VARDHAMAN COLLEGE OF ENGINEERING
(AUTONOMOUS)

Department of Information Technology

LAB MANUAL

OBJECT OREINTED PROGRAMMING

II Yr I SEM SEM B.TECH CSE/IT

Course Title	**Object Oriented Programming**
Course Type	**Integrated**
Category	**Core Engineering**
Regulation	**VCE-R19**
Academic Year	**2019-2020**
Course Code	**A5601**

	CSE	IT
Course Instructors	A. Mr. A. Ramesh (Section-B) B. Dr. D. Raman (Section-A) C. Mr. Para Upendar (Section-D) D. V. Vijaya Bhaskar Reddy (Section-C)	E. Mrs. E Aruna (Section-A) (Course Lead) F. Mr. S Nagarjuna Reddy (Section- B)

COURSE OVERVIEW

This integrated course provides a comprehensive coverage of theory and practice of OOP concepts using Java. The course focuses on different aspect of core Java Environment suitable to write efficient, maintainable, and portable code. It also ignites Object Oriented thinking and explores with the evolution of Java and its basics. It provides strong foundation on Inheritance, Packages, and Interfaces and also illustrates Exception Handling and Multithreaded mechanisms. In depth knowledge to implement Collection frameworks and Event handling is discussed. Emphasis on Swing concepts used for GUI applications is given. The course plays a vital role in develop front-end interface for Mini and Major Projects.

COURSE OBJECTIVE

The course enables the learner to apply Object-oriented Programming concepts to develop Console and GUI components as per needs and specifications.

COURSE OUTCOMES (COs)

After the completion of the course, the student will be able to:

CO#	Course Outcomes	POs	PSOs
A45051	Understand the principles of object oriented programming.	-	-
A45052	Design user defined packages and interfaces.	1,5	1,2
A45053	Analyze the applications for handling exceptions and multithreading.	2,5	1,2
A45054	Implement Collection Frameworks to retrieve data efficiently and to handle events.	1,5	1,2
A45055	Build GUI applications using swings.	3,5	1,2

BLOOM'S LEVEL OF THE COURSE OUTCOMES

CO#	Bloom's Level					
	Remember (L1)	Understand (L2)	Apply (L3)	Analyze (L4)	Evaluate (L5)	Create (L6)
A45051		✓				
A45052			✓			
A45053				✓		
A45054			✓			
A45055			✓			

COURSE ARTICULATION MATRIX

CO#/ POs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
A45052	3				3								2	2
A45053		2			3								2	2
A45054	3				3								2	2
A45055			3		3								3	3

Note: 1-Low, 2-Medium, 3-High

COURSE ASSESSMENT

S No	Component		Duration in Hours	Component Wise Marks	Total Marks	Weightage	Marks
1	Continuous Internal Evaluation (CIE)	Theory: Test-1	1	20	100	0.3	30
2		Theory: Test-2	1	20			
3		Alternate Assessment*	-	20			
4		**Practical Exam**	**2**	**40**			
5	Semester End Exam (SEE)		3	100	100	0.7	70
						Total Marks	**100**

LIST OF PROGRAMS FOR PRACTICE.

No	Title of the Experiment	Tools and Techniques	Expected Skills/Ability
1	**Implement Control statements and Arrays** a.**Read the marks of a student in 4 subjects and find grade.** **b.Program to check a number is Amstrong or not.** **c.Program to display prime numbers from m to n.**	Open source Java Tool kit: JDK 8 and above versions, Open Source IDE: Eclipse / NetBeans OS: Windows / Linux Web browser: Internet Explorer/ Google/ Firefox	Apply OOP principles to solve problems
2	**Implement OOP First principle - Encapsulation** **a. Define a class Rectangle with data member's length and width. Write methods to find perimeter and area of a rectangle. (class and object)** **b. Create a class Account with data members name, acno and balance. Use appropriate methods to perform various operations like deposit,withdraw, balnceCheck.** **c. Create a class Student with appropriate data and methods using constructor.**		
3	**Implement OOP Second principle – Polymorphism.** a. **Create overloaded methods to find volume of Sphere, Cylinder & Cone.** **b. To sort given list of elements in ascending order.** **c. Read two matrices of size m*n , p*q , perform the multiplication of matrices.**		
4	**Implement Java String Class.** **a. Check a string is palindrome or not.** **b. Given a string and an int n, return a string made of n repetitions of the last n characters of the string. You may assume that n is between 0 and the length of the string, inclusive. Write a Java program.** **repeatEnd("Hello",3)→"llollollo** **repeatEnd("Hello",2)→"lolo"** **repeatEnd("Hello", 1) → "o"** **c. We'll say that a "triple" in a string is a char appearing three times in a row. Return the number of triples in the given string. The triples may**		Develop user defined packages and Interfaces

No	Title of the Experiment	Tools and Techniques	Expected Skills/Ability
	overlap. Write a Java program. **countTriple("abcXXXabc")→1,** **countTriple("xxxabyyyycd")→3** **countTriple("a") → 0** **d. Read array of City names and Sort in dictionary order.(Ascending order).**		
5	**Implement OOP Third principle – Inheritance.** **a. Declare a class called Employee having employee _id and employee_name as members. Extend class Employee to have a subclass called Salary having designation and monthly_salary as members. Define following:** • **Required constructor** • **A method to find and display all details of employees drawing salary more than Rs.20000/-** • **main () method to create an array.** **b. Write a Java program that create an abstract base class Shape with two members base and height, a member function for initialization and a function to compute shapeArea(). Derive two specific classes Triangle and Rectangle which override the function shapeArea(). Write a driver classes (main) to display the area of the triangle and the rectangle.(Use super keyword).**		
6	**Implement Packages and Interfaces.** **a. Create a Package Measure; in which store a class named Convertor that contains methods to convert mm to cm, cm to m and m to km. Define a class Need_Convertor that imports the Convertor class, now store Need_Convertor outside the package Measure. Perform path settings accordingly.**		

No	Title of the Experiment	Tools and Techniques	Expected Skills/Ability
	b. Write a Java program that implements an interface Student which has two methods displayGrade() and attendance(). Implement two classes PG_Student and UG_Student with necessary inputs of data.		
7	**Implement Exception Handling .** **a. The user reads two numbers as strings, Num1 and Num2. The division of Num1 andNum2 is displayed in the Result. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 is Zero, the program would throw an ArithmeticException. Display the exception message.** **b. In the CustomExceptionTest class, the age is expected to be a positive number. It would throw the user defined exception NegativeAgeException if the age is assigned a negative number.**		Analyze exceptions and multithreading and handle them
8	**Develop applications on Multithreaded Programming and thread synchronization.** **a. Create a multithreaded java program by creating a subclass of Thread and then creating, initializing, and staring two Thread objects from your class. The threads will execute concurrently and display "Java is object oriented" in console window.** **c. Implement the concept of producer consumer problem using thread synchronization.**		
9	**Implement Collection Frameworks to retrieve data.** **a. Use an ArrayList to manage Employee objects for insertion, display and remove.** **b. Use HashSet methods to perform operations on collection of data.**		Develop Collection frameworks and handle events

No	Title of the Experiment	Tools and Techniques	Expected Skills/Ability
10	**Implement Mouse and Key events.** **a.ImplementMouseListener and MouseMotionListener to handle various mouse events.** **b. Implement KeyListener to handle key events.**		
11	**Develop GUI applications using AWT.** **a. Create a Simple login window to validate a user with name and password.** **b. Using Grid Layout design a Simple calculator with appropriate event handling.**		Develop GUI applications using AWT and Swings
12	**Develop GUI applications using Swing Controls.** a**. Create a user interface to insert employee details, Display the data in Text area.** **b. Create a JTable to display various fields of Student data like RollNo, Name, Branch ,Year, Percentage etc.**		

WEEK-1

After the completion of the practice session, the student will be able to Implement Control statements

Using Java Scanner Class to read input

Scanner is a class in "java.util" package used for obtaining the input of the primitive types like int, double, and strings etc. It is the easiest way to read input in a Java program

The Java Scanner class breaks the input into tokens using a delimiter that is whitespace by default. It provides many methods to read and parse various primitive values. Java Scanner class is widely used to parse text for string and primitive types using regular expression.

Commonly used methods of Scanner class

Method	Description
public String next()	It returns the next token (string) from the scanner.
public String nextLine()	It moves the scanner position to the next line and returns the value as a string.
public byte nextByte()	It scans the next token as a byte.
public short nextShort()	It scans the next token as a short value.
public int nextInt()	It scans the next token as an int value.
public long nextLong()	It scans the next token as a long value.
public float nextFloat()	It scans the next token as a float value.
public double nextDouble()	It scans the next token as a double value.

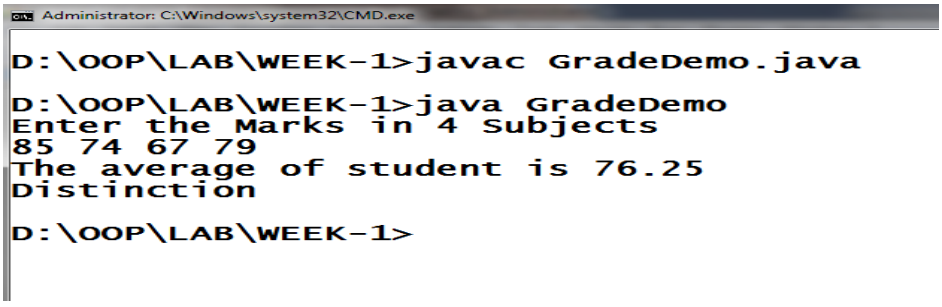
// Example to read and display data

```
import java.util.*;
import java.io.*;
public class DataRead
    {
        public static void main(String args[])
           {
              Scanner sc=new Scanner(System.in);

              System.out.println("Enter your rollno");
              int rollno=sc.nextInt();

              System.out.println("Enter your name");
              String name=sc.next();

              System.out.println("Enter your fee");
              double fee=sc.nextDouble();

                    System.out.println("Roll No  : " +rollno);
                    System.out.println("Name is " +name);
                    System.out.println("Feee is " +fee);

          //System.out.println("Rollno:"+rollno  +" name:" +name + " fee:" +fee);

              sc.close();

        }
  }
```

a) Program to find Grade of a Student in 4 Subjects

```
import java.util.*;
public class  GradeDemo
{
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter the Marks in 4 Subjects");
                int m1= sc.nextInt();
                int m2= sc.nextInt();
                double m3= sc.nextDouble();
                double m4= sc.nextDouble();

                double avg = (m1+m2+m3+m4)/4.0;

               System.out.println("The average of student is " +avg);

                if(avg >= 70)
                        System.out.println("Distinction");
                else if (avg >=60 && avg <70)
                        System.out.println("First Class");
                else if (avg >=50 && avg < 60)
                        System.out.println("Second Class");
                else if (avg >=40 && avg < 50)
                        System.out.println("Second Class");
                else
                        System.out.println("Fail");

        }
```

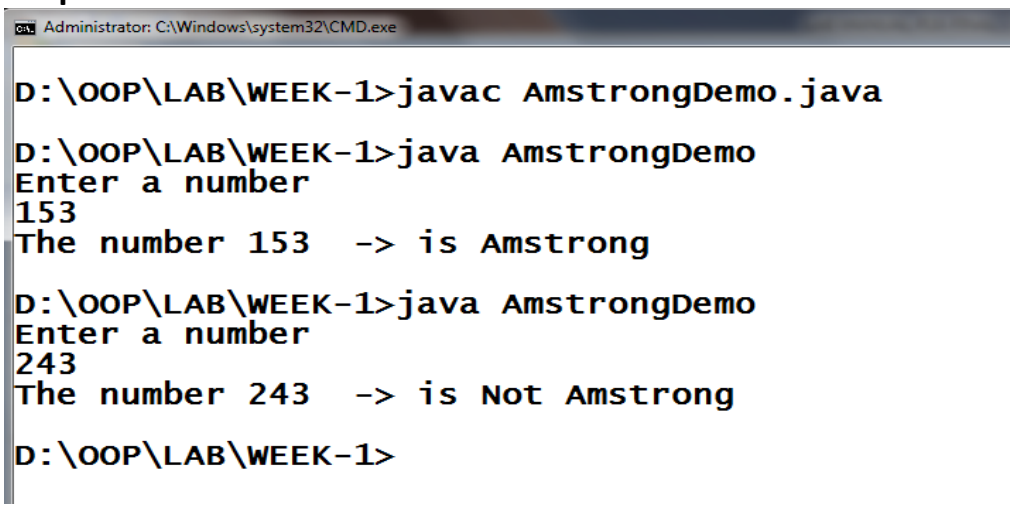
Output:



b) Program to check a Number is Amstrong or Not

```
import java.util.*;
class  AmstrongDemo
{
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter a number");
                int n = sc.nextInt();
                int s=0, r , m;
                m=n;
                while(n > 0)
                {
                        r = n % 10;
                        s = s  + (r*r*r);
                        n = n/10;
                }

                if (s==m)
                System.out.println("The number " +m + "  -> is Amstrong");
                else
                System.out.println("The number " +m + "  -> is Not Amstrong");

        }
}
```

