OBJECT ORIENTED PROGRAMMING LAB

CYCLE:1

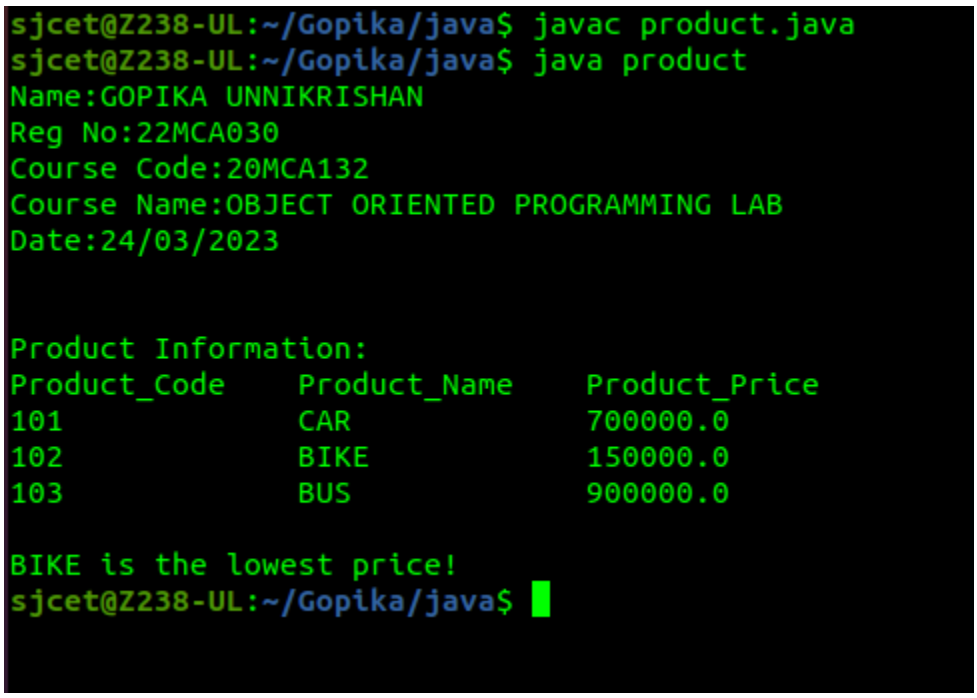
1)Define a class ‘product’ with data members pcode, pname and price. Create 3 objects of the class and find the product having the lowest price.

CODE:

```
public class product
{
	int pcode;
	String pname;
	double price;
	double lowest;
	void data(int c, String n, double p){
	    pcode=c;
	    pname=n;
	    price=p;
	}
	void display(){
	    System.out.println(pcode+"\t\t"+pname+"\t\t"+price);

	}
	static void findLowest(double price1,double price2, double price3){
	if(price1<=price2 && price1<=price3){
	    System.out.println("\nCAR isthe lowest price!");

	}
	else if(price2<=price1 && price2<=price3){
	    System.out.println("\nBIKE is the lowest price!");

	}
	else{
	    System.out.println("\n BUS is the lowest price!");

	}

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

```
        System.out.println("Name:GOPIKA UNNIKRISHAN\nReg No:22MCA030\nCourse Code:20MCA132\nCourse Name:OBJECT ORIENTED PROGRAMMING LAB\nDate:24/03/2023\n\n");
          product obj1 = new product();
          product obj2 = new product();
          product obj3 = new product();
          obj1.data(101,"CAR",700000.00);
          obj2.data(102,"BIKE",150000.00);
          obj3.data(103,"BUS",900000.00);
          System.out.println("Product Information:\nProduct_Code\tProduct_Name\tProduct_Price");
          obj1.display();
          obj2.display();
          obj3.display();
          findLowest(obj1.price,obj2.price,obj3.price);

      }
   }
```

OUTPUT



2)Read 2 matrices from the console and perform matrix addition.

CODE:

```
import java.util.*;
public class Matrix
{
public static void main(String[] args)
{
System.out.println("Name:GOPIKA UNNIKRISHAN\nReg No:22MCA030\nCourse
Code:20MCA132\nCourse Name:OBJECT ORIENTED PROGRAMMING
LAB\nDate:24/03/2023\n\n");
int r,c;
Scanner x = new Scanner (System.in);
System.out.println("Number of rows");
r=x.nextInt();
System.out.println("Number of coloumn");
c=x.nextInt();
  int m1[][]=new int[r][c];
  int m2[][]=new int[r][c];
  int m3[][] = new int[r][c];
           System.out.println("Enter all the elements of first matrix:");
          for (int i = 0; i < r; i++)
          {
              for (int j = 0; j < c; j++)
              {
                  m1[i][j] = x.nextInt();
              }
          }
          System.out.println("");
          System.out.println("Enter all the elements of second matrix:");
          for (int i = 0; i < r; i++)
          {
              for (int j = 0; j < c; j++)
              {
                  m2[i][j] = x.nextInt();
              }
          }
          System.out.println("");
          System.out.println("First Matrix:");
          for (int i = 0; i < r; i++)
```

