**Name:**

**Roll No:**

**Batch:**

**Date:**

**OBJECT ORIENTED PROGRAMMING LAB**

**Experiment No.: 1**

**Aim**

Define a class product with data members p\_id, p\_name and price. Create their object using the class and find the product having lowest price.

**Procedure**

**Source Code**

class product{

    int p\_id;

    String p\_name;

    double price;

    void details(){

        System.out.println("product id is "+p\_id);

        System.out.println("product name is "+p\_name);

        System.out.println("product price "+price);

    }

}

public class product\_details{

    public static void main(String[] args) {

        product p1=new product();

        p1.p\_id=123;

        p1.p\_name="milk";

        p1.price=120;

        p1.details();

        product p2=new product();

        p2.p\_id=234;

        p2.p\_name="bread";

        p2.price=90;

        p2.details();

        product p3=new product();

        p3.p\_id=233;

        p3.p\_name="jam";

        p3.price=80;

        p3.details();

        if(p1.price<p2.price && p1.price<p3.price){

            System.out.println("product 1 is lesser cost");

        }

        else if(p2.price<p3.price){

            System.out.println("product 2 is lesser cost");

        }

        else{

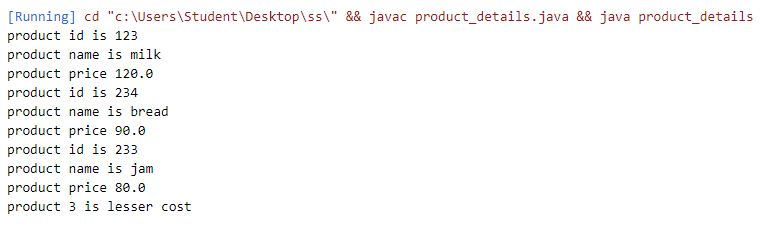
            System.out.println("product 3 is lesser cost");

        }

    }

}

**Output Screenshot**

****

**Name:**

**Roll No:**

**Batch:**

**Date:**

**OBJECT ORIENTED PROGRAMMING LAB**

**Experiment No.: 2**

**Aim:**

Read 2 matrices from the console and perform matrix addition.

**Procedure:**

**Source Code**

import **java**.**util**.**Scanner**;

public class **matrixAddition** {

    public void **Display**(int [][] arr,int row,int col){

        for(int i=0;i<row;i++){

            for(int j=0;j<col;j++){

**System**.out.**print**(arr[i][j]+"\t");}

**System**.out.**println**();}}

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

        int[][] mat1=new int[5][5];

        int[][] mat2=new int[5][5];

        int[][] mat3=new int[5][5];

        int rows1, cols1, rows2, cols2;

**matrixAddition** obj=new **matrixAddition**();

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

**System**.out.**println**("Enter the number of rows and columns of matrix 1");

        rows1=s.**nextInt**();

        cols1=s.**nextInt**();

**System**.out.**println**("Enter the elements of matrix 1");

        for(int i=0;i<rows1;i++){

            for(int j=0;j<cols1;j++){

                mat1[i][j]=s.**nextInt**();}}

**System**.out.**println**("Enter the number of rows and columns of matrix 2");

        rows2=s.**nextInt**();

        cols2=s.**nextInt**();

**System**.out.**println**("Enter the elements of matrix 2");

        for(int i=0;i<rows2;i++){

            for(int j=0;j<cols2;j++){

                mat2[i][j]=s.**nextInt**();}}

        if(rows1==rows2 && cols1==cols2){

            for(int i=0;i<rows1;i++){

                for(int j=0;j<cols1;j++){

                    mat3[i][j]=mat1[i][j]+mat2[i][j];}}

**System**.out.**println**("1st matrix");

            obj.**Display**(mat1,rows1,cols1);

**System**.out.**println**("2nd matrix");

            obj.**Display**(mat2,rows2,cols2);