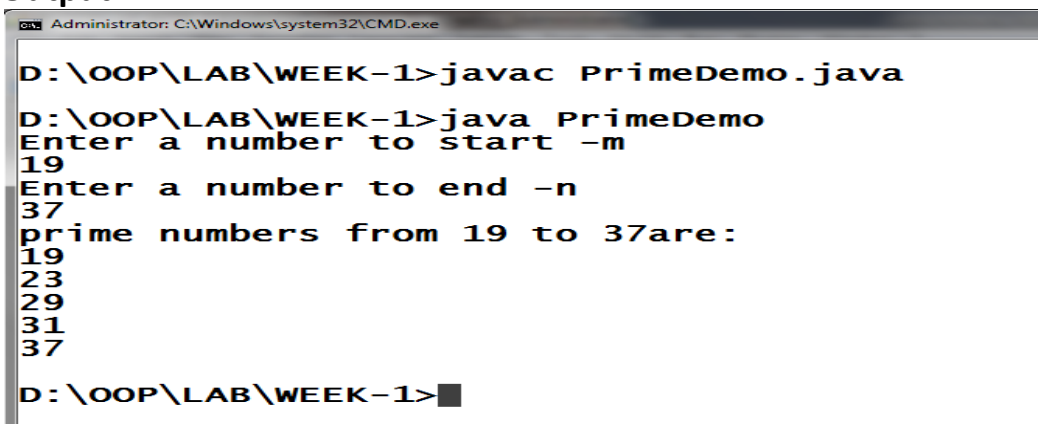
Output:



C) Program to display Prime Numbers from m to n

```
import java.util.*;
public class  PrimeDemo
{
        public static void main(String[] args)
        {
                        Scanner sc = new Scanner(System.in);
                    System.out.println("Enter a number to start -m ");
                    int m = sc.nextInt();
                        System.out.println("Enter a number to end -n ");
                    int n = sc.nextInt();
                         int i , num;
                System.out.println("prime numbers from " + m + " to "+n+"are:");
                        for (i = m; i <= n; i++)
                        {
                            int count=0;
                             for(num =1; num<=i; num++)
                              {
                                        if(i%num==0)
                                   count = count + 1;
                                }
                             if (count ==2)
                                  System.out.println(i);
                        }
            }
```

Output:



WEEK-2

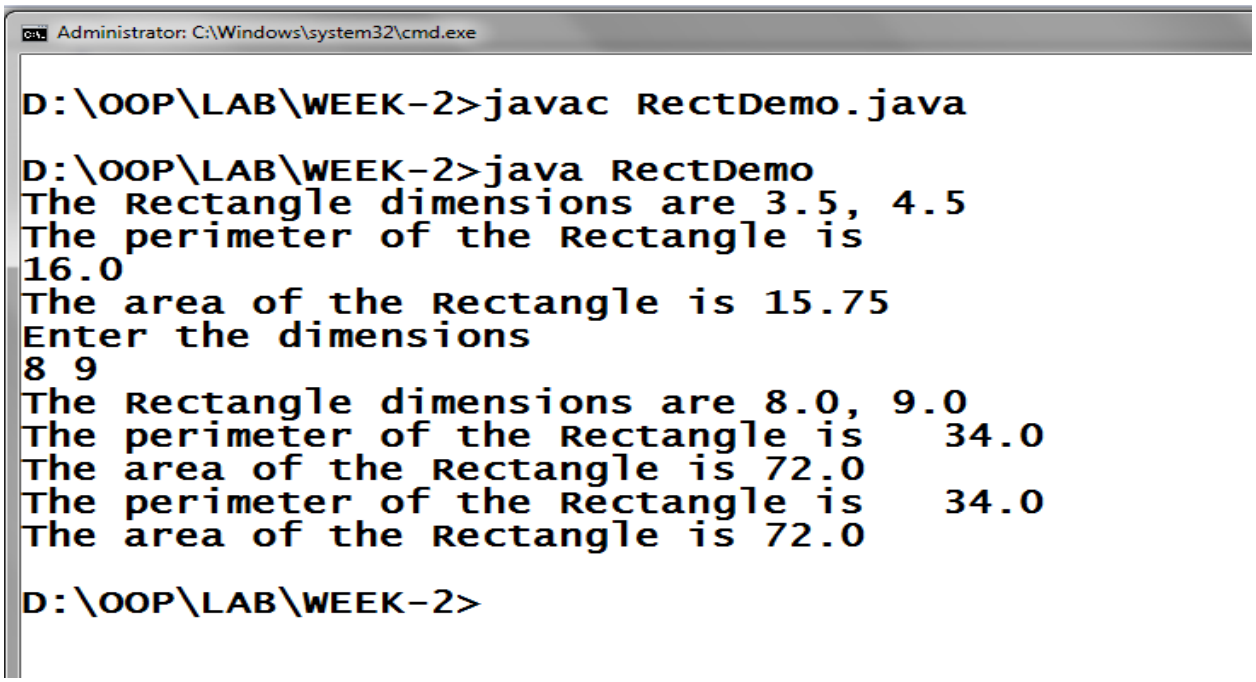
After the completion of the practice session, the student will be able to Implement OOP First principle – Encapsulation.

a) Define a class Rectangle with data member's length and width. Write methods to find perimeter and area of a rectangle. (class and object)

```
import java.util.*;
class Rect
{
        private double len,wid;
        public void setData(double d1 , double d2)
        {
                len=d1;    wid=d2;
        }
        public double perimeter( )
        {
                double res= 2*(len+wid);
                return res;
        }
          public double area( )
            {
                double res=(len*wid);
                return res;
             }
        public void display( )
        {
                System.out.println("The Rectangle dimensions are " +len  +", " + wid);
         }
} //close Rect


 public class RectDemo
        {
                public static void main(String args[])
                {

                Rect r1 ;  //class variable declaration
                //r1.len=2.3; //error -private member
                r1 = new Rect();  // Object
                r1.setData(3.5,4.5);
                r1.display();
```

```
                double x = r1.perimeter();
                System.out.println("The perimeter of the Rectangle is   " );
                System.out.println(x);
                System.out.println("The area of the Rectangle is " +r1.area());

                //reading data from keyboard
                System.out.println("Enter the dimensions");
                Scanner sc = new Scanner(System.in);
                double x1 = sc.nextDouble( );
                double x2 = sc.nextDouble( );
                 Rect  r2 = new Rect( ); // declartion and creation
                r2.setData(x1,x2);
                 r2.display();
              System.out.println("The perimeter of the Rectangle is   "+r2.perimeter());
             System.out.println("The area of the Rectangle is " +r2.area());

                 Rect r3 = r2;  //reference variable
               System.out.println("The perimeter of the Rectangle is   "+r3.perimeter());
                System.out.println("The area of the Rectangle is " +r3.area());
                 sc.close();
        }
    }
```

Output:

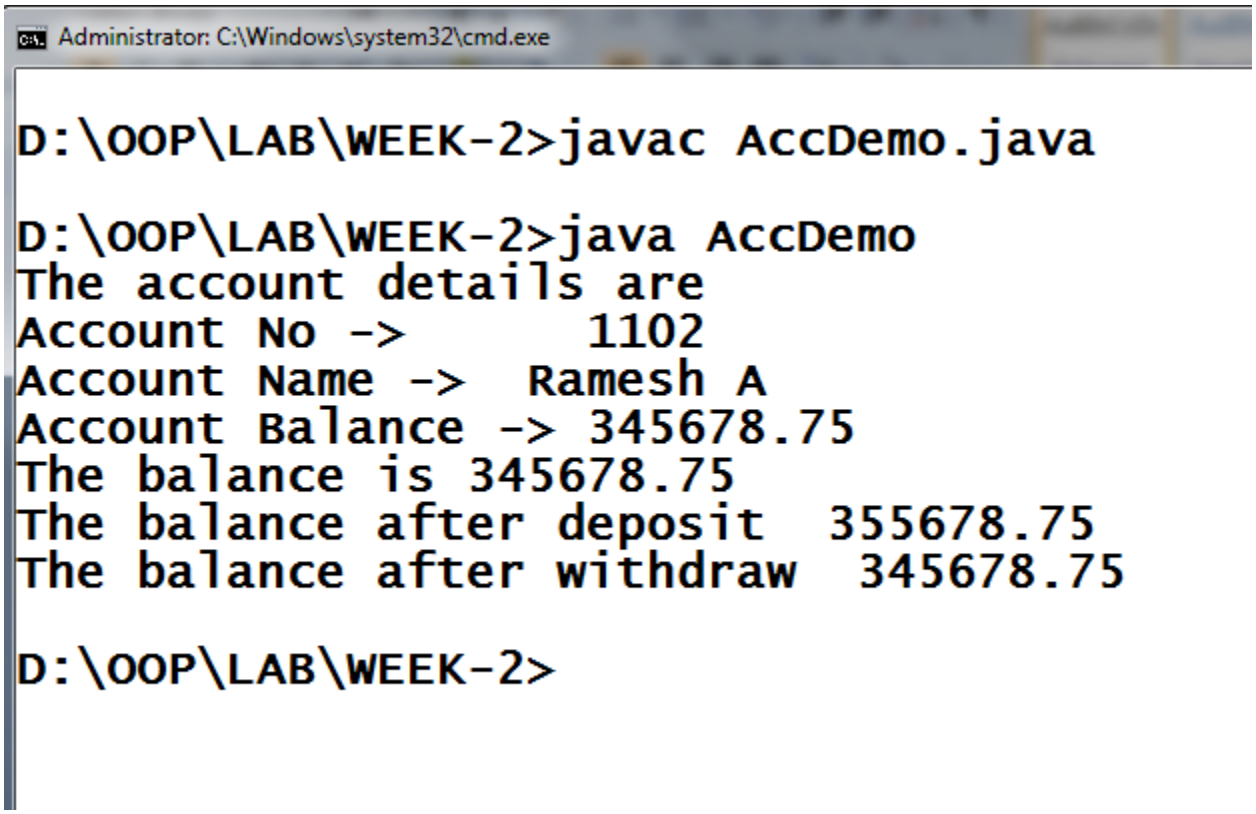


b) Create a class Account with data members name, acno and balance. Use appropriate methods to perform various operations like deposit,withdraw, balnceCheck.

```
class Account
{
        int acno;
        String name;
        double balance;
        public void  setData(int x , String y , double z)
        {
                acno=x;
                name=y;
                balance=z;
        }
        public double getBal( )
        {
                return (balance);
         }
      public void deposit( double amt)
         {
                 balance = balance + amt;
        }

        public void withdraw( double amt)
        {
                 balance = balance -amt;
        }

        public void display()
        {
                System.out.println("The account details are");
                System.out.println("Account No ->      " +acno);
                System.out.println("Account Name ->  " +name);
                System.out.println("Account Balance -> " +  balance);
         }
}
```

```
public class  AccDemo
{
        public static void main(String[] args)
        {
                Account ac1 = new Account();
                ac1.setData(1102 , "Ramesh A" , 345678.75);
                ac1.display();
                 System.out.println("The balance is " + ac1.getBal());
                 ac1.deposit(10000);
                  System.out.println("The balance after deposit  " + ac1.getBal());
                  ac1.withdraw(10000);
                  System.out.println("The balance after withdraw  " + ac1.getBal());

        }
}
```

Output:

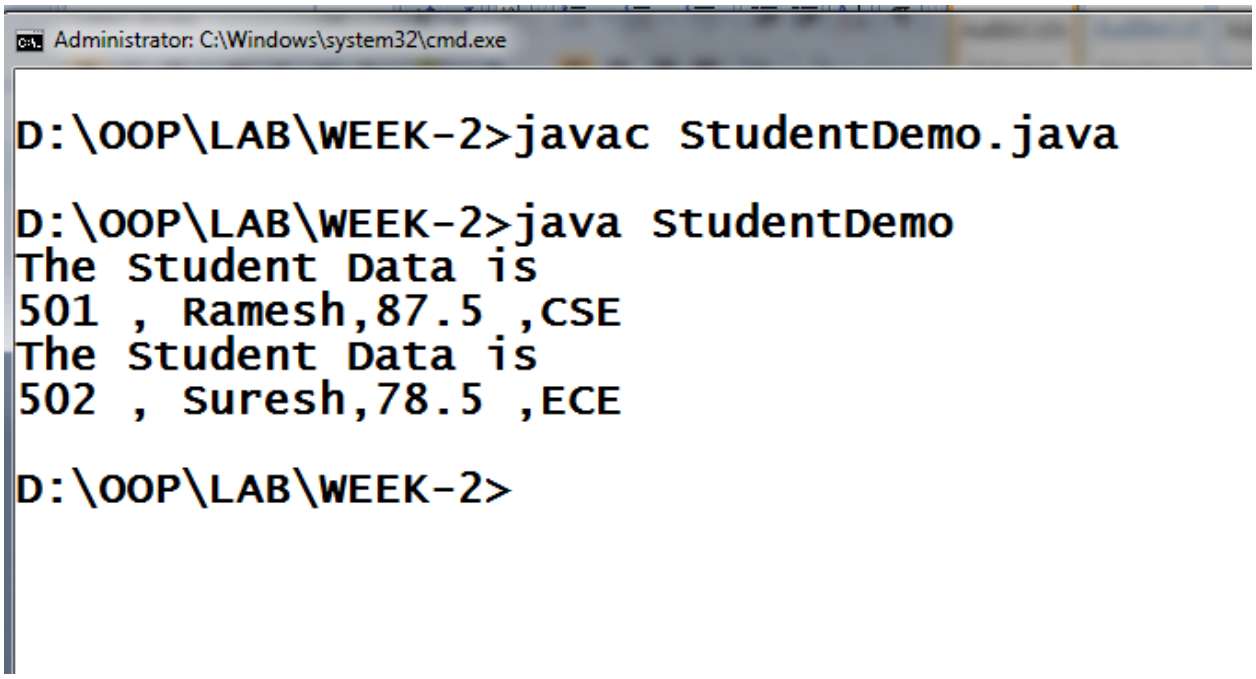


c) Create a class Student with appropriate data and methods using constructor.

```
class Student
 {
            int rollno;     String name, dept;
            double avg;
             Student()
             {     }
             Student(int x)
              {
                 rollno = x;
               }
           Student(int x , String y)
            {
                    this(x);
                    name=y;
            }
           Student(int x , String y , double z )
           {
                    this(x,y);
                    avg=z;
            }
           Student(int x , String y , double z , String p )
           {
                    this(x,y,z);
                    dept=p;
            }

            public void display()
              {
                      System.out.println("The Student Data is");
                            System.out.println(rollno + " , " +name +"," +avg + " ," +dept );
               }

}

public class StudentDemo
        {

                public static void main(String args[])

               {

                        Student s1 = new Student(501, "Ramesh" );
```

```
                s1.avg = 87.5;
                s1.dept = "CSE";
                s1.display();
                Student s2 = new Student(502, "Suresh" , 78.5 , "ECE");
                s2.display();
        }

}
```

Output:



WEEK-3

After the completion of the practice session, the student will be able to Implement OOP Second principle – Polymorphism.

