("False");

}

}

}

}



**Exceptions**

**1.Write a custom exception class InsufficientBalanceException. Write a class which has withDrawMoney method which takes amountToWithDraw as parameter.Inside the method, compare the balance with amountToWithdraw. If balance is less than amountToWithdraw then raise an exception InsufficientBalanceException.**

import java.util.\*;

class InsufficientBalanceException extends Exception

{

InsufficientBalanceException(String s)

{

super(s);

}

}

class WithDraw

{

double bal;

void getBal()

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter balance");

bal=sc.nextDouble();

}

void withDrawMoney(double atwd)throws InsufficientBalanceException

{

if(bal<=atwd)

{

throw new InsufficientBalanceException("You Cannot WithDraw Amount ur balance is less than the amount u need!!!");

}

else

{

System.out.println(" "+(bal-atwd));

}

}

}

class ExceptionFirst

{

public static void main(String s[])

{

double amt;

Scanner sc=new Scanner(System.in);

WithDraw wd=new WithDraw();

wd.getBal();

System.out.println("Enter Withdraw");

amt=sc.nextDouble();

try

{

wd.withDrawMoney(amt);

}

catch(Exception m)

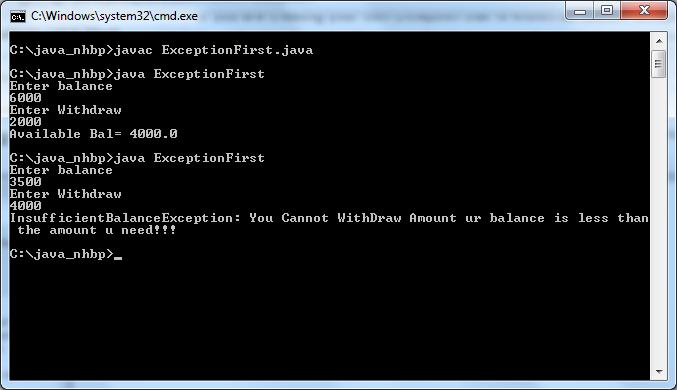
{

System.out.println(m);

}

}

}



**2.Write a program to have try catch and finally blocks for the following exceptions**

**1.Divide by zero**

**2.Accessing array with indices**

**3.Call a method of using a null variable**

import java.util.\*;

class ExceptionTwo

{

public static void main(String s[])

{

try

{

int a[]={1,2};

Scanner sc=new Scanner(System.in);

System.out.print("Enter the value to divide");

int n=sc.nextInt();

a[0]=a[1]/n;

}

catch(ArithmeticException e)

{

System.out.println("ArithmeticException Caught:"+e);

}

try

{

int a[]={1,2};

a[1]=a[5];

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println("ArrayIndexOutOfBound Caught:"+e);

}

try

{

String str=null;

if(str.equals("asdf"))

System.out.println("NO Null");

}

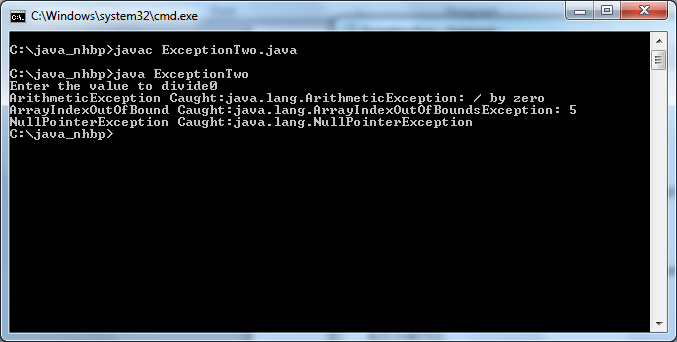
catch(NullPointerException e)

{

System.out.print("NullPointerException Caught:"+e);

}

}

} 

**3.Write a program without try catch block for the above program, write code to avoid the above runtime exceptions (clue: check the null value before make a call)**

import java.util.\*;

class ExceptionThree

{

public static void main(String s[])