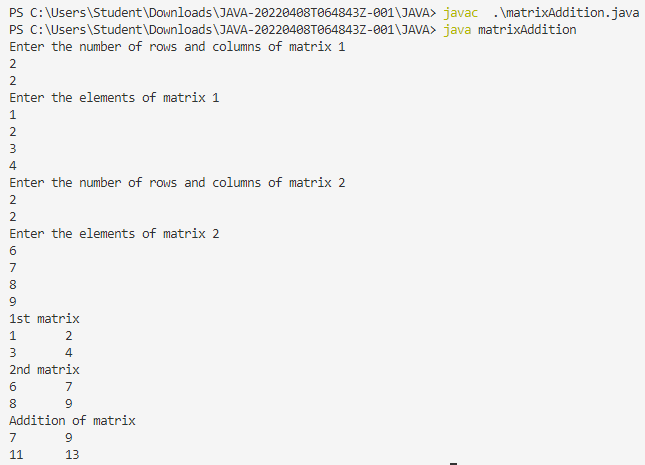
**System**.out.**println**("Addition of matrix");

            obj.**Display**(mat3,rows1,cols1);}

        else{

**System**.out.**println**("The matrices cannot be added");}}}

**Output Screenshort:**

****

**Name:**

**Roll No:**

**Batch:**

**Date:**

**OBJECT ORIENTED PROGRAMMING LAB**

**Experiment No.: 3**

**Aim:**

Add complex numbers

**Procedure:**

**Source Code**

import **java**.**util**.**Scanner**;

class **complex**{

    int r,b;

    void **display**(){

**System**.out.**println**("Complex Number is "+r+"+"+b+"i");}}

public class **addComplex** {

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

**complex** c1=new **complex**();

**complex** c2=new **complex**();

**complex** c3=new **complex**();

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

**System**.out.**println**("Enter the real and imaginary parts of complex number 1");

        c1.r=s.**nextInt**();

        c1.b=s.**nextInt**();

**System**.out.**println**("Enter the real and imaginary parts of complex number 2");

        c2.r=s.**nextInt**();

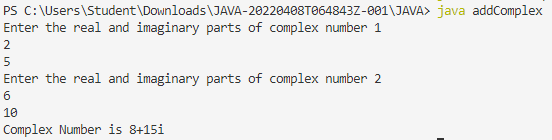
        c2.b=s.**nextInt**();

        c3.r=c1.r+c2.r;

        c3.b=c1.b+c2.b;

        c3.**display**();}}

**Output Screenshort:**

****

**Name:**

**Roll No:**

**Batch:**

**Date:**

**OBJECT ORIENTED PROGRAMMING LAB**

**Experiment No.: 4**

**Aim:**

Read a matrix from the console and check whether it is symmetric or not.

**Procedure:**

**Source Code**

import **java**.**util**.**Scanner**;

public class **symmetricMatrix** {

    public void **Display**(int [][] arr,int row,int col){

        for(int i=0;i<row;i++){

            for(int j=0;j<col;j++){

**System**.out.**print**(arr[i][j]+"\t");

            }

**System**.out.**println**();}}

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

        int [][] mat = new int[3][3];

        int [][] trans=new int[3][3];

        int row,col;

**symmetricMatrix** obj=new **symmetricMatrix**();

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

**System**.out.**println**("Enter the rows and columns of the matrix");

        row=s.**nextInt**();

        col=s.**nextInt**();

**System**.out.**println**("Enter the elements of the matrix");

        for(int i=0;i<row;i++){

            for(int j=0;j<col;j++){

                mat[i][j]=s.**nextInt**();}}

        for(int i=0;i<row;i++){

            for(int j=0;j<col;j++){

                trans[j][i]=mat[i][j];}

}

**System**.out.**println**("Entered matrix");

        obj.**Display**(mat,row,col);

**System**.out.**println**("Transpose of the matrix");

        obj.**Display**(trans,row,col);

        for(int i=0;i<row;i++){

            for(int j=0;j<col;j++){

                if(mat[i][j]!=trans[i][j]){

**System**.out.**println**("Matrix is not symmetric");

**System**.**exit**(0);}

}

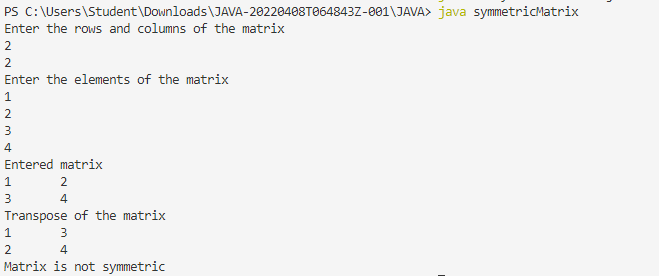
}

**System**.out.**println**("The given matrix is symmetric");

}

}

**Output Screenshort:**

****