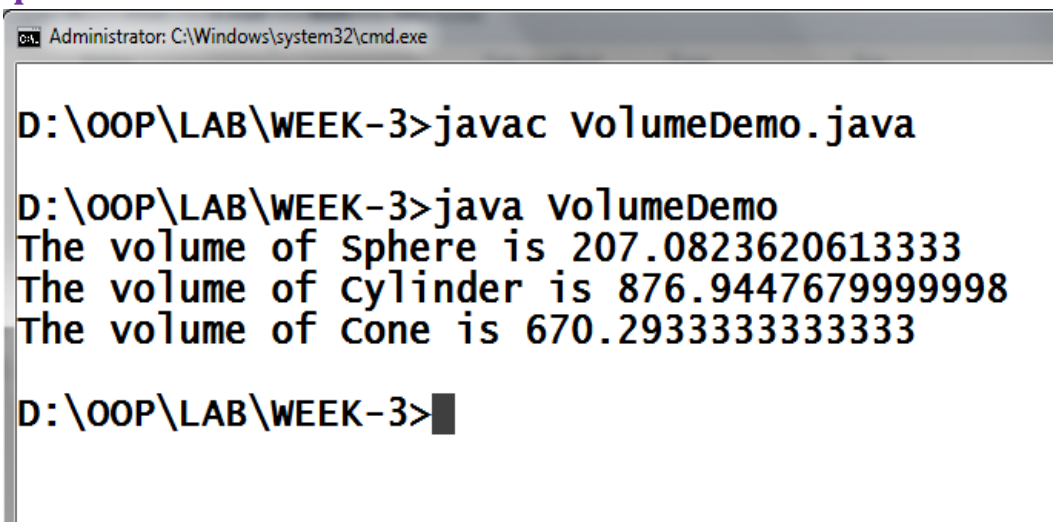
a) Create overloaded methods to find volume of Sphere, Cylinder & Cone.

/* Use Method overloading concept to solve the problem.

- Volume of Sphere: 4/3 p r3
- Volume of Cylinder: p r2h
- Volume of Cone: 1/3 p r2h */

```
class Volume
{
        final double PI= 3.142; //final to define constants
         double getVolume(double r)
          {
              return(4.0/3 * PI * r*r*r);
           }
          void getVolume(double r , double h)
              {
                                System.out.print("The volume of Cylinder is ");
                                 double res = PI*r*r*h;
                                 System.out.println(res);
               }
           double getVolume(int r , int h)
            {
                        return(1/3.0 * PI *r*r*h);
             }
    }
public class VolumeDemo
{
    public static void main(String args[])
      {
          Volume v1 = new Volume();
          System.out.println("The volume of Sphere is " + v1.getVolume(3.67));
          v1.getVolume(5.6,8.9);
          System.out.println("The volume of Cone is " +v1.getVolume(8,10));
      }
}
```

Output:

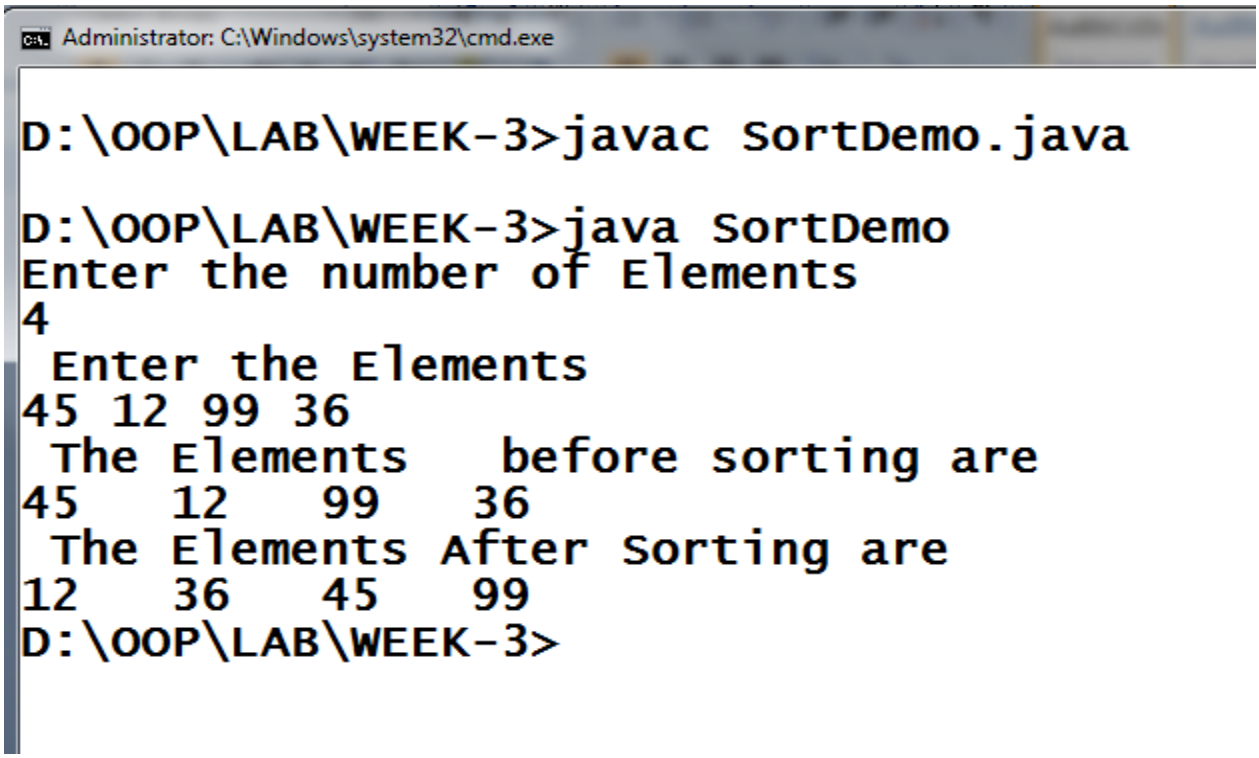


b) To sort given list of elements in ascending order.

```
import java.util.*;
public class  SortDemo
{
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);

                System.out.println("Enter the number of Elements");
                int n = sc.nextInt();
                int s[] = new int[n];
                System.out.println(" Enter the Elements");
                    for(int i=0; i< n;i++)
                          s[i] = sc.nextInt();
                System.out.println(" The Elements   before sorting are");
                      for(int i=0; i< s.length;i++)
                           System.out.print(s[i] + "   ");
                 for(int i= 0; i < s.length; i++)
                   {
                           for(int j = i + 1; j < s.length; j++)
                           {
                                   if( s[i] >s[j])
                                    {
                                      int  tmp = s[i];
                                      s[i] = s[j];
```

```
                                    s[j] = tmp;
                                }
                            }
                        }
                System.out.println(" \n The Elements After Sorting are");
                        for(int i=0; i< s.length;i++)
                          System.out.print(s[i] + "   “);
        }
}
```

Output:

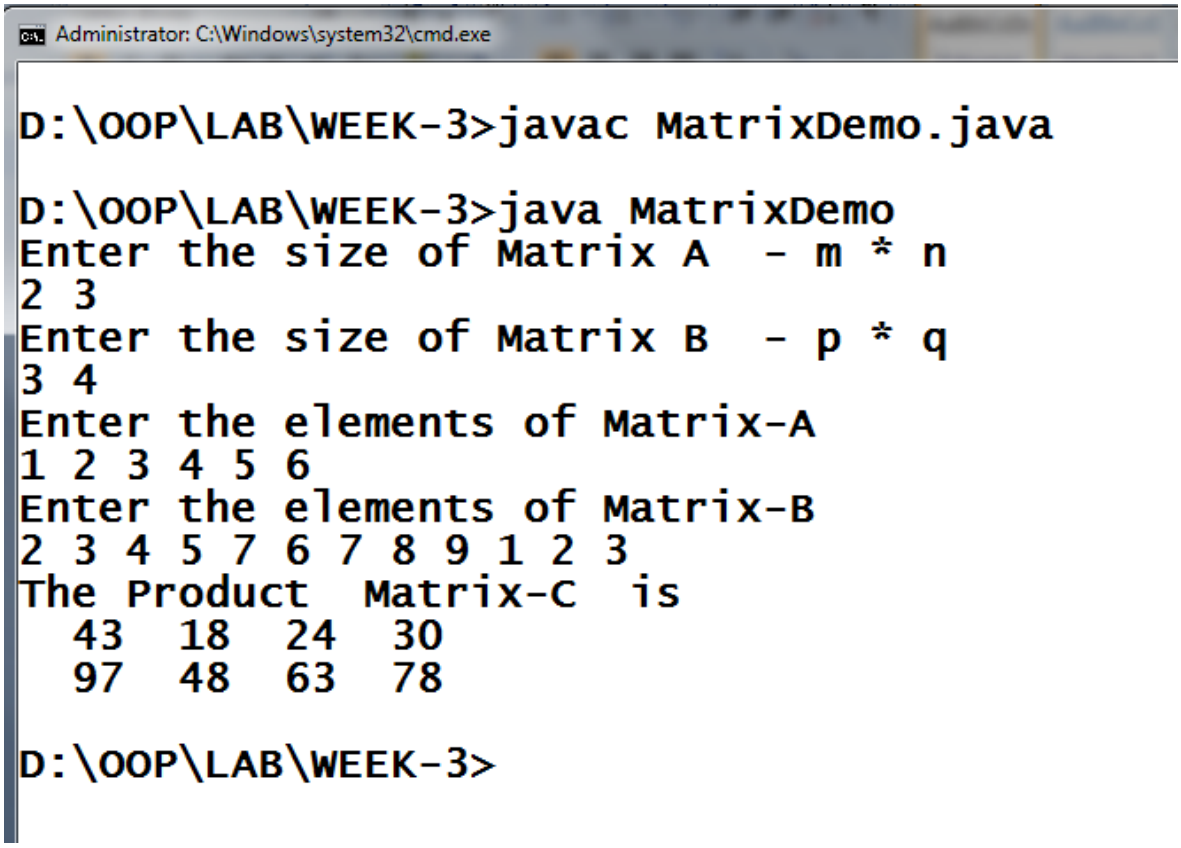


c) Read two matrices of size m*n , p*q , perform the multiplication of matrices.

```
import java.util.*;
public class  MatrixDemo
{
        public static void main(String  args[])
          {
                    Scanner sc = new Scanner(System.in);
                    System.out.println("Enter the size of Matrix A  - m * n");
                    int m = sc.nextInt();
                    int n = sc.nextInt();
                    System.out.println("Enter the size of Matrix B  - p * q");
                    int p = sc.nextInt();
                    int q = sc.nextInt();
                   if (n != p)
                      {
                              System.out.println("Multiplication cannot be performed");
                              System.exit(0);
                      }
                    int a[][] = new int[m][n];
                    int b[][] = new int[p][q];
                   int c[][] = new int[m][q];

                      System.out.println("Enter the elements of Matrix-A");
                                for(int i =0; i<m;i++)
                                      for(int j=0;j<n;j++)
                                              a[i][j] = sc.nextInt();
                       System.out.println("Enter the elements of Matrix-B");
                          for(int i =0; i<p;i++)
                              for(int j=0;j<q;j++)
                                              b[i][j] = sc.nextInt();
          for(int i =0 ; i<m ;i++)
                    {
                          for(int j=0;j<q;j++)
                            {
                              c[i][j]=0;
                               for(int k=0;k<p;k++)
                                  c[i][j] = c[i][j] + a[i][k]*b[k][j];
                            }
                    }
                     System.out.println("The Product  Matrix-C  is");
                          for(int i =0; i<m;i++)
```

```
                    {
                        for(int j=0;j<q;j++)
                            System.out.print("  " + c[i][j]);
                            System.out.println();
                    }

        }
}
```

Output:



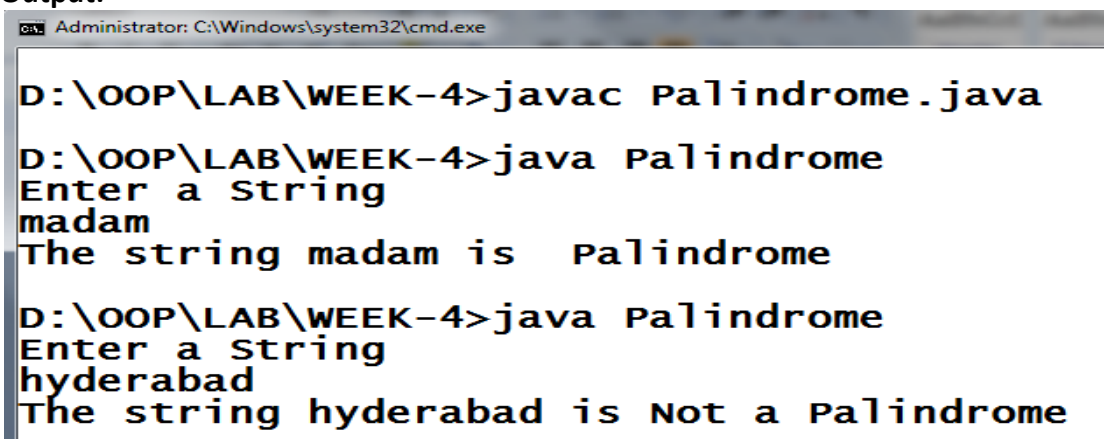
WEEK-4

After the completion of the practice session, the student will be able to Implement Java String Class.

a) Check a string is palindrome or not.

```
import java.util.*;
public class  Palindrome
{
        public static void main(String[] args)
        {
          Scanner sc = new Scanner(System.in);
          System.out.println("Enter a String");
          String s = sc.next();
           int len = s.length();
            int i , j=len-1;
           boolean flag=true;
            for(i=0 ; i < len/2 ; i++)
                {
                        if(s.charAt(i) != s.charAt(j))
                                {
                                        flag=false;
                                }
                                                j--;
                }
              if(flag==true)
                                System.out.println("The string " +s + " is  Palindrome");
                   else
                        System.out.println("The string " +s + " is Not a Palindrome");
        }
}
```

Output:



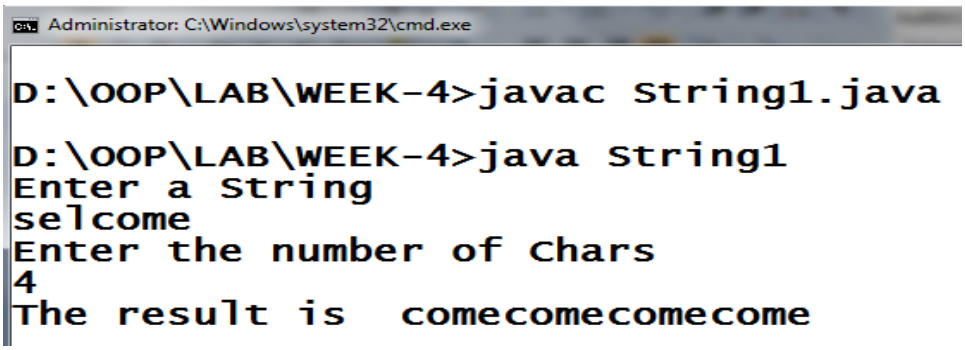
b) Given a string and an int n, return a string made of n repetitions of the last n characters of the string. You may assume that n is between 0 and the length of the string, inclusive. Write a Java program.

repeatEnd("Hello",3)→"llollollo

repeatEnd("Hello",2)→"lolo"

repeatEnd("Hello", 1) → "o"

```
import java.util.*;
public class String1
 {
    public static  String repeatEnd(String str, int n)
     {
          int len = str.length();
          String res = " ";
          for (int i = 0; i < n; i++)
           {
            res = res+str.substring(len - n, len);
          }
          return res;
        }
        public static void main(String args[])
          {
                  Scanner sc = new Scanner(System.in);
                  System.out.println("Enter a String");
                  String str = sc.next();
                 System.out.println("Enter the number of Chars");
                 int n = sc.nextInt();
                 String res = repeatEnd( str , n);
                 System.out.println("The result is " +res);

           }
        }
```

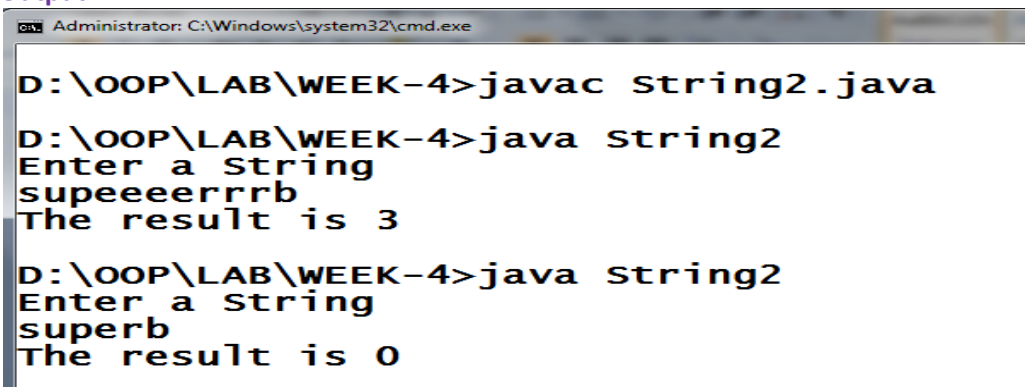


c) We'll say that a "triple" in a string is a char appearing three times in a row. Return the number of triples in the given string. The triples may overlap. Write a Java program.

countTriple("abcXXXabc")→1,
countTriple("xxxabyyyycd")→3
countTriple("a") → 0

```
import java.util.*;
public class String2
 {
    public static int countTriple(String str)
        {
           int len = str.length();
           int count = 0;
         for (int i = 0; i < len-2; i++)
          {
            char tmp = str.charAt(i);
            if (tmp == str.charAt(i+1) && tmp == str.charAt(i+2))
            count++;
           }
          return count;
         }
        public static void main(String args[])
         {
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter a String");

            String str = sc.next();
             int  res = countTriple( str );
            System.out.println("The result is " +res);
         }
}
```

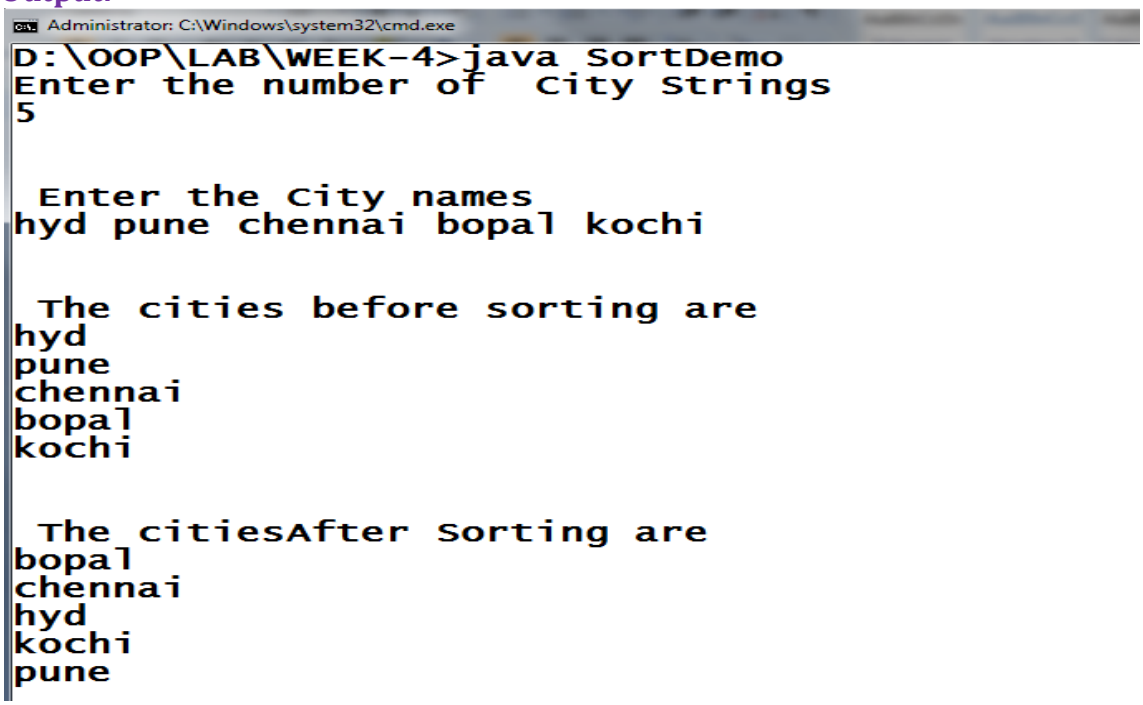
Output:



d) Read array of City names and Sort in dictionary order.(Ascending order).

```
import java.util.*;
public class  SortDemo
{
        public static void main(String[] args)
        {
           Scanner sc = new Scanner(System.in);
            System.out.println("Enter the number of  City Strings");
            int n = sc.nextInt();
            String s[] = new String[n];
            System.out.println("\n\n Enter the City names");
                    for(int i=0; i< n;i++)
                            s[i] = sc.next();
             System.out.println("\n\n The cities before sorting are");
                      for(int i=0; i< s.length;i++)
                           System.out.println(s[i]);
              for(int i= 0; i < s.length; i++)
                     {
                       for(int j = i + 1; j < s.length; j++)
                             {
                                if( s[i].compareTo(s[j]) > 0 )
                                  {
                                     String tmp = s[i];
                                      s[i] = s[j];
                                      s[j] = tmp;
                                  }
                        }
                   }
           System.out.println("\n\n The citiesAfter Sorting are");
                        for(int i=0; i< s.length;i++)
                                    System.out.println(s[i]);

        }
}
```

Output:


Administrator: C:\Windows\system32\cmd.exe
D:\OOP\LAB\WEEK-4>java SortDemo
Enter the number of City Strings
5
Enter the City names
hyd pune chennai bopal kochi
The cities before sorting are
hyd
pune
chennai
bopal
kochi
The citiesAfter Sorting are
bopal
chennai
hyd
kochi
pune

WEEK-5

After the completion of the practice session, the student will be able to implement OOP Third principle – Inheritance.

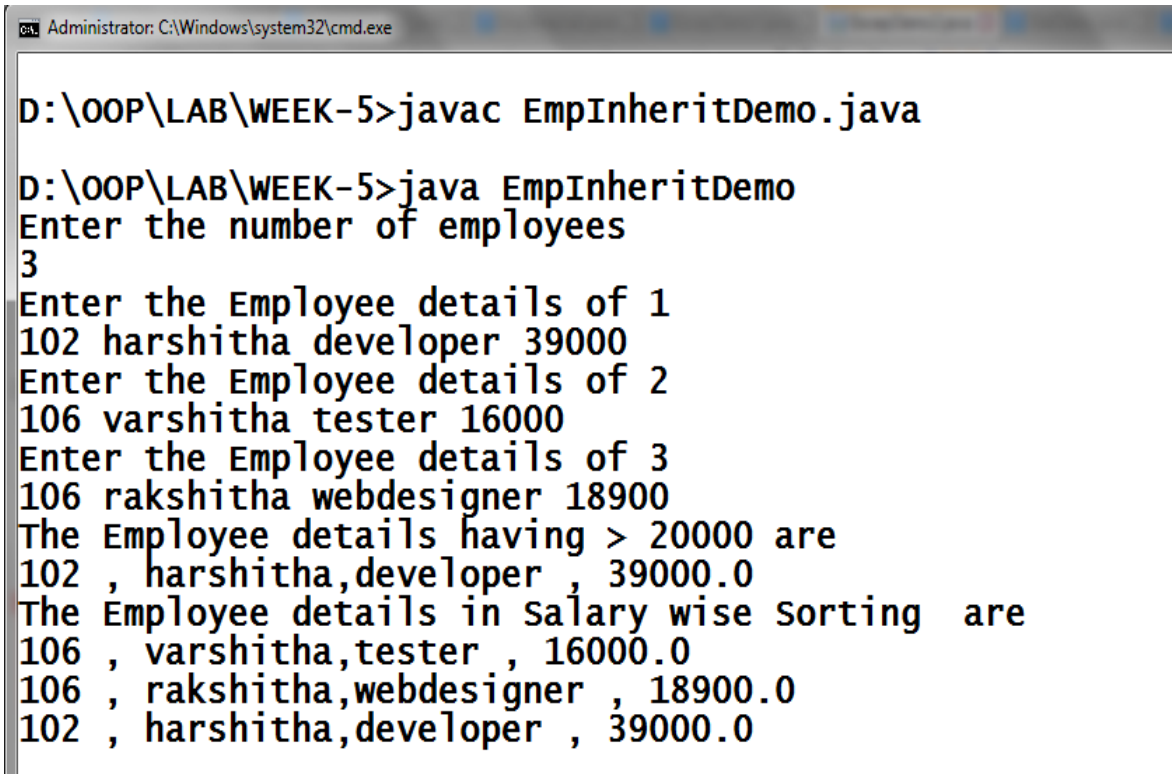
a) **Declare a class called Employee having employee _id and employee_name as members. Extend class Employee to have a subclass called Salary having designation and monthly_salary as members. Define following:**
   - **Required constructor**
   - **A method to find and display all details of employees drawing salary more than Rs.20000/-**
   - **main () method to create an array.**

```
import java.util.*;
class  Emp
{
        int empid;
        String empname;
        Emp(int no)
        {
                empid = no;
        }
        Emp(int no , String name)
        {
           this(no);
          empname = name;
    }
}

class Salary extends Emp
{

        String designation;

        double sal;

        Salary(int no , String name , String des , double salary)
        {
                super(no,name);
                designation = des;
                sal = salary;
        }
```

```
        void display()
        {
           System.out.println(empid + " , " + empname + "," +designation + " , " +sal);
        }
}

public class EmpInheritDemo
{
        public static void main(String args[])
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter the number of employees");
                int n = sc.nextInt();
                Salary s[] = new Salary[n];
                for(int i =0 ; i <n ; i++)
                {
                        System.out.println("Enter the Employee details of " +(i+1));
                        int id = sc.nextInt();
                        String name = sc.next();
                        String desg = sc.next();
                        double sal = sc.nextDouble();
                        s[i] = new Salary(id,name,desg,sal);
                }

     System.out.println("The Employee details having > 20000 are");
     for(int i=0;i<n ; i++)
        {
                        if(s[i].sal >20000)
                                s[i].display();
        }
    System.out.println("The Employee details in Salary wise Sorting  are");

 //sorting objects - additional

                    for(int i=0; i<n; i++)
                      {
                        for(int j=i+1;j<n;j++)
                                {
                                        if(s[i].sal > s[j].sal)
                                        {
                                                Salary temp = s[i];
```

```
                                        s[i] = s[j];
                                        s[j] = temp;
                                    }
                                }
                            }

                for(int i=0;i<n;i++)
                            s[i].display();

    }
}
```