{

int a[]={1,2};

String str;

Scanner sc=new Scanner(System.in);

str=sc.nextLine();

if(str.equals("null"))

System.out.println("Not valid");

else

System.out.println("valid");

System.out.print("Enter the value to divide");

int n=sc.nextInt();

if(n!=0)

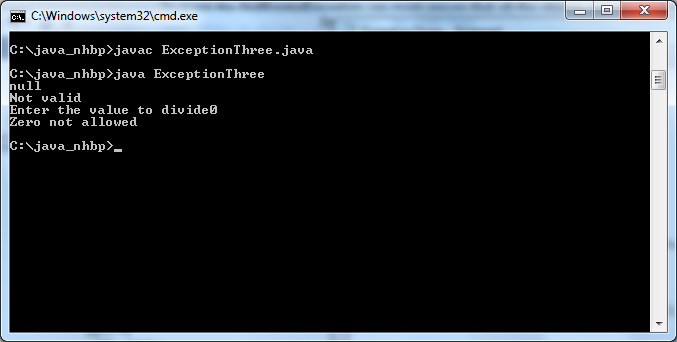
a[2]=a[1]/n;

else

System.out.println("Zero not allowed");

}

}



**1.Develope a Java application that implements Stack which throws corresponding exceptions when trying to push items(stack is full) and when trying to pop items(stack is empty):-**

import java.util.\*;

class ArrayStack

{

protected int arr[];

protected int top,size,len;

public ArrayStack(int n)

{

size=n;

len=0;

top=-1;

arr=new int[n];

}

public int getSize()

{

return size;

}

public boolean isEmpty()

{

if(top==-1)

return true;

else

return false;

}

public boolean isFull()

{

if(top==size-1)

return true;

else

return false;

}

public int peek()

{

if(isEmpty())

throw new NoSuchElementException("Underflow Exception");

return arr[top];

}

public void push(int i)

{

if(top+1>=size)

throw new ArrayIndexOutOfBoundsException("Overflow Exception");

if(top+1<size)

arr[++top]=i;

len++;

}

public int pop()

{

if(isEmpty())

throw new NoSuchElementException("Underflow Exception");

len--;

return arr[top--];

}

public void display()

{

System.out.println("\nStack = ");

if(len==0)

{

System.out.println("Empty");

return;

}

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

System.out.println(arr[i]+" ");

System.out.println();

}

}

class StackException

{

public static void main(String args[])

{

Scanner s=new Scanner(System.in);

System.out.println("\nStack Test\n");

System.out.println("Enter the size of the stack: ");

int n=s.nextInt();

ArrayStack stk=new ArrayStack(n);

char ch;

do{

System.out.println("\n Stack Operation");

System.out.println("1. Push");

System.out.println("2. Pop");

System.out.println("3. Peep");

System.out.println("4. Check Empty");

System.out.println("5. Check Full");

System.out.println("6. Size");

int choice=s.nextInt();

switch(choice)

{

case 1:

System.out.println("Enter the element to push");

try

{

stk.push(s.nextInt());

}

catch(Exception e)

{

System.out.println("Error"+e.getMessage());

}

break;

case 2:

try

{

System.out.println("Popped Element:"+stk.pop());

}

catch(Exception e)

{

System.out.println("Error"+e.getMessage());

}

break;

case 3:

try

{

System.out.println("Peek Element:"+ stk.peek());

}

catch(Exception e)

{

System.out.println("Error"+e.getMessage());

}

break;

case 4:

System.out.println("Empty status:"+stk.isEmpty());

break;

case 5:

System.out.println("Full Stack:"+stk.isFull());

break;

case 6:

System.out.println("Size:"+stk.getSize());

break;

default:

System.out.println("Wrong Entry\n");

break;

}

stk.display();

