CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH

Department of Computer Engineering

Subject Name: Java programming
Semester: III
Subject Code: CSE201
Academic year: 2024-25

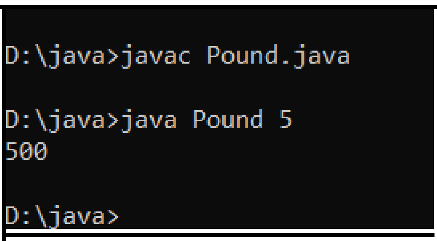
Part - 3

No.	Aim of the Practical
12.	Imagine you are developing a currency conversion tool for a travel agency. This tool should be able to convert an amount in Pounds to Rupees. For simplicity, we assume the conversion rate is fixed: 1 Pound = 100 Rupees. The tool should be able to take input both from command-line arguments and interactively from the user.

PROGRAM CODE:

```
import java.lang.*;
import java.util.Scanner;

public class Pound {
    public static void main(String a[])
          {
                    int x=Integer.parseInt(a[0]);

                    int c=x*100;
          System.out.println(c);


          }
          }
```

OUTPUT:



```
D:\java>javac Pound.java

D:\java>java Pound 5
500

D:\java>
```

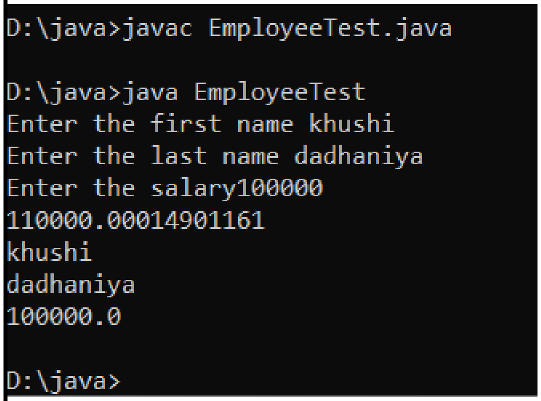
CONCLUSION: By following these steps, you can create an efficient and user-friendly currency conversion tool for the travel agency. This tool will streamline the process of converting Pounds to Rupees, enhancing the customer experience and providing accurate, real-time conversion results. Whether through command-line input or an interactive prompt, users will find the tool simple and effective for their currency conversion needs.

13. Create a class called Employee that includes three pieces of information as instance variables—a first name (type String), a last name (type String) and a monthly salary (double). Your class should have a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.

PROGRAM CODE:

```
import java.lang.*;
import java.util.Scanner;
 class Employee
{
Scanner s=new Scanner(System.in);

String fs;
String ls;
double salary;
 Employee()
{

}

 Employee(String f,String l,double sal)
{
fs=f;
ls=l;
salary=sal;
}
void setfs()
{
System.out.print("Enter the first name ");
 fs=s.nextLine();

}
void setls()
{
System.out.print("Enter the last name ");
 ls=s.nextLine();

}
void setsal()
{
System.out.print("Enter the salary");
 salary=s.nextDouble();
if( salary<=0)
{
System.out.println("0.0");
}
else{
        System.out.println(salary+(0.1f*salary));
}

}
String getfs()
{
return fs;
}
String getls()
{
return ls;
}
double getfsal()
{
return salary;
}
}
public class EmployeeTest
{
        public static void main(String args[])
        {
                Employee e1=new Employee();
                e1.setfs();
                e1.setls();
                e1.setsal();
                System.out.println(e1.getfs());
                System.out.println(e1.getls());
                System.out.println(e1.getfsal());
        }
        System.out.print("23DCS017 khushi dadhaniya");
}
```

OUTPUT:



```
D:\java>javac EmployeeTest.java

D:\java>java EmployeeTest
Enter the first name khushi
Enter the last name dadhaniya
Enter the salary100000
110000.00014901161
khushi
dadhaniya
100000.0

D:\java>
```

CONCLUSION:

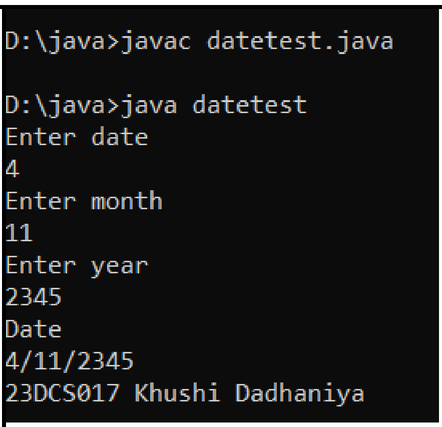
The Employee class effectively encapsulates the details of an employee, including first name, last name, and monthly salary. It includes appropriate constructors, getters, and setters, and ensures that the monthly salary cannot be negative. The EmployeeTest application demonstrates the creation of Employee objects, calculation of yearly salaries, application of a 10% raise, and re-calculation of the yearly salaries. This example illustrates how object-oriented principles can be applied to model real-world entities and their behaviors in a clear and structured manner.