Output:

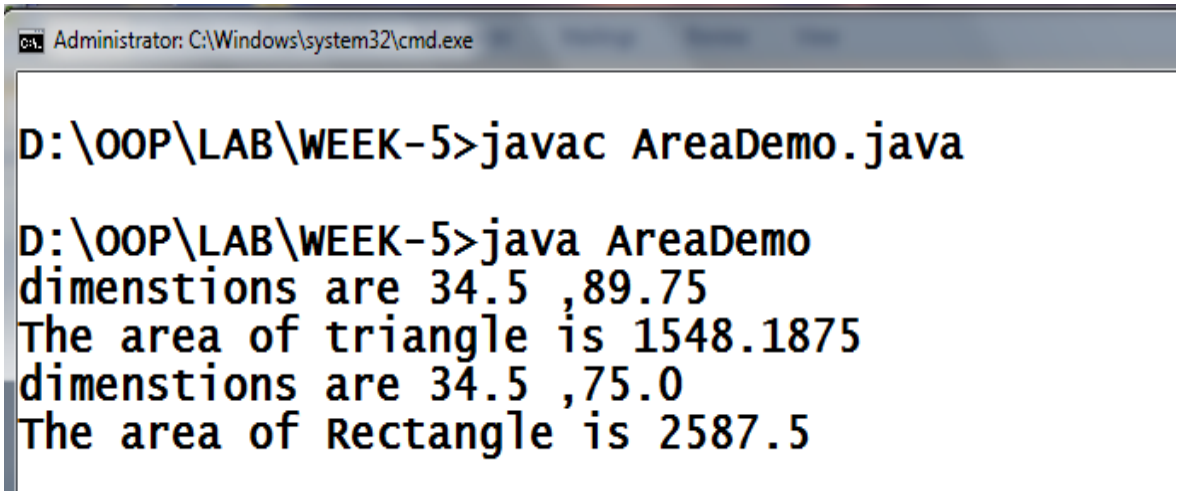


b) Write a Java program that create an abstract base class Shape with two members base and height, a member function for initialization and a function to compute shapeArea(). Derive two specific classes Triangle and Rectangle which override the function shapeArea(). Write a driver classes (main) to display the area of the triangle and the rectangle.(Use super keyword).

```
abstract class Shape
    {
          double base;
           double hgt;
       Shape(double b , double h)
         {
                  base = b;
                 hgt = h;
         }
       abstract double shapeArea();
       public void display()
         {
             System.out.println("dimenstions are " + base  + " ," +hgt);
          }
}

class Triangle extends Shape
{
          Triangle(double b , double h)
                {
                  super(b,h);
                 }
         double shapeArea()
         {
           return(0.5*base*hgt);
        }
 }

class Rectangl extends Shape
{
          Rectangl (double b , double h)
                {
                 super(b,h);
                }
```

```
                double shapeArea()
                  {
                        return(base*hgt);
                   }
}


public class AreaDemo
{
          public static void main(String args[])
        {
                        Shape t = new Triangle(34.5,89.75);  //upcasting
                        t.display();
                        System.out.println("The area of triangle is "+t.shapeArea());

                        Shape r = new Rectangl(34.5 , 75);  //upcasting
                        r.display();
                        System.out.println("The area of Rectangle is "+r.shapeArea());

        }

}
```

Output:



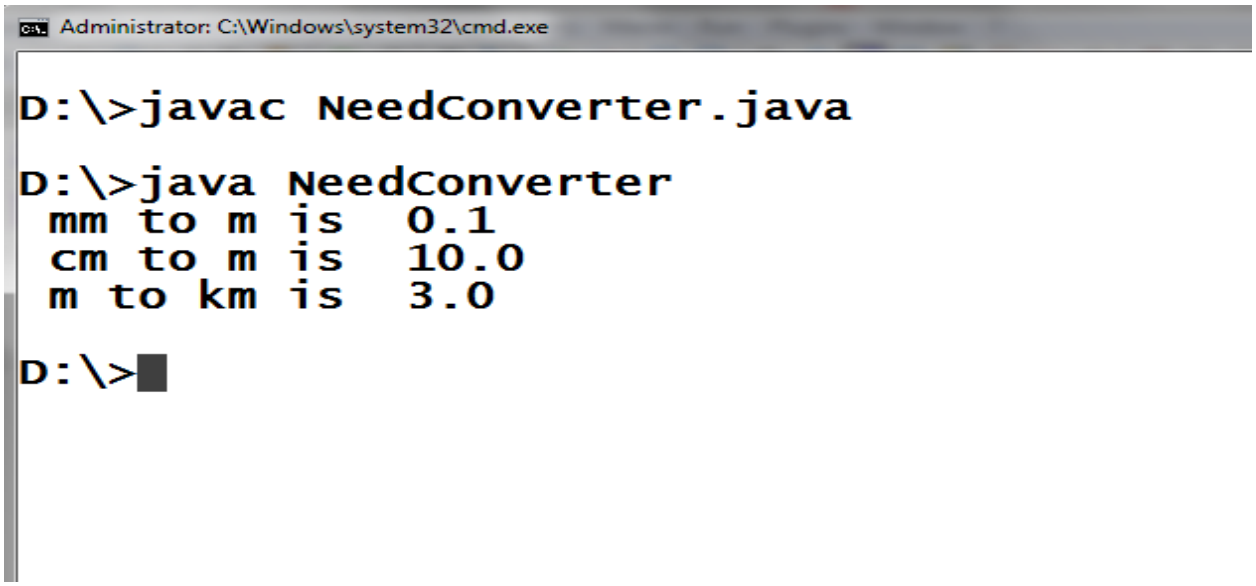
WEEK-6

After the completion of the practice session, the student will be able to implement Packages and Interfaces.

a) Create a Package Measure; in which store a class named Convertor that contains methods to convert mm to cm, cm to m and m to km. Define a class Need_Convertor that imports the Convertor class, now store Need_Convertor outside the package Measure. Perform path settings accordingly.

```
package Measure;
public class Converter
{
        public float mmtom(float mm)
        {
                float m=(mm/1000);
                return m;
        }
        public float cmtom(float cm)
        {
                float m=(cm/100);
                return m;
        }
        public float mtokm(float m)
        {
                float km=(m/1000);
                return km;
        }
}


        import Measure.*;
        public class NeedConverter
        {
                public static void main(String args[])
                {
                        Converter c=new Converter();
                        System.out.println(" mm to m is  "+c.mmtom(100));
                        System.out.println(" cm to m is  "+c.cmtom(1000));
                        System.out.println(" m to km is  "+c.mtokm(3000));
                }
        }
```

Output:



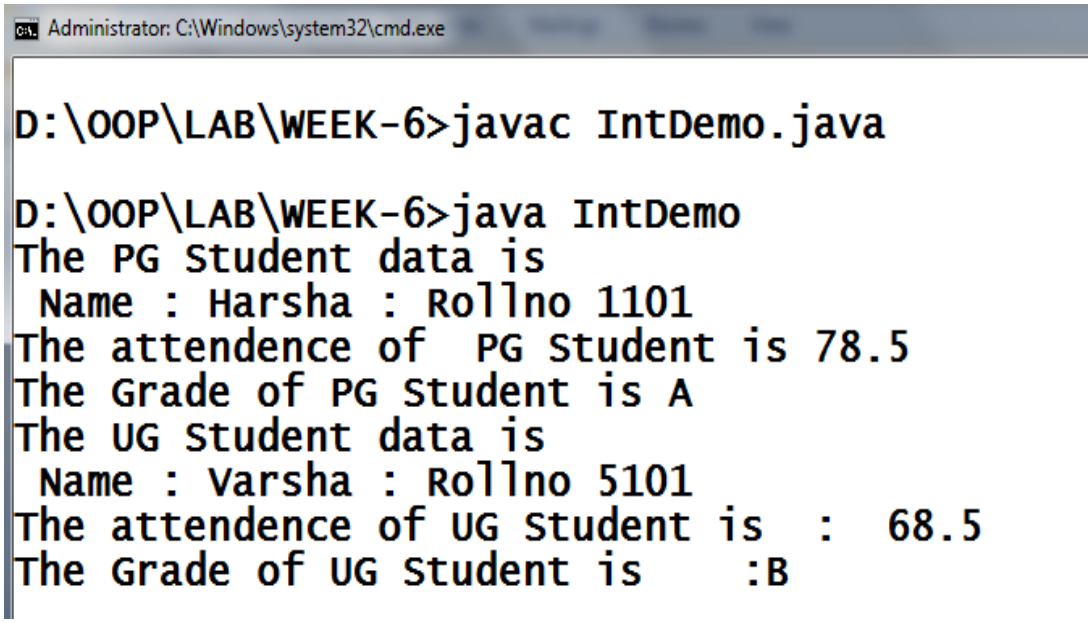
b) Write a Java program that implements an interface Student which has two methods displayGrade() and attendance(). Implement two classes PG_Student and UG_Student with necessary inputs of data.

```
interface Student
{
        void displayGrade();
        void attendence();
}

class  PGStudent implements Student
{
       String name;
         int rollno;
         String grade;
       double att;
        PGStudent(String n , int r , String g , double a)
        {
                name = n;
                rollno=r;
                grade=g;
                att = a;
         }
```

```
 public void display()
        {
                System.out.println("The PG Student data is ");
                 System.out.println(" Name : " +name  +" : "+ "Rollno " + rollno);
        }
   public void displayGrade()
        {
                System.out.println("The Grade of PG Student is " +grade);
        }

        public void attendence()
        {
                System.out.println("The attendence of  PG Student is " +att);
        }

}
class  UGStudent implements Student
{
    String name;
         int rollno;
         String grade;
    double att;

        UGStudent(String n , int r , String g , double a)
        {
                name = n;
                rollno=r;
                grade=g;
                att = a;
   }

  public void display()
        {
                System.out.println("The UG Student data is ");
                 System.out.println(" Name : " +name  +" : "+ "Rollno " + rollno);
        }

        public void displayGrade()
        {
                System.out.println("The Grade of UG Student is    :" +grade);
        }
```

```
        public void attendence()
        {
                System.out.println("The attendence of UG Student is  :  " +att);
        }


}
public class IntDemo
{
        public static void main(String args[])
        {
                PGStudent s = new PGStudent("Harsha", 1101, "A",78.5);
                s.display();
                s.attendence();
                s.displayGrade();

                UGStudent u = new UGStudent("Varsha",5101,"B",68.5);
                u.display();
                u.attendence();
                u.displayGrade();

        }
}
```

Output:



WEEK-7

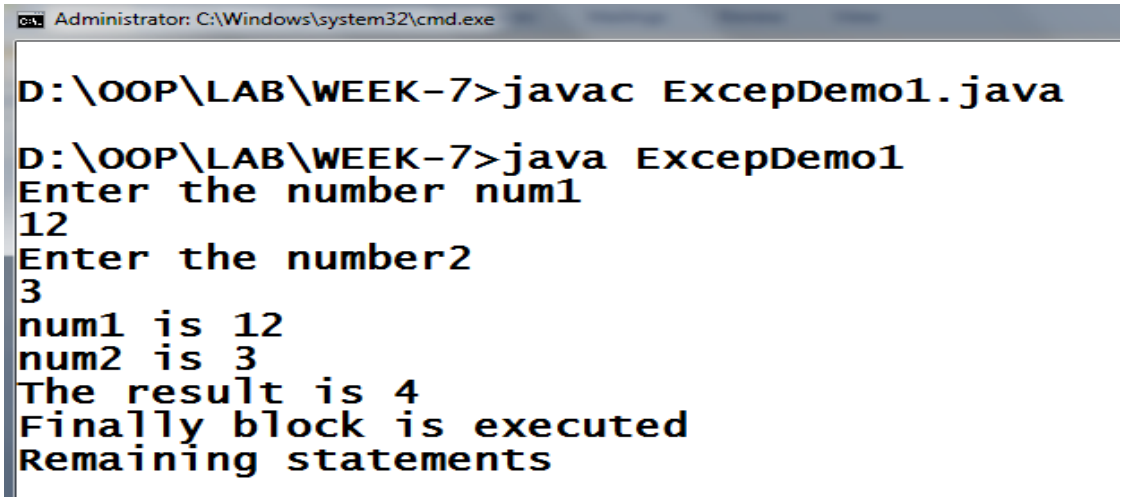
After the completion of the practice session, the student will be able to implement Implement Exception Handling .