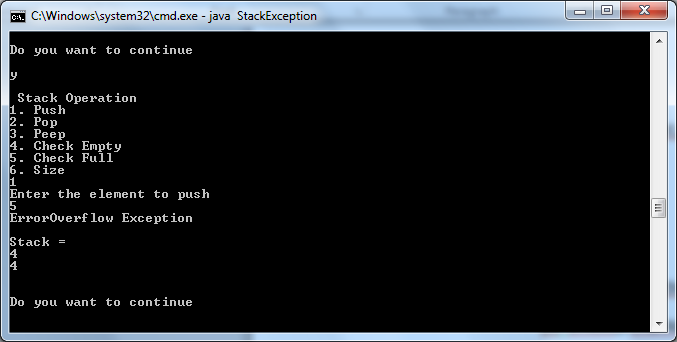
System.out.println("\nDo you want to continue\n");

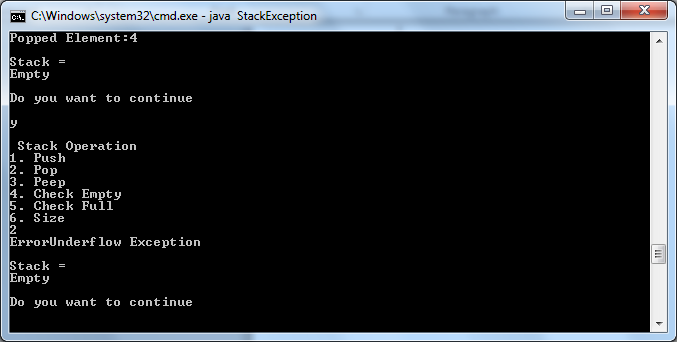
ch=s.next().charAt(0);

}while(ch=='Y'||ch=='y');

}

}





**2. Write an example that will throw your own exception explicitly using throw keyword (UserDefined Exception):-**

import java.util.\*;

class UserDefException extends Exception

{

UserDefException(String s)

{

super(s);

}

}

class UserDefined

{

public static void main(String s[])throws UserDefException

{

try

{

int n;

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

if(n>100)

throw new UserDefException("Value Should not exceed 100");

}

catch(Exception e)

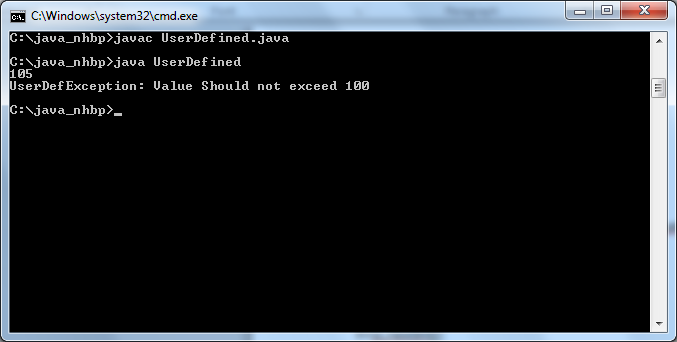
{

System.out.println(e);

}

}

}



1. **Write a program that asks a user for a file name and prints the number of characters, words, and lines in that file.**

import java.util.\*;

import java.io.\*;

public class FileInfoPrinter

{

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

{

Scanner console = new Scanner(System.in);

System.out.println("File to be read: ");

String inputFile = console.next();

File file = new File(inputFile);

Scanner in = new Scanner(file);

int words = 0;

int lines = 0;

int chars = 0;

while(in.hasNext())

{

in.next();

words++;

}

while(in.hasNextLine())

{

in.nextLine();

lines++;

}

while(in.hasNext())

{

chars += in.next().length();