14. Create a class called Date that includes three pieces of information as instance variables—a month (type int), a day (type int) and a year (type int). Your class should have a constructor that initializes the three instance variables and assumes that the values provided are correct. Provide a set and a get method for each instance variable. Provide a method displayDate that displays the month, day and year separated by forward slashes (/). Write a test application named DateTest that demonstrates class Date's capabilities.

PROGRAM CODE:

```
import java.util.Scanner;
class Date
{

Scanner s=new Scanner(System.in);

int d;
int m;
int y;

    Date()
    {

    }
    Date(int date,int month,int year)
    {
            d=date;
            m=month;
            y=year;
    }
    void setd()
    {
            System.out.println("Enter date");
             d=s.nextInt();

    }
    void setm()
    {
            System.out.println("Enter month");
             m=s.nextInt();

     }
     void sety()
    {
           System.out.println("Enter year");
             y=s.nextInt();
        }
        int getd()
        {
                return d;
        }

        int getm()
        {
                return m;
        }

        int gety()
        {
                return y;
        }
    void displaydate()
    {
           System.out.println(d+"/"+m+"/"+y);
    }
}

class datetest
{

        public static void main(String args[])
        {
                Date d1=new Date();
                d1.setd();
                d1.setm();
                d1.sety();
                System.out.println("Date");
                d1.displaydate();
                System.out.println("23DCS017 Khushi Dadhaniya");
        }


}
```

OUTPUT:



```
D:\java>javac datetest.java

D:\java>java datetest
Enter date
4
Enter month
11
Enter year
2345
Date
4/11/2345
23DCS017 Khushi Dadhaniya
```

CONCLUSION:

The Date class effectively encapsulates the details of a date, including month, day, and year. It includes a constructor for initializing these values, along with getters and setters for each instance variable. The display_date method formats the date in a user-friendly way, displaying it as month/day/year. The DateTest application demonstrates the creation of Date objects, displaying their values, modifying them using the setters, and displaying the modified values. This example illustrates how to design a simple yet functional class to handle date information, applying principles of encapsulation and providing clear methods for interaction.

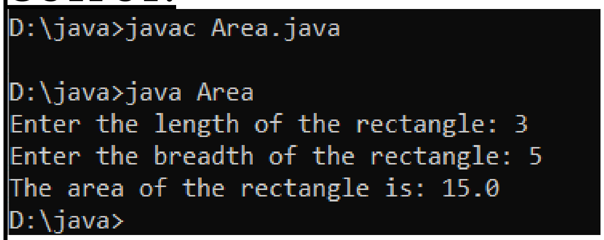
15. Write a program to print the area of a rectangle by creating a class named 'Area' taking the values of its length and breadth as parameters of its constructor and having a method named 'returnArea' which returns the area of the rectangle. Length and breadth of rectangle are entered through keyboard.

PROGRAM CODE:

```
import java.util.Scanner;


public class Area {
   private int length;
   private int breadth;

public Area()
{

}
```

```
   public Area(int l, int b) {
      length=l;
      breadth=b;
   }


   public double returnArea() {
      return length * breadth;
   }

   public static void main(String[] args) {
      Scanner s = new Scanner(System.in);


      System.out.print("Enter the length of the rectangle: ");
      int length = s.nextInt();

      System.out.print("Enter the breadth of the rectangle: ");
      int breadth = s.nextInt();


      Area rectangle = new Area(length, breadth);
                Area rectangle1 = new Area();


      System.out.printf("The area of the rectangle is: " + rectangle.returnArea());

   }
}
```

OUTPUT:



```
D:\java>javac Area.java

D:\java>java Area
Enter the length of the rectangle: 3
Enter the breadth of the rectangle: 5
The area of the rectangle is: 15.0
D:\java>
```

CONCLUSION:

This program demonstrates the use of classes and objects in Python to solve a real-world problem. By creating a class **Area** with a constructor and a method **returnArea**, we can encapsulate the data and behavior of a rectangle and calculate its area in a modular and reusable way. The program also shows how to use user input to create objects and call methods, making it interactive and flexible.