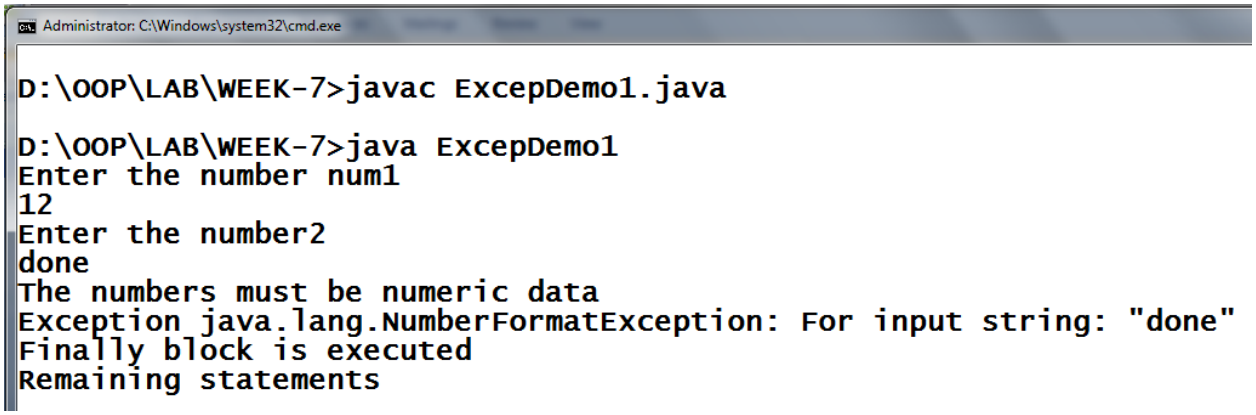
a) **Creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 andNum2 is displayed in the Result field when the Div- id button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 is Zero, the program would throw an ArithmeticException. Display the exception in a message dialog box.**

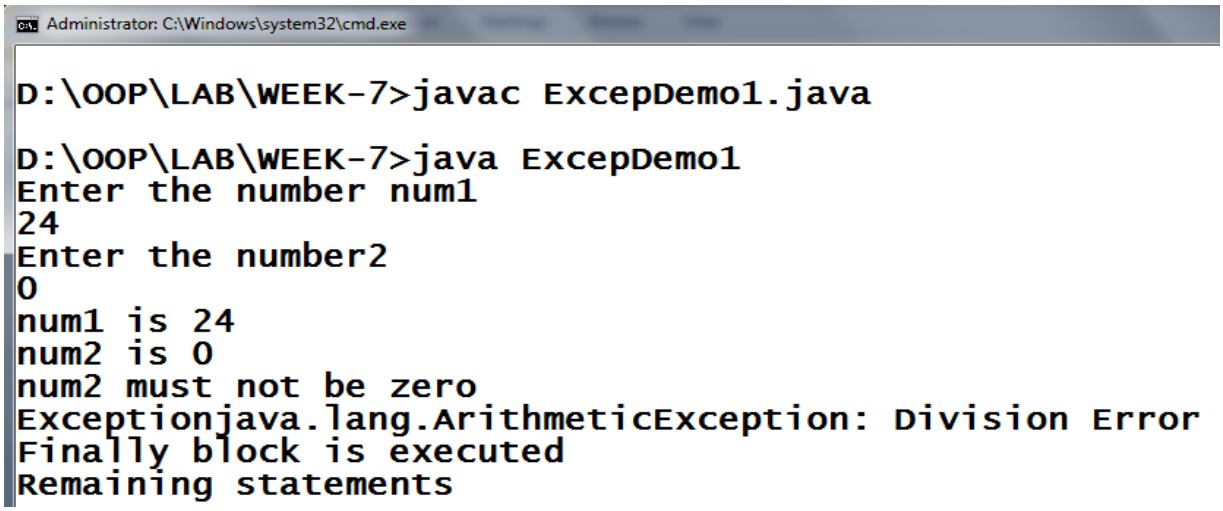
```
import java.util.*;
public class ExcepDemo1
{
        public static void main (String args[])
    {
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter the number num1");
            String s1 = sc.next();
             System.out.println("Enter the number2");
             String s2=sc.next();
              try
                {
                   int num1 = Integer.parseInt(s1);
                   int num2 = Integer.parseInt(s2);
                   System.out.println("num1 is " +num1);
                   System.out.println("num2 is " +num2);
                    if(num2 ==0)
                         throw new ArithmeticException ("Division Error");

                     int res = num1/num2;
                          System.out.println("The result is " +res);
                   }
                catch(NumberFormatException e)
                   {
                    System.out.println("The numbers must be numeric data");
                      System.out.println("Exception " +e);
                    }
```

```
                    catch(ArithmeticException e)
                    {
                        System.out.println("num2 must not be zero");
                         System.out.println("Exception" +e);
                     }

                    finally
                    {
                      System.out.println("Finally block is executed");
                     }
                System.out.println("Remaining statements");
                }
        }
```

Output:





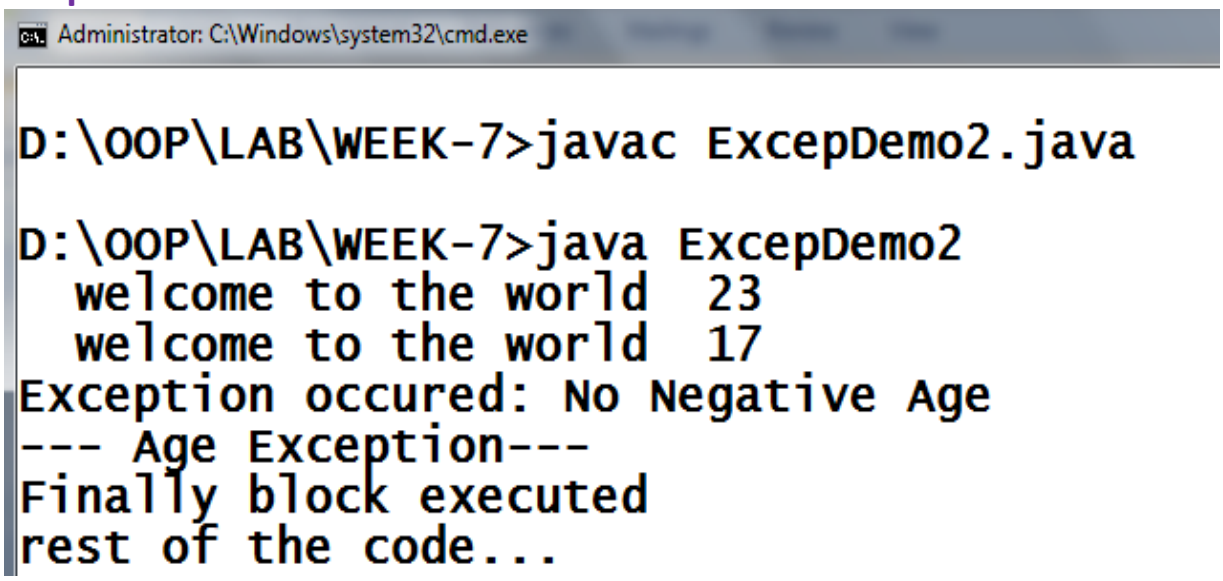


b) In the CustomExceptionTest class, the age is expected to be a positive number. It would throw the user defined exception NegativeAgeException if the age is assigned a negative number.

```
class NegativeAgeException extends Exception
    {
        NegativeAgeException(String s)
          {
            super(s);
          }

        public  String toString()
                {
          return "Age Exception";
        }
      }

public class ExcepDemo2
  {
     static void validate(int age) throws NegativeAgeException
        {
            if(age<0)
              throw new NegativeAgeException("  not valid  "+age);
            else
               System.out.println("  welcome to the world  " +age);
         }
```

```
    public static void main(String args[])
      {
       try{
              validate(23);
              validate(17);
              validate(-12);
               validate(25);  // Not Executed
           }
       catch(Exception m)
         {
             System.out.println("Exception occured: No Negative Age");
        System.out.println("--- " + m + "---"); // Description of Message
          }
        finally {
                System.out.println("Finally block executed");
            }
       System.out.println("rest of the code...");
     }
}
```

Output:

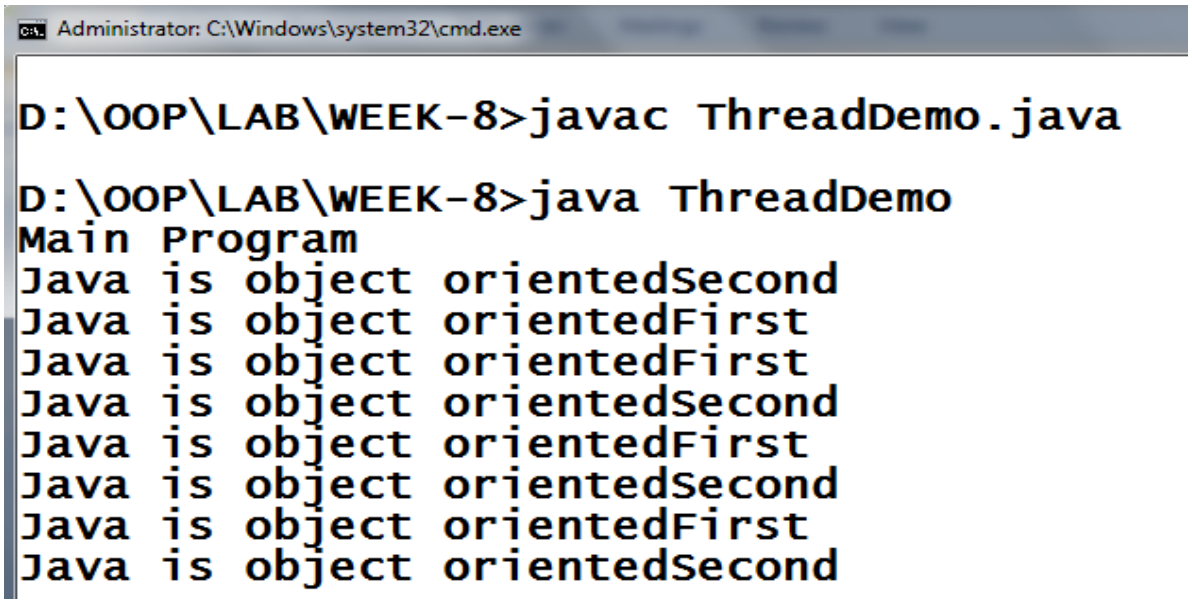


WEEK-8

After the completion of the practice session, the student will be able to Develop applications on Multithreaded Programming and thread synchronization.

a) **Create a multithreaded java program by creating a subclass of Thread and then creating, initializing, and staring two Thread objects from your class. The threads will execute concurrently and display "Java is object oriented" in console window.**

```
class NewThread extends Thread
{
        NewThread(String name)
        {
                super(name);
                //start();
        }

        public void run()
        {
              try
               {
                  for(int i=1;i<=4;i++ )
                        {
                 System.out.println("Java is object oriented" + getName());
                           sleep(1000);
                        }
                 }
                 catch (InterruptedException ie)
                 {
                System.out.println("Child Thread - Exception caught");
                 }
        }
}
public class ThreadDemo
{
          public static void main(String args[])
      {
                        NewThread t1 = new NewThread("First");
                        NewThread t2 = new NewThread("Second");

                                t1.start();
                                t2.start();
                                System.out.println("Main Program");
                    }
}
```



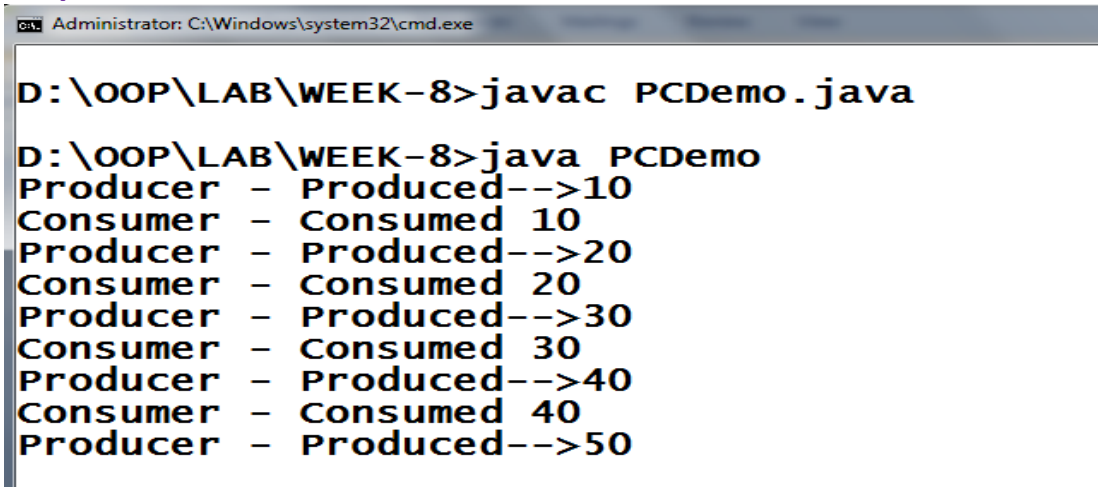
b) **Implement the concept of producer consumer problem using thread synchronization.**

```
class Buffer
{
        int item;
        boolean produced = false;
        synchronized void  produce(int x)
        {
                if(produced)
                {
                        try{
                                wait();
                           }
                        catch(InterruptedException ie)
                        {
                                System.out.println("Exception Caught");
                        }
                }

                item =x;
                System.out.println("Producer - Produced-->" +item);
                produced =true;
                notify();
        }
```

```
        synchronized int consume()
        {
                if(!produced)
                {
                        try{
                                wait();
                           }
                         catch(InterruptedException ie)
                        {
                                System.out.println("Exception Caught " +ie);
                        }
                }

                System.out.println("Consumer - Consumed " +item);
                produced = false;
                notify();
                return item;
  }
}

class Producer extends Thread
{

        Buffer b;
   Producer( Buffer b)
        {
                this.b = b;
                start();
        }

  public void run()
        {
                b.produce(10);
                b.produce(20);
                b.produce(30);
                b.produce(40);
                b.produce(50);

        }
}
```

```
class Consumer extends Thread
{
        Buffer b;

        Consumer(Buffer b)
        {
                this.b = b;
                start();
        }

        public void run()
        {

        b.consume();
        b.consume();
          b.consume();
               b.consume();

    }
}

public class PCDemo
        {
                public static void main(String args[])
                {
                        Buffer b = new Buffer();  //Synchronized Object
                        new Producer(b);
                        new Consumer(b);
          }
        }
```