}

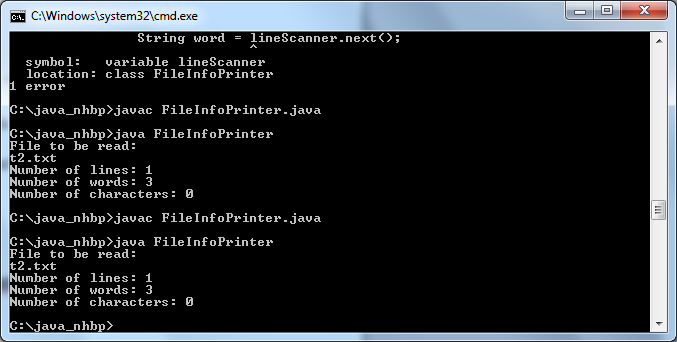
System.out.println("Number of lines: " + lines);

System.out.println("Number of words: " + words);

System.out.println("Number of characters: " + chars);

}

}



1. **Write an application that uses BufferedReader and BufferedWriter classes**

import java.io.\*;

public class Buffer {

public static void main(String[] argv) throws Exception {

BufferedReader br = new BufferedReader(new FileReader("t7.txt"));

BufferedWriter bw = new BufferedWriter(new FileWriter("nhbp.txt"));

int i;

do {

i = br.read();

if (i != -1) {

bw.write((char) i);

}

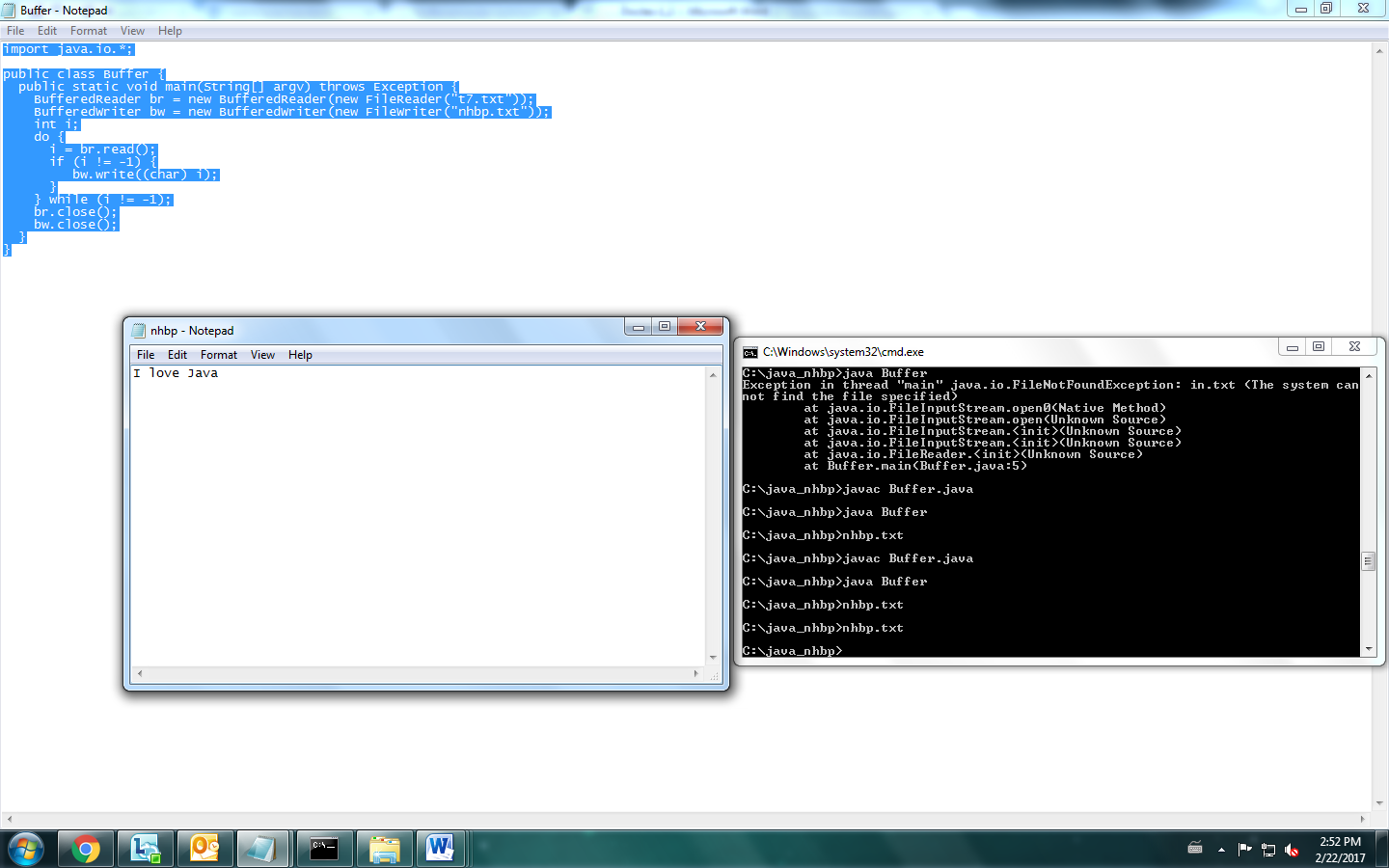
} while (i != -1);