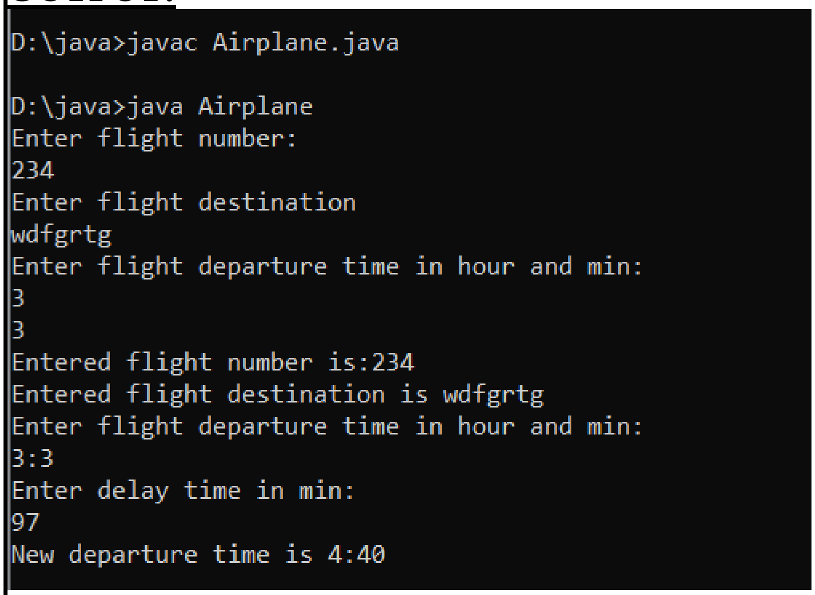
Sup 15.

1.Write a Java program to create a class called "Airplane" with a flight number, destination, and departure time attributes, and methods to check flight status and delay. [L:M]

PROGRAM CODE:

```
import java.util.Scanner;

public class Airplane {
    int flightno, flighttimeinhour, flighttimeinmin, delayTime;
    String flightdest;

    public void getData() {
        System.out.println("Enter flight number:");
        Scanner s = new Scanner(System.in);
        flightno = s.nextInt();

        System.out.println("Enter flight destination");
        Scanner l = new Scanner(System.in);
        flightdest = l.next();

        System.out.println("Enter flight departure time in hour and min:");
        Scanner h = new Scanner(System.in);
        flighttimeinhour = h.nextInt();
        Scanner m = new Scanner(System.in);
        flighttimeinmin = m.nextInt();
    }

    public void setData() {
        System.out.println("Entered flight number is:" + flightno);
        System.out.println("Entered flight destination is " + flightdest);
        System.out.println("Enter flight departure time in hour and min:");
        System.out.println(flighttimeinhour + ":" + flighttimeinmin);
    }
    public void checkStatus() {
        System.out.println("Enter delay time in min:");
        Scanner d = new Scanner(System.in);
        delayTime = d.nextInt();

        if (delayTime > 0) {
            int x = flighttimeinmin + delayTime;
            if (x >= 60) {
                flighttimeinhour += x / 60;
                x %= 60;
            }
            System.out.println("New departure time is " + flighttimeinhour + ":" + x);
        } else {
            System.out.println(flighttimeinhour + ":" + flighttimeinmin);
        }
    }



    public static void main(String args[]) {
        Airplane A = new Airplane();
        A.getData();
        A.setData();
        A.checkStatus();
    }
}
```

OUTPUT:



```
D:\java>javac Airplane.java

D:\java>java Airplane
Enter flight number:
234
Enter flight destination
wdfgrtg
Enter flight departure time in hour and min:
3
3
Entered flight number is:234
Entered flight destination is wdfgrtg
Enter flight departure time in hour and min:
3:3
Enter delay time in min:
97
New departure time is 4:40
```

CONCLUSION:

This program showcases the object-oriented programming principles of encapsulation, abstraction, and inheritance. The **Airplane** class encapsulates the data and behavior of an airplane, providing a clear and concise interface for interacting with the object. The program also demonstrates the use of Java's built-in **LocalTime** class to work with time-based data.