```
        {
            for (int j = 0; j < c; j++)
            {
                System.out.print(m1[i][j]+" ");
            }
            System.out.println("");
        }
        System.out.println("Second Matrix:");
        for (int i = 0; i < r; i++)
        {
            for (int j = 0; j < c; j++)
            {
                System.out.print(m2[i][j]+" ");
            }
            System.out.println("");
        }
        for (int i = 0; i < r; i++)
        {
            for (int j = 0; j < c; j++)
            {
                for (int k = 0; k < c; k++)
                {
                    m3[i][j] = m1[i][j] + m2[i][j];
                }
            }
        }
        System.out.println("Matrix after addition:");
        for (int i = 0; i < r; i++)
        {
            for (int j = 0; j < c; j++)
            {
                System.out.print(m3[i][j]+" ");
            }
            System.out.println("");
        }
    }

}
```

OUTPUT

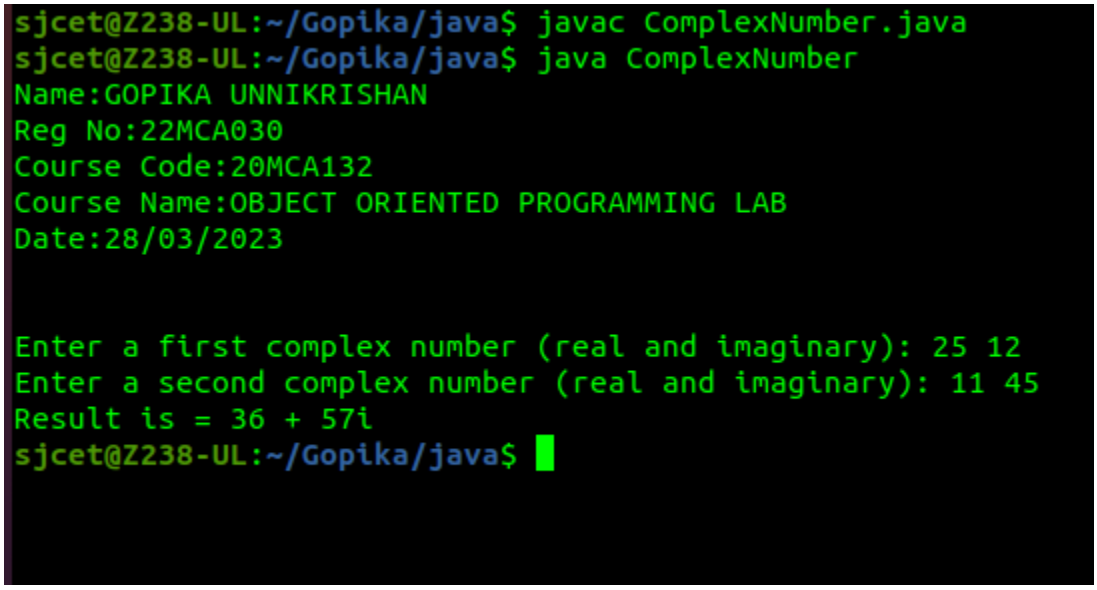
```
sjcet@Z238-UL:~/Gopika/java$ javac Matrix.java
sjcet@Z238-UL:~/Gopika/java$ java Matrix
Name:GOPIKA UNNIKRISHAN
Reg No:22MCA030
Course Code:20MCA132
Course Name:OBJECT ORIENTED PROGRAMMING LAB
Date:24/03/2023


Number of rows
2
Number of coloumn
3
Enter all the elements of first matrix:
1 2 3
4 5 6

Enter all the elements of second matrix:
7 8 9
4 5 6

First Matrix:
1 2 3
4 5 6
Second Matrix:
7 8 9
4 5 6
Matrix after addition:
8 10 12
8 10 12
sjcet@Z238-UL:~/Gopika/java$
```

3)Add complex numbers

CODE:

```
import java.util.Scanner;

class Complex
{
	int real;
	int img;
}
public class ComplexNumber
{
	public static void main(String[] args)
	{
		System.out.println("Name:GOPIKA UNNIKRISHAN\nReg
No:22MCA030\nCourse Code:20MCA132\nCourse Name:OBJECT ORIENTED
PROGRAMMING LAB\nDate:28/03/2023\n\n");
		Scanner SC = new Scanner(System.in);
		Complex num1 = new Complex();
		Complex num2 = new Complex();
		Complex num3 = new Complex();
		System.out.printf("Enter a first complex number (real and imaginary): ");
		num1.real = SC.nextInt();
		num1.img = SC.nextInt();
		System.out.printf("Enter a second complex number (real and imaginary):
");
		num2.real = SC.nextInt();
		num2.img = SC.nextInt();
		num3.real = num1.real + num2.real;
		num3.img = num1.img + num2.img;
		if(num3.img >= 0)
		{
			System.out.printf("Result is = %d + %di\n", num3.real, num3.img);
		}
		else
		{
			System.out.printf("Result is = %d %di\n", num3.real, num3.img);
		}

	}
}
```

OUTPUT



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

CODE:

```
import java.util.*;
public class Symmetric
{
	public static void main(String[] args)
	{
		System.out.println("Name:GOPIKA UNNIKRISHAN\nReg No:22MCA030\nCourse Code:20MCA132\nCourse Name:OBJECT ORIENTED PROGRAMMING LAB\nDate:28/03/2023\n\n");
		Scanner s=new Scanner(System.in);
		int r,c;
		System.out.println("enter no of rows and columns");
		r=s.nextInt();
		c=s.nextInt();
		int[][]  a=new int[r][c];
		int[][]  b=new int[r][c];
		System.out.println("enter the matrix elements");
		for(int i=0;i<r;i++)
		{
			for(int j=0;j<c;j++)
			{
				a[i][j]=s.nextInt();
```

```
                }
            }
            if(r==c)
            {
                for(int i=0;i<r;i++)
                {
                    for(int j=0;j<c;j++)
                    {
                        b[i][j]=a[j][i];
                    }
                }
            }
        int x=0;
        for(int i=0;i<r;i++)
        {
            for(int j=0;j<c;j++)
            {
                if(a[i][j]==b[i][j])
            x=1;
        }
    }
        if(x==1)
        {
            System.out.println("\nMatrix is a Symmetric Matrix");
        }
        else
        {
            System.out.println("\nMatrix is  not a Symmetric Matrix");

        }
        }
}
```

OUTPUT

```
sjcet@Z238-UL:~/Gopika/java$ javac Symmetric.java
sjcet@Z238-UL:~/Gopika/java$ java Symmetric
Name:GOPIKA UNNIKRISHAN
Reg No:22MCA030
Course Code:20MCA132
Course Name:OBJECT ORIENTED PROGRAMMING LAB
Date:28/03/2023


enter no of rows and columns
3
2
enter the matrix elements
1 2
4 5
7 8

Matrix is  not a Symmetric Matrix
sjcet@Z238-UL:~/Gopika/java$ javac Symmetric.java
sjcet@Z238-UL:~/Gopika/java$ java Symmetric
Name:GOPIKA UNNIKRISHAN
Reg No:22MCA030
Course Code:20MCA132
Course Name:OBJECT ORIENTED PROGRAMMING LAB
Date:28/03/2023


enter no of rows and columns
3
3
enter the matrix elements
1 2 3
2 4 5
3 5 6

Matrix is a Symmetric Matrix
sjcet@Z238-UL:~/Gopika/java$
```

5)Create CPU with attribute price. Create inner class Processor (no. of cores, manufacturer) and static nested class RAM (memory, manufacturer). Create an object of CPU and print information of Processor and RAM.

CODE:

```
class CPU
{
	double price;
	class Processor
	{
	double cores;
	String manufacturer;

	double getCache()
	{
			return 4.3;
	}
	}
	protected class RAM
	{
	double memory;
	String manufacturer;

	double getClockSpeed()
	{
			return 5.5;
	}
      }
}

class main
{
	public static void main(String[] args)
	{
		System.out.println("Name:GOPIKA UNNIKRISHAN\nReg No:22MCA030\nCourse Code:20MCA132\nCourse Name:OBJECT ORIENTED PROGRAMMING LAB\nDate:04/0/2023\n\n");
	CPU cpu = new CPU();
	CPU.Processor processor = cpu.new Processor();
	CPU.RAM ram = cpu.new RAM();
```

```
        System.out.println("Processor Cache = " + processor.getCache());
        System.out.println("Ram Clock speed = " + ram.getClockSpeed());
        }
}
```

OUTPUT