br.close();

bw.close();

}

}



**Threads:**

1. **Create a thread using both thread class and runnable interface.**

class Multi1 extends Thread

{

public void run()

{

System.out.println("thread is running on Class Multi one...");

}

}

class Multi2 implements Runnable

{

public void run()

{

System.out.println("thread is running Class Multi Two...");

}

}

class Multi

{

public static void main(String args[])

{

Multi1 t1=new Multi1();

t1.start();

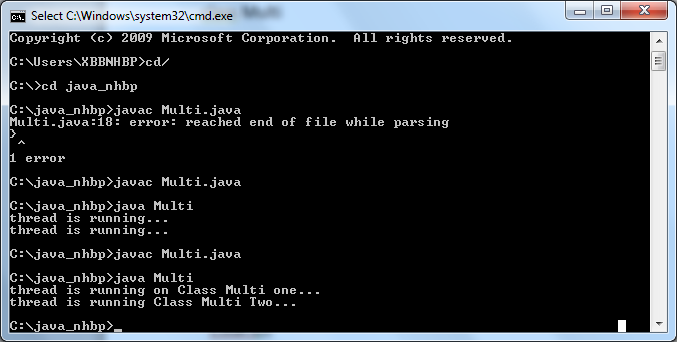
Multi2 m1=new Multi2();

Thread t =new Thread(m1);

t.start();

}

}



1. **Develop a Java application that handles deadlock problem using synchronization.**
2. **Develop an application that uses thread and set the priority of all created threads and trace the execution of threads based on its priority.**

class MultiPri extends Thread

{

public void run()

{

System.out.println(Thread.currentThread().getName()+" "+Thread.currentThread().getPriority());

}

}

class MultiThreadPriDemo

{

public static void main(String s[])

{

MultiPri t1=new MultiPri();

t1.setName("Thread\_1");

t1.setPriority(2);

MultiPri t2=new MultiPri();

t2.setName("Thread\_2");

t2.setPriority(5);

MultiPri t3=new MultiPri();

t3.setName("Thread\_3");

t3.setPriority(1);

MultiPri t4=new MultiPri();

t4.setName("Thread\_4");

t4.setPriority(4);

MultiPri t5=new MultiPri();

t5.setName("Thread\_5");

t5.setPriority(3);

t1.start();

t2.start();

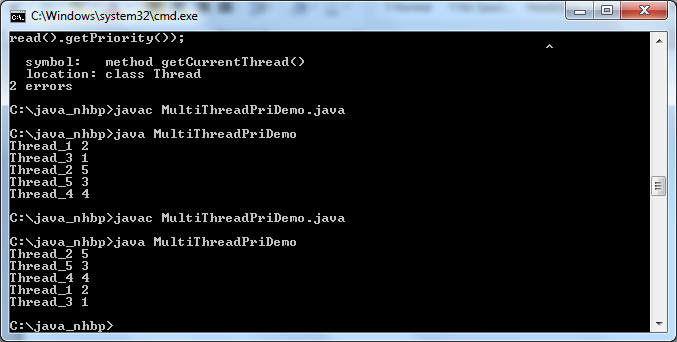
t3.start();

t4.start();

t5.start();

}

}



1. **Write a program to iterate through an ArrayList.**

import java.util.\*;

class IteratorDemo

{

public static void main(String s[])

{

ArrayList al=new ArrayList();

al.add(1);

al.add(22);

al.add(13);

al.add(46);

al.add(75);

Iterator itr=al.iterator();

while(itr.hasNext())

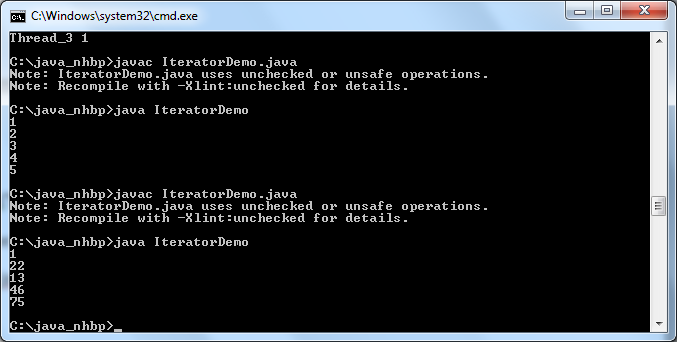
{

System.out.println(itr.next());

}

}

}



1. **Write a program to sort Employee objects based on highest salary using Comparator**

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

public class ComparatorDemo {

public static void main(String a[]){

List<Empl> list = new ArrayList<Empl>();

list.add(new Empl("Abirami",4000));

list.add(new Empl("Rahul",7000));

list.add(new Empl("Jeya",1000));

list.add(new Empl("Chandhu",2400));

list.add(new Empl("Monika",500));

list.add(new Empl("Ram",10000));

list.add(new Empl("Priya",102800));

Collections.sort(list,new MySalaryComp());

System.out.println("Sorted list entries: ");

for(Empl e:list){

System.out.println(e);

}

}

}

class MySalaryComp implements Comparator<Empl>{

@Override

public int compare(Empl e1, Empl e2) {

if(e1.getSalary() < e2.getSalary()){

return 1; } else {

return -1;

}

}

}

class Empl{

private String name;

private int salary;

public Empl(String n, int s){

this.name = n;

this.salary = s;

}

public String getName() {

return name;

}

public void setName(String name)

{this.name = name;

}

public int getSalary() {

return salary;

}

public void setSalary(int salary) {

this.salary = salary;

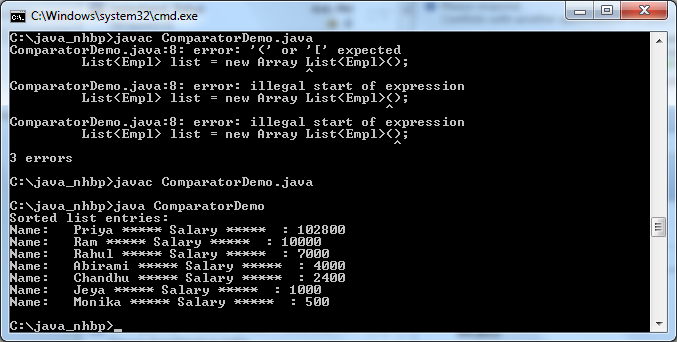
}

public String toString(){

return "Name: "+this.name +" \*\*\*\*\* Salary \*\*\*\*\* : "+this.salary;

}

}



**3. Write a program to eliminate duplicate keys (user defined objects) with Hashtable?**

**Hint: Use user defined objects (Employee) as keys. To avoid duplicates keys you have to implement equals and hashcode methods. (To successfully store and retrieve objects from a hashtable, the objects used as keys must implement the hashCode method and the equals method.).**