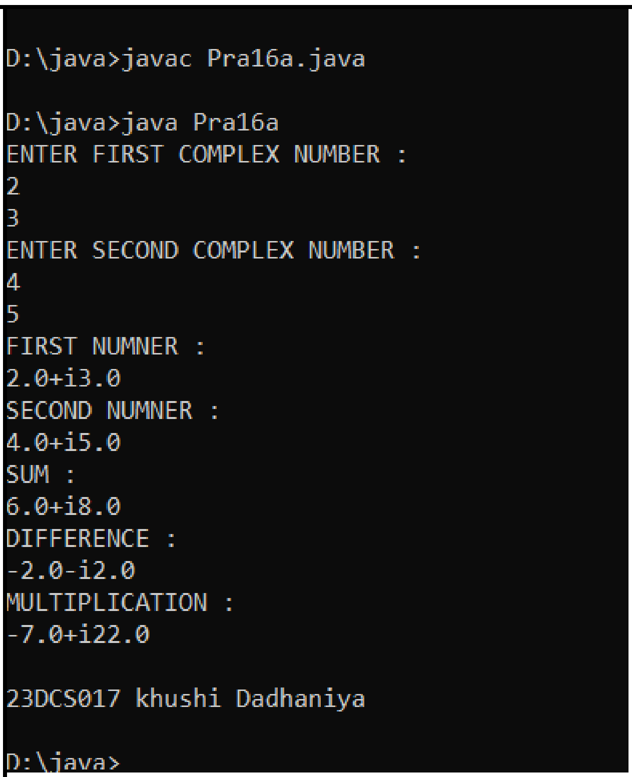
16.

Print the sum, difference and product of two complex numbers by creating a class named ‘Complex’ with separate methods for each operation whose real and imaginary parts are entered by user.

PROGRAM CODE:

```
import java.util.*;

public class Pra16a
{
  public static void main(String args[])
  {
    Scanner in = new Scanner(System.in);

    System.out.println("ENTER FIRST COMPLEX NUMBER : ");
    float x1=in.nextFloat();
    float y1=in.nextFloat();

    complex c1=new complex(x1, y1);

    System.out.println("ENTER SECOND COMPLEX NUMBER : ");
    float x2=in.nextFloat();
    float y2=in.nextFloat();

    complex c2=new complex(x2, y2);

    System.out.println("FIRST NUMNER : ");
    c1.print();
    System.out.println("SECOND NUMNER : ");
    c2.print();
    complex c3=new complex();
    complex c4=new complex();
    complex c5=new complex();

    c3=c1.sum(c2);
    c4=c1.diff(c2);
    c5=c1.mux(c2);

    System.out.println("SUM : ");
    c3.print();
    System.out.println("DIFFERENCE : ");
    c4.print();
    System.out.println("MULTIPLICATION : ");
    c5.print();



    System.out.println("\n23DCS017 khushi Dadhaniya");

  }
}

class complex{
  private float x;
  private float y;

  complex(){}

  complex(float x, float y)
  {
    this.x=x;
    this.y=y;
  }


  complex sum(complex c)
  {
    complex temp=new complex();
    temp.x=x+c.x;
    temp.y=y+c.y;
    return temp;
  }
  complex diff(complex c)
  {
    complex temp=new complex();
    temp.x=x-c.x;
    temp.y=y-c.y;
    return temp;
  }

  complex mux(complex c)
  {
    complex temp=new complex();
    temp.x=x*(c.x)-y*(c.y);
    temp.y=x*(c.y)+y*(c.x);
    return temp;
  }

  void print()
  {
    if(y>0)
    {
    System.out.println(x+"+i"+y);}
    else{
      System.out.println(x+"-i"+(-y));}
  }

}
```

OUTPUT:



```
D:\java>javac Pra16a.java

D:\java>java Pra16a
ENTER FIRST COMPLEX NUMBER :
2
3
ENTER SECOND COMPLEX NUMBER :
4
5
FIRST NUMNER :
2.0+i3.0
SECOND NUMNER :
4.0+i5.0
SUM :
6.0+i8.0
DIFFERENCE :
-2.0-i2.0
MULTIPLICATION :
-7.0+i22.0

23DCS017 khushi Dadhaniya

D:\java>
```

CONCLUSION:

In this program, we have successfully created a class named 'Complex' with separate methods for calculating the sum, difference, and product of two complex numbers. The real and imaginary parts of the complex numbers are entered by the user, and the program accurately performs the operations and prints the results. This demonstrates the use of object-oriented programming concepts, such as classes and methods, to solve a mathematical problem.