Output:



WEEK-9

After the completion of the practice session, the student will be able to implement Collection Frameworks to retrieve data.

a) Use an ArrayList to manage Employee objects for insertion, display and remove.

```
import java.util.*;
class Employee
{
        int eid;
        String ename;
        double sal;

          public Employee(int x, String y, double z)
           {
            eid=x;
            ename=y;
            sal = z;
           }
     }

public class EmpArrayList
{
         public static void main(String[] args)
          {
              ArrayList<Employee>  list =new ArrayList<Employee>();
            //Creating Employees
            Employ e1=new Employee(101,"A.Harsha",75000.50);
            Employee e2=new Employee(102,"B.Varsha",85000.50);
            Employee e3=new Employee(103,"C.Sirisha",95000.50);
            Employee e4=new Employee(104,"D.Sandeep",195000.50);
           //Adding Employees to list
                        list.add(e1);
                        list.add(e2);
                        list.add(e3);
                        list.add(e4);

          //Displaying Number of Employees
          System.out.println("\n The number of employees is ->" +list.size());
```

```
            //Displaying Details of Employees
             System.out.println("\n The employess data is \n");
             for(Employee e:list)
              {
                  System.out.println(e.eid+":"+e.ename+":"+e.sal);
                  System.out.println();
               }
            //Deleting an Employee
                  list.remove(2);
System.out.println("\n After removing number of employees are ->" + list.size());
            //Displaying Details of Employees
                System.out.println("\n The employess data after removing is \n");
                  for(Employee e:list)
                  {
                          System.out.println(e.eid+":"+e.ename+":"+e.sal);
                           System.out.println();
                  }
             }
      }
```

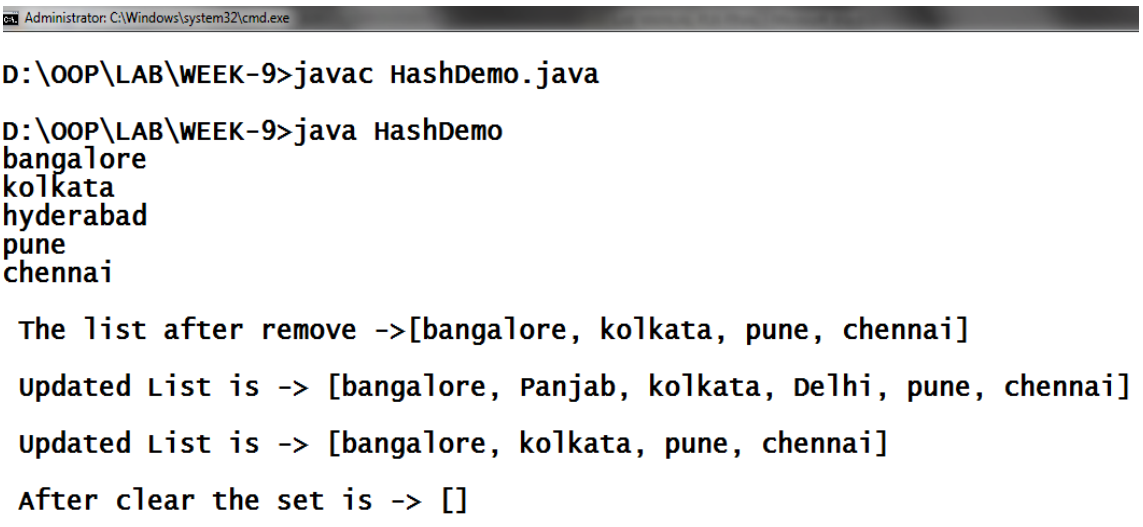
Output:

```
D:\OOP\LAB\WEEK-9>java EmpArrayList
 The number of employees is ->4
 The employess data is
101:A.Harsha:75000.5
102:B.Varsha:85000.5
103:C.Sirisha:95000.5
104:D.Sandeep:195000.5

 After removing number of employees are ->3
 The employess data after removing is
101:A.Harsha:75000.5
102:B.Varsha:85000.5
104:D.Sandeep:195000.5
```

b) Use HashSet methods to perform operations on collection of data.

```
import java.util.*;
class HashDemo
{
  public static void main(String args[])
   {
        //Creating HashSet
        HashSet<String> set=new HashSet<String>();

          //Adding Elements to HashSet - ignores duplicates
        set.add("hyderabad");
        set.add("hyderabad");
        set.add("bangalore");
        set.add("chennai");
        set.add("kolkata");
        set.add("kolkata");
        set.add("pune");

        Iterator<String> i=set.iterator();
        while(i.hasNext())
        {
         System.out.println(i.next());
        }

    //Removing specific element from HashSet
        set.remove("hyderabad");
     //Displaying set
       System.out.println("\n The list after remove ->" +set);

    //adding another set
       HashSet<String> set1=new HashSet<String>();
        set1.add("Panjab");
        set1.add("Delhi");
        set.addAll(set1);
        System.out.println("\n Updated List is -> "+set);
  //removing new set from list
         set.removeAll(set1);
         System.out.println("\n Updated List is -> "+set);
  //Removing all the elements available in the set
        set.clear();
        System.out.println("\n After clear the set is -> "+set);

              }
}
```

Output:


Administrator: C:\Windows\system32\cmd.exe
D:\OOP\LAB\WEEK-9>javac HashDemo.java
D:\OOP\LAB\WEEK-9>java HashDemo
bangalore
kolkata
hyderabad
pune
chennai
The list after remove ->[bangalore, kolkata, pune, chennai]
Updated List is -> [bangalore, Panjab, kolkata, Delhi, pune, chennai]
Updated List is -> [bangalore, kolkata, pune, chennai]
After clear the set is -> []

WEEK-10

After the completion of the practice session, the student will be able to implement Collection Frameworks to retrieve data.

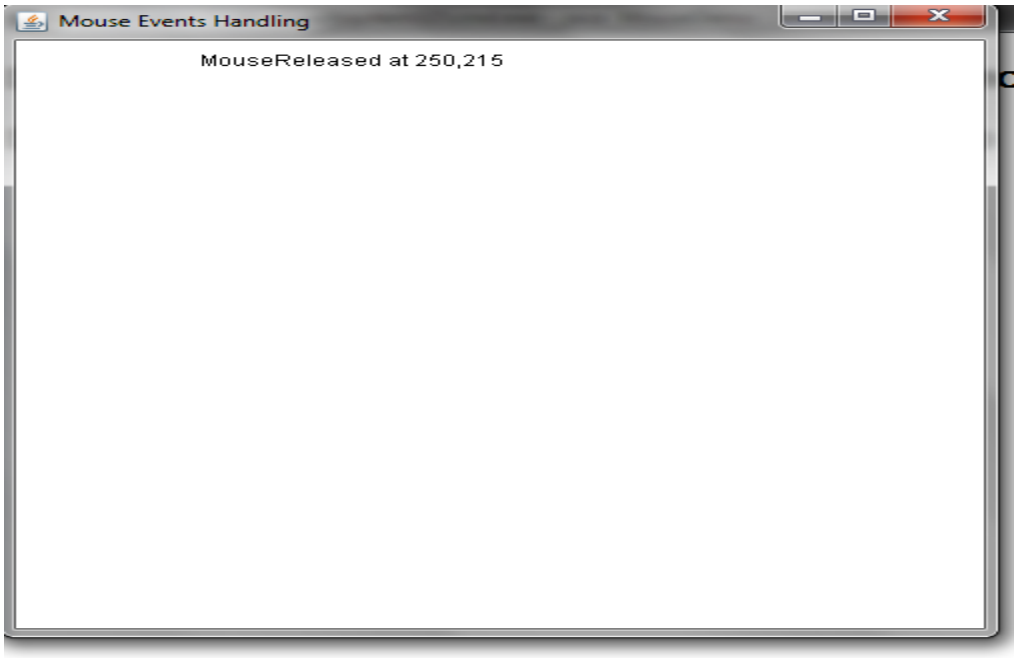
a) **Implement MouseListener and MouseMotionListener.**

```
import java.awt.*;
import java.awt.event.*;

public class MouseDemo extends Frame implements MouseListener,MouseMotionListener
  {

            int x=0, y=0;
            String msg= "";

        MouseDemo(String title)
            {
              super(title);
              addMouseListener(this);
              addMouseMotionListener(this);
              setSize(500,500);
              setVisible(true);
             //window close
              addWindowListener(new WindowAdapter()
                    {
                          public void windowClosing(WindowEvent e)
                                {
                                      dispose();
                                 }
                    }       );

        }

      public void mouseClicked(MouseEvent e)
      {

            msg= "MouseClicked";
            x = e.getX();
            y = e.getY();
            repaint();
      }
```

```
public void mousePressed(MouseEvent e)
{
        msg= "MousePressed";
        x = e.getX();
        y = e.getY();
        repaint();

}

public void mouseReleased(MouseEvent e)
{
        msg = "MouseReleased";
        x = e.getX();
        y = e.getY();
        repaint();

}

public void mouseEntered(MouseEvent e)
{
        msg= "MouseEntered";
        x = e.getX();
        y = e.getY();
        repaint();

}

public void mouseExited(MouseEvent e)
{
        msg= "MouseExited";
        x = e.getX();
        y = e.getY();
        repaint();
}

public void mouseMoved(MouseEvent e)
{
        msg= "*";
        x = e.getX();
        y = e.getY();
        repaint();
}
```

```
    public void mouseDragged(MouseEvent e)
    {
        msg= "#";
        x = e.getX();
        y = e.getY();
        repaint();
    }

    public void paint(Graphics g)
    {
        g.drawString(msg + " at " + x + "," + y, 100,50);
    }

    public static void main(String[] args)
    {

        MouseDemo  f = new MouseDemo("Mouse Events Handling");
    }

}
```

Output:

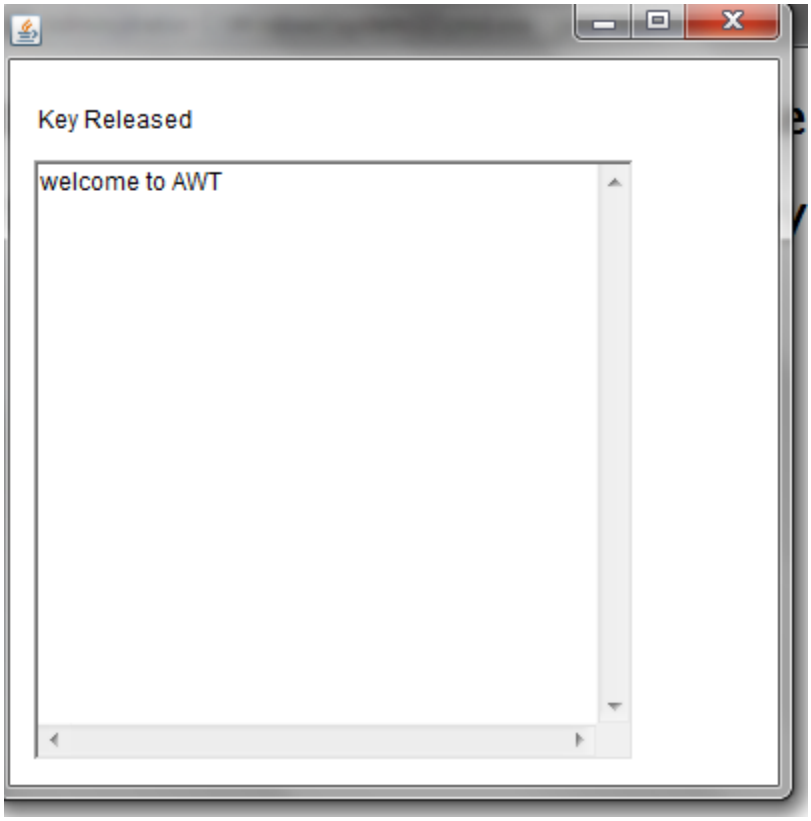


b) **Implement KeyListener to handle key events.**

```
import java.awt.*;
import java.awt.event.*;
public class KeyDemo extends Frame implements KeyListener
 {
                Label l;
                TextArea area;
                KeyDemo()
                {
                        l=new Label();
                        l.setBounds(20,50,100,20);
                        area=new TextArea();
                        area.setBounds(20,80,300, 300);
                        area.addKeyListener(this);

                        add(l);add(area);
                        setSize(400,400);
                        setLayout(null);
                        setVisible(true);

                        addWindowListener(new WindowAdapter()
                        {
                            public void windowClosing(WindowEvent e)
                             {
                               dispose();
                             }
                        } );
                }

                public void keyPressed(KeyEvent e)
                {
                        l.setText("Key Pressed");
                }
                public void keyReleased(KeyEvent e)
                {
                        l.setText("Key Released");
                }

                public void keyTyped(KeyEvent e)
                {
                        l.setText("Key Typed");
                }
```

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

Output:



WEEK-11

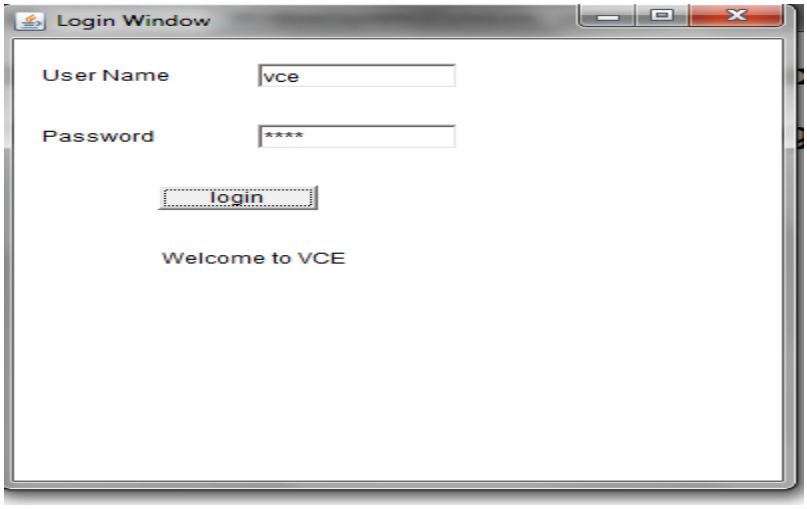
After the completion of the practice session, the student will be able to Develop GUI applications using AWT.

a) Create a Simple login window to validate a user with name and password.

```
import java.awt.*;
import java.awt.event.*;
public class LoginDemo extends Frame implements ActionListener
 {
                Label l1,l2,l3;
                TextField t1,t2;
                Button b1;
                LoginDemo(String name)
                {
                        super(name);
                        l1=new Label("User Name");
                        l1.setBounds(20,50,100,20);
                        //x,y -> top left point , width,height are dimensions
                        l2=new Label("Password");
                        l2.setBounds(20,100,100,20);

                        t1= new TextField();
                        t1.setBounds(130,50,100,20);

                        t2= new TextField();
                        t2.setBounds(130,100,100,20);
                        t2.setEchoChar('*');

                        b1= new Button("login");
                        b1.setBounds(80,150,80,20);

                        l3 = new Label("->");
                        l3.setBounds(80,200,200,20);
                        add(l1);add(t1);
                        add(l2);add(t2);
                        add(b1);add(l3);

                        b1.addActionListener(this);
                        setSize(400,400);
                        setLayout(null);
                        setVisible(true);
```

```
        addWindowListener(new WindowAdapter()
         {
         public void windowClosing(WindowEvent e)
          {
            dispose();
          }});

}

   public void actionPerformed(ActionEvent e)
         {
                  String uname = t1.getText();
                   String  pwd = t2.getText();

                  if(uname.equals("vce") && pwd.equals("root"))
                           l3.setText("Welcome to VCE");
                  else
                           l3.setText("Invalid Username or Password");
   }


   public static void main(String[] args)
         {
      new LoginDemo("Login Window");
   }
}
```

Output:



b) Using Grid Layout design a Simple calculator with appropriate event handling.

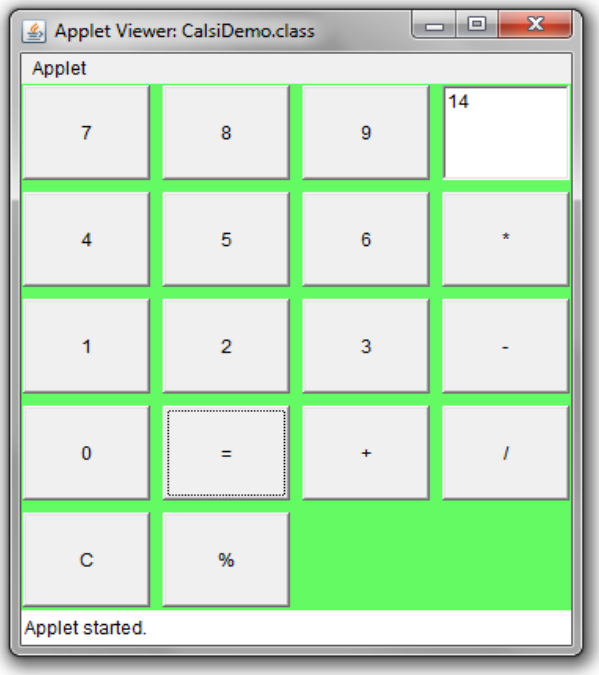
```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

//<applet code = "CalsiDemo.class"  width = 350 height = 350></applet>

public class CalsiDemo extends Applet implements ActionListener
{
        TextField t1;
        Button b1,b2,b3,b4,b5,b6,b7,b8,b9,b0;
        Button add,sub,mul,div, eql,clr,rem;
        String msg="",tmp;
        int a, b;
        public void init()
        {
                Color c1 = new Color(100,250,100);
                setBackground(c1);

                GridLayout gl= new GridLayout(5,4,8,8);
                setLayout(gl);
                t1=new TextField();
                b1=new Button("1");
                b2=new Button("2");
                b3=new Button("3");
                b4=new Button("4");
                b5=new Button("5");
                b6=new Button("6");
                b7=new Button("7");
                b8=new Button("8");
                b9=new Button("9");
                b0=new Button("0");
                add=new Button("+");
                sub=new Button("-");
                div=new Button("/");
                mul=new Button("*");
                eql=new Button("=");
                clr = new Button("C");
                rem = new Button("%");
```

```
        add(b7);
        add(b8);
        add(b9);
        add(t1);

        add(b4);
        add(b5);
        add(b6);
        add(mul);

        add(b1);
        add(b2);
        add(b3);
        add(sub);

        add(b0);
        add(eql);
        add(add);
        add(div);
        //t1.setBounds(30,30,200,40);
        add(clr);
        add(rem);

        b0.addActionListener(this);
        b1.addActionListener(this);
        b2.addActionListener(this);
        b3.addActionListener(this);
        b4.addActionListener(this);
        b5.addActionListener(this);
        b6.addActionListener(this);
        b7.addActionListener(this);
        b8.addActionListener(this);
        b9.addActionListener(this);
        div.addActionListener(this);
        mul.addActionListener(this);
        add.addActionListener(this);
        sub.addActionListener(this);
        eql.addActionListener(this);
        clr.addActionListener(this);
        rem.addActionListener(this);
    }
```

```
public void actionPerformed(ActionEvent ae)
        {
                String str = ae.getActionCommand();
if(str.equals("+")||str.equals("-")||str.equals("*")||str.equals("/") || str.equals("%"))
                {
                        String str1 = t1.getText();
                        tmp=str;
                        a = Integer.parseInt(str1);
                        msg="";
                }
                else if(str.equals("="))
                {
                        String str2 = t1.getText();
                        b = Integer.parseInt(str2);
                        int sum=0;
                        if(tmp=="+")
                                sum=a+b;
                        else if(tmp=="-")
                                sum=a-b;
                        else if(tmp=="*")
                                sum=a*b;
                        else if(tmp=="/")
                                sum=a/b;
                        else if(tmp=="%")
                                sum=a%b;
                        else
                                sum=0;
                        //String s=String.valueOf(sum);  //Converts value to string
                        t1.setText(""+sum);
                        msg="";
                }
                else if(str=="C")
                {
                        t1.setText("");
                }
                else
                {
                        msg+=str;
                        t1.setText(""+msg);
                }
        }

}
```

Output:

```
Administrator: C:\Windows\system32\cmd.exe - appletviewer  CalsiDemo.java

D:\OOP\LAB\WEEK-11>javac CalsiDemo.java

D:\OOP\LAB\WEEK-11>appletviewer CalsiDemo.java
```



WEEK-12

After the completion of the practice session, the student will be able to Develop GUI applications using Swing Controls.

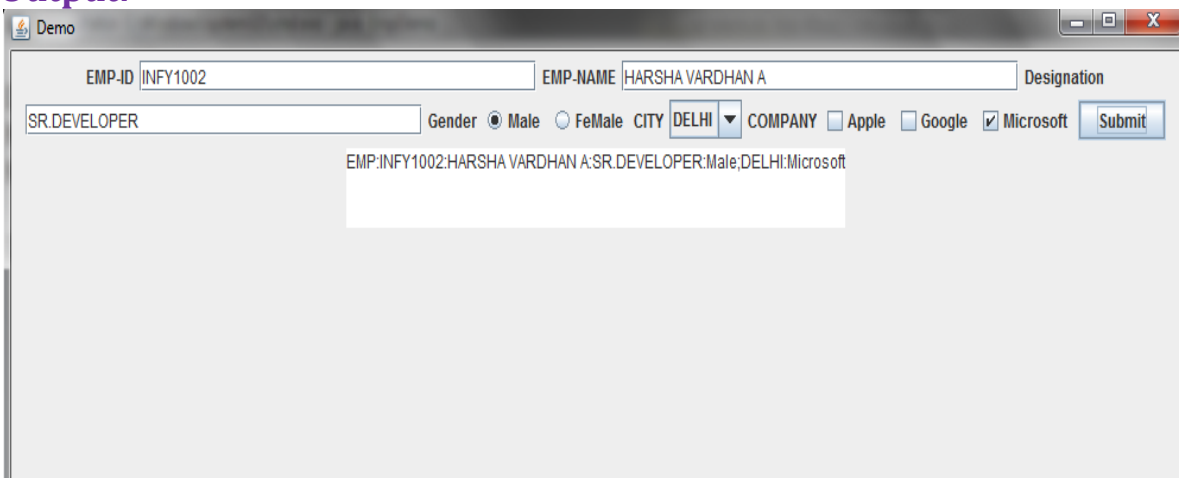
a) Create a user interface to insert employee details, Display the data in Text area.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;

public class EmpDemo  implements ActionListener
  {
            String data="EMP:" ;
            JFrame jf;
            JPanel jp;
            JLabel l1,l2,l3,l4,l5,l6;
                  JTextField t1,t2,t3;
                  JRadioButton  r1,r2;
                  ButtonGroup bg;
                  JComboBox jc;
                  JCheckBox c1,c2,c3;
          JButton b1;
                  JTextArea ta1;
            String cities[] = {"HYD", "BNGL","PUNE","DELHI"};

                  EmpDemo( )
            {
                        jf = new JFrame( ); //Top or High  Level Window
                        jf.setSize(300,300);
                        jf.setTitle("Demo");
                        jp = new JPanel();  // lower or second level window

                        l1 = new JLabel("EMP-ID");

                        l2 = new JLabel("EMP-NAME");

                        l3 = new JLabel("Designation");

                        l4 = new JLabel("Gender");
```

```
l5 = new JLabel("CITY");

l6= new JLabel("COMPANY");


t1= new JTextField(30);

t2= new JTextField(30);

t3= new JTextField(30);



r1= new JRadioButton("Male");

r2= new JRadioButton("FeMale");

bg = new ButtonGroup();

bg.add(r1);
bg.add(r2);

jc = new JComboBox(cities);


c1= new JCheckBox("Apple");

c2= new JCheckBox("Google");

c3= new JCheckBox("Microsoft");

b1 = new JButton("Submit");

ta1 = new JTextArea(30,10);

jf.setLayout(new GridLayout(5,4));

jp.add(l1);      jp.add(t1);
jp.add(l2);      jp.add(t2);
jp.add(l3);      jp.add(t3);
jp.add(l4);      jp.add(r1);jp.add(r2);
jp.add(l5); jp.add(jc);

jp.add(l6); jp.add(c1); jp.add(c2); jp.add(c3);
```

```
                    jp.add(b1);     jp.add(ta1);

                    jf.add(jp);
                     jf.setSize(1000,600);
                    jf.setVisible(true);
                    b1.addActionListener (this);


          }

        public void actionPerformed(ActionEvent e)
                {
                        data= data + t1.getText() + ":";
                        data= data + t2.getText() + ":";
                        data= data + t3.getText() + ":";

                        if(r1.isSelected())
                                data= data+ "Male" +";";
                        else
                                data= data+ "FeMale" +";";

                        String str = (String) jc.getSelectedItem();

                         data = data + str + ":";

                        if(c1.isSelected())
                                data= data + c1.getText();
                        else if(c2.isSelected())
                                data=data +c2.getText();
                        else
                                data=data+c3.getText();

                        ta1.setText(data);

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

Output:



b) Create a JTable to display various fields of Student data like RollNo, Name, Branch ,Year, Percentage etc.

```
    import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;
/* import javax.swing.*/

public class JStudent
 {

  JFrame f;
  JTable j;

  JStudent()
  {

    f = new JFrame();

    f.setTitle("JTable Example");

 String[][] data = {
      { "101", "Rajesh", "CSE","II","78.5"},
      { "102", "Harsha", "CSE","II","87.5"},
      { "103", "Varsha", "CSE","II","65.5"},
      { "104", "Kiran",  "IT","II","75.5"},
      { "105", "Karan",  "IT","II","87.5"},
    };
```

```
        // Column Names
String[] head = { "RollNo", "Name", "Department","Branch" , "Percentage"};

        // Initializing the JTable
        j = new JTable(data, head);
        j.setBounds(30, 40, 200, 300);

        // adding it to JScrollPane
        JScrollPane sp = new JScrollPane(j);
        f.add(sp);
        f.setSize(500, 200);

        f.setVisible(true);
    }

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

Output: