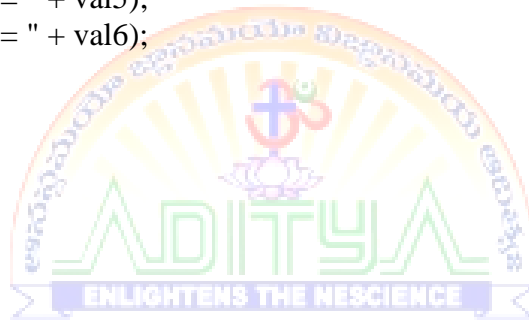


**Exercise : 1****Date:****Aim: 1) Basic Operations in Java Programming****a) Write a Java program to display default value of all primitive data type of JAVA.****Source Code:**

```
public class Demo
{
    static boolean val1;
    static double val2;
    static float val3;
    static int val4;
    static long val5;
    static String val6;
    public static void main(String[] args)
    {
        System.out.println("Default values.....");
        System.out.println("Val1 = " + val1);
        System.out.println("Val2 = " + val2);
        System.out.println("Val3 = " + val3);
        System.out.println("Val4 = " + val4);
        System.out.println("Val5 = " + val5);
        System.out.println("Val6 = " + val6);
    }
}
```

**Output:**

b) Write a Java program that display the roots of a quadratic equation  $ax^2+bx+c=0$ . Calculate the discriminant  $D$  and the basing on value of  $D$ , describe the nature of root.

**Source Code:**

```
import java.util.*;
public class Roots
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);
        // value a, b, and c
        double a , b, c;
        double root1, root2;

        System.out.println("Enter a, b and c values");
        a=sc.nextDouble();
        b=sc.nextDouble();
        c=sc.nextDouble();

        // calculate the determinant (b2 - 4ac)
        double determinant = b * b - 4 * a * c;

        // check if determinant is greater than 0
        if (determinant > 0) {

            // two real and distinct roots
            root1 = (-b + Math.sqrt(determinant)) / (2 * a);
            root2 = (-b - Math.sqrt(determinant)) / (2 * a);

            System.out.format("root1 = %.2f and root2 = %.2f", root1, root2);
        }

        // check if determinant is equal to 0
        else if (determinant == 0) {

            // roots are equal
            root1 = root2 = -b / (2 * a);
            System.out.format("root1 = root2 = %.2f;", root1);
        }

        // if determinant is less than zero
        else {

            // roots are complex number and distinct
            double real = -b / (2 * a);
            double imaginary = Math.sqrt(-determinant) / (2 * a);
            System.out.format("root1 = %.2f+%.2fi", real, imaginary);
            System.out.format("\nroot2 = %.2f-%.2fi", real, imaginary);
        }
    }
}
```

**Output:**



c) Five Bikers Compete in a race such that they drive at a constant speed which may or may not be the same as the other. To qualify the race, the speed of a racer must be more than the average speed of all 5 racers. Take as input the speed of each racer and print back the speed of qualifying racers.

**Source Code:**

```
import java.util.Scanner;
class Bike_Racers
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int racer1_Speed,racer2_Speed,racer3_Speed,racer4_Speed,racer5_Speed;
        int sum;
        float avg_Speed;

        System.out.println("Enter 5 Bike Racers Speeds");
        racer1_Speed=sc.nextInt();
        racer2_Speed=sc.nextInt();
        racer3_Speed=sc.nextInt();
        racer4_Speed=sc.nextInt();
        racer5_Speed=sc.nextInt();

        sum=racer1_Speed+racer2_Speed+racer3_Speed+racer4_Speed+racer5_Speed;
        avg_Speed=(float)sum/5;

        System.out.println("Average Speed is:"+avg_Speed);

        System.out.println("The Qualified Racers are:");
        if(racer1_Speed>avg_Speed)
            System.out.println(racer1_Speed);
        if(racer2_Speed>avg_Speed)
            System.out.println(racer2_Speed);
        if(racer3_Speed>avg_Speed)
            System.out.println(racer3_Speed);
        if(racer4_Speed>avg_Speed)
            System.out.println(racer4_Speed);
        if(racer5_Speed>avg_Speed)
            System.out.println(racer5_Speed);
    }
}
```

**Output:**

**Exercise : 2****Date:****Aim: 2) Operations, Expressions, Control-flow, Strings**

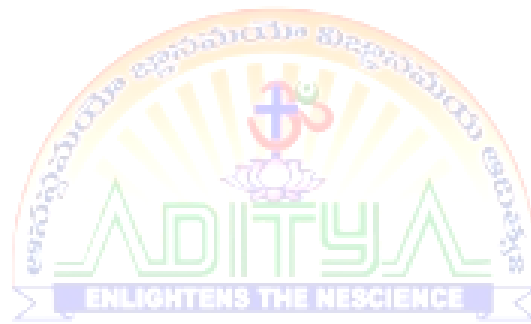
- a) Write a Java program to search for an element in a given list of elements using binary search mechanism.

**Source Code:**

```
import java.util.*;
class BinarySearchExample
{
    public static void binarySearch(int arr[], int first, int last, int key)
    {
        int mid = (first + last)/2;
        while( first <= last )
        {
            if ( arr[mid] < key )
            {
                first = mid + 1;
            }
            else if(arr[mid] == key )
            {
                System.out.println("Element is found at index: " + mid);
                break;
            }
            else
            {
                last = mid - 1;
            }
            mid = (first + last)/2;
        }
        if ( first > last )
        {
            System.out.println("Element is not found!");
        }
    }
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        int n,key,arr[];
        System.out.println("Enter the number of elements");
        n=sc.nextInt();
        arr=new int[n];
        System.out.println("Enter "+n+" elements");
        for(int i=0;i<n;i++)
            arr[i]=sc.nextInt();
        System.out.println("Enter the number to search");
        key=sc.nextInt();
        int last=n-1;
        binarySearch(arr,0,last,key);
    }
}
```

```
}
```

**Output:**



**b) Write a Java program to sort for an element in a given list of elements using bubble Sort****Source Code:**


```
import java.util.Scanner;
class BubbleSortExample
{
    public static void bubbleSort(int[] arr)
    {
        int n=arr.length,temp;
        for(int i=0;i<n-1;i++)
        {
            for(int j=0;j<n-i-1;j++)
            {
                if(arr[j]>arr[j+1])
                {
                    temp=arr[j];
                    arr[j]=arr[j+1];
                    arr[j+1]=temp;
                }
            }
        }
    }
    public static void main(String[] args)
    {
        int arr[],n;
        Scanner sc=new Scanner(System.in);
        n=sc.nextInt();
        arr=new int[n];

        for(int i=0;i<n;i++)
            arr[i]=sc.nextInt();

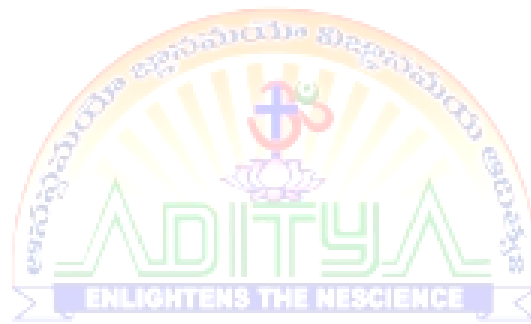
        System.out.println("Array Before Bubble Sort");
        for(int i=0; i < arr.length; i++)
        {
            System.out.print(arr[i] + " ");
        }
        System.out.println();

        bubbleSort(arr);          //sorting array elements using bubble sort

        System.out.println("Array After Bubble Sort");
        for(int i=0; i < arr.length; i++)
        {
            System.out.print(arr[i] + " ");
        }
    }
}
```

The logo of Aditya College of Engineering and Technology is a circular emblem. It features a central sun-like symbol with rays, surrounded by a green border. The word 'ADITYA' is written in large, stylized green letters across the middle. Below it, a blue banner contains the text 'ENLIGHTENS THE NESCIENCE'. The entire logo is semi-transparent and overlaid on the code.

**Output:**





c) Write a Java program to sort for an element in a given list of elements using merge Sort.

**Source Code:**

```
class Merge_Sort
{
    /* Function to merge the subarrays of a[] */
    public static void merge(int a[], int beg, int mid, int end)
    {
        int i,j,k;
        i=beg;
        j=mid+1;
        int x=0;
        int temp[]=new int[end-beg+1];
        while(i<=mid && j<=end)
        {
            if(a[i]<=a[j])
                temp[x++]=a[i++];
            else
                temp[x++]=a[j++];
        }
        while(i<=mid)
            temp[x++]=a[i++];
        while(j<=end)
            temp[x++]=a[j++];
        k=0;
        for(int s=beg;s<=end;s++)
            a[s]=temp[k++];
    }
    public static void mergeSort(int a[], int beg, int end)
    {
        if(beg < end)
        {
            int mid = (beg + end) / 2;
            mergeSort(a, beg, mid);
            mergeSort(a, mid + 1, end);
            merge(a, beg, mid, end);
        }
    }
    /* Function to print the array */
    public static void printArray(int a[], int n)
    {
        int i;
        for (i = 0; i < n; i++)
            System.out.print(a[i] + " ");
    }
}
```

```
public static void main(String args[])
{
    int a[] = { 11, 30, 24, 7, 31, 16, 39, 41 };
    int n = a.length;
    Merge_Sort m1 = new Merge_Sort();
    System.out.println("\nBefore sorting array elements are - ");
    m1.printArray(a, n);
    m1.mergeSort(a, 0, n - 1);
    System.out.println("\nAfter sorting array elements are - ");
    m1.printArray(a, n);
    System.out.println("");
}
}
```

**Output:**



d) Write a Java program using StringBuffer to delete or remove character.

**Source Code:**

```
class StringBuffer_Demo
{
    public static void main(String args[])
    {
        StringBuffer s1=new StringBuffer();
        System.out.println(s1.capacity());    // 16 => C=(S+1)*2 , 34
        System.out.println(s1.length());      // 0
        StringBuffer s2=new StringBuffer("Welcome ");
        System.out.println(s2.capacity());    // 24

        System.out.println(s2.charAt(4));    // o

        s2.setCharAt(4,'a');
        System.out.println(s2);    // Welcame

        s2.deleteCharAt(4);
        System.out.println(s2);    // Welcme

        s2.append(" Srinu");
        System.out.println(s2);    // Welcme Srinu

        s2.insert(4,"a");
        System.out.println(s2);    // Welcame Srinu

        s2.delete(8,13);
        System.out.println(s2);    // Welcame

        s2.append(true);
        System.out.println(s2);    // Welcame true

        s2.reverse();
        System.out.println(s2);    // eurt emacleW
    }
}
```

**Output:**

**Exercise : 3****Date:****Aim: 3) Class, Objects**

a) Write a Java program to implement class mechanism. Create a class, method and invoke them inside main method.

**Source Code:**

```
import java.util.*;
class Employee
{
    int empid;
    String empname, desg, Organization, ug, pg, address;
    float Sal, per_marks_ug, per_marks_pg;

    Scanner sc=new Scanner(System.in);

    public void get_PersonalInfo()
    {
        System.out.println("Enter your empid, name, desg, salary, organization name,
address");
        empid=sc.nextInt();
        sc.nextLine();
        empname=sc.nextLine();
        desg=sc.nextLine();
        Sal=sc.nextFloat();
        sc.nextLine();
        Organization=sc.nextLine();
        address=sc.nextLine();
    }
    public void get_QualificationInfo()
    {
        System.out.println("Enter your UG course, marks and PG course and
Marks");
        ug=sc.nextLine();
        per_marks_ug=sc.nextFloat();
        sc.nextLine();
        pg=sc.nextLine();
        per_marks_pg=sc.nextFloat();
    }
    public void show_PersonalInfo()
    {
        System.out.println("=====");
        System.out.println("                PERSONAL INFORMATION                ");
        System.out.println("=====");
        System.out.println("EMPID: "+empid);
        System.out.println("EMP NAME: "+empname);
        System.out.println("DESGINATION: "+desg);
        System.out.println("SALARY: "+Sal);
        System.out.println("ORGANIZATION NAME: "+Organization);
        System.out.println("ADDRESS: "+address);
    }
}
```

```
public void show_QualificationInfo()
{
    System.out.println("=====");
    System.out.println("                QUALIFICATION INFORMATION                ");
    System.out.println("=====");

    System.out.println("UG COURSE: "+ug);
    System.out.println("UG PERCENTAGE: "+per_marks_ug);
    System.out.println("PG COURSE: "+pg);
    System.out.println("PG PERCENTAGE: "+per_marks_pg);

}

public static void main(String args[])
{
    // Object Creation - memory for member variable declared inside the class
    Employee e1=new Employee();
    // reading of employee information
    e1.get_PersonalInfo();
    e1.get_QualificationInfo();
    // showing of employee information
    e1.show_PersonalInfo();
    e1.show_QualificationInfo();
}
}
```



**Output:**

**b) Write a Java program to implement Constructor.****Source Code:**

```
class Constructor
{
    private Constructor() // Default Constructor
    {
        System.out.println("Default Constructor");
        System.out.println("Welcome to Guest");
    }
    private Constructor(String name) // Parameterized Constructor
    {
        System.out.println("Parameterized Constructor");
        System.out.println("Welcome to "+name);
    }
    public static void main(String args[])
    {
        Constructor c1=new Constructor(); // called Default Constructor
        Constructor c2=new Constructor("samyuktha"); // called Parameterized Constructor
    }
}
```

**Output:**

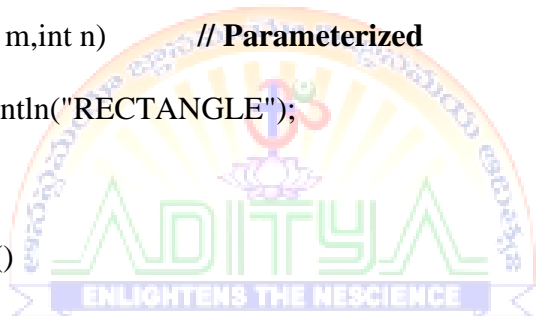
**Exercise: 4****Date:****Aim: 4) Methods****a) Write a Java program to implement constructor overloading.****Source Code:**

```
class Box_Demo
{
    int l,b,area;

    public Box_Demo()                // Default Constructor
    {
        System.out.println("Default");
        l=b=1;
    }
    public Box_Demo(int m)            // Parameterized
    {
        System.out.println("SQUARE");
        l=m;
        b=m;
    }
    public Box_Demo(int m,int n)      // Parameterized
    {
        System.out.println("RECTANGLE");
        l=m;
        b=n;
    }
    public void Cal_Area()
    {
        area=l*b;
        System.out.println("Area is: "+area);
    }
    public static void main(String args[])
    {
        Box_Demo b1=new Box_Demo(5);
        b1.Cal_Area();

        Box_Demo b2=new Box_Demo(3,4);
        b2.Cal_Area();

        Box_Demo b3=new Box_Demo();
        b3.Cal_Area();
    }
}
```


The logo of Aditya College of Engineering and Technology is a circular emblem. It features a central sun-like symbol with rays, surrounded by a decorative border. Below the emblem, the text 'ADITYA' is written in a stylized font, and a banner at the bottom reads 'ENLIGHTENS THE MESCIENCE'.**Output:**

b) Write a Java program to implement method overloading.

**Source Code:**

```
class Method_Overloading
{
    public void methodOne()
    {
        System.out.println("no argument");
    }
    public void methodOne(int x,int y)
    {
        System.out.println(x+y);
    }
    public void methodOne(int d)
    {
        System.out.println(d);
    }
    public void methodOne(double d)
    {
        System.out.println(d);
    }
    public static void main(String args[])
    {
        Method_Overloading mo=new Method_Overloading();

        mo.methodOne();
        mo.methodOne(10);
        mo.methodOne(10,20);
        mo.methodOne(3.14);
    }
}
```



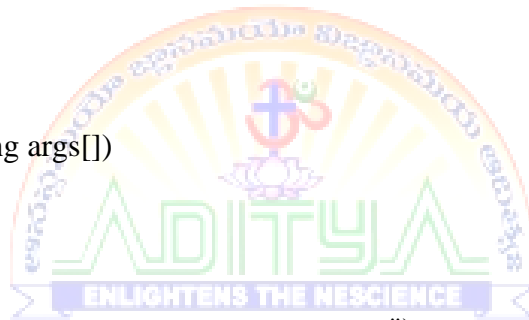
**Output:**



**Exercise: 5****Date:****Aim: 5) Inheritance****a) Write a Java program to implement Single Inheritance.****Source Code:**

```
class A
{
    int x=10;
    public void showX()
    {
        System.out.println("X = "+x);
    }
}
class B extends A
{
    int y=20;
    public void showY()
    {
        System.out.println("Y = "+y);
    }
}
class SingleLevel
{
    public static void main(String args[])
    {
        A a=new A();
        a.showX();

        System.out.println("=====");
        B b=new B();
        b.showX();
        b.showY();
    }
}
```

**Output:**

**b) Write a Java program to implement multi level Inheritance.****Source Code:**

```
import java.util.*;
class Customer
{
    String cust_id, cust_name, address;
    float balance;
    Scanner sc=new Scanner(System.in);

    public void get_CustomerInfo()
    {
        System.out.println("Enter Customer ID, Name, Balance and address");
        cust_id=sc.nextLine();
        cust_name=sc.nextLine();
        balance=sc.nextFloat();
        sc.nextLine();
        address=sc.nextLine();
    }
    public void show_CustomerInfo()
    {
        System.out.println("Customer Details are:");
        System.out.println("Id: "+cust_id);
        System.out.println("Name: "+cust_name);
        System.out.println("Balance: "+balance);
        System.out.println("Address: "+address);
    }
}
class Transaction extends Customer
{
    public void deposit(float amt)
    {
        System.out.println("Amount Deposited: "+amt);
        balance=balance+amt;
    }
    public void withdraw(float amt)
    {
        System.out.println("Amount withdrawn: "+amt);
        balance=balance-amt;
    }
    public void show_Bal()
    {
        System.out.println("Available Balance: "+balance);
    }
}
class Bank extends Transaction
```

```
{  
    static String bankname="Canara Bank",ifsc="CBN0003268";  
  
    public void show_BankInfo()  
    {  
        System.out.println("Bank Name: "+Bank.bankname);  
        System.out.println("IFSC Code: "+Bank.ifsc);  
    }  
  
    public static void main(String args[])  
    {  
        Bank c1=new Bank();  
  
        c1.get_CustomerInfo();  
        c1.show_CustomerInfo();  
        c1.show_BankInfo();  
  
        c1.deposit(10000);  
        c1.show_Bal();  
  
        c1.withdraw(5000);  
        c1.show_Bal();  
    }  
}
```



**Output:**

c) Write a Java program for abstract class to find areas of different shapes.

**Source Code:**

```
import java.util.*;
abstract class Shape
{
    Scanner sc=new Scanner(System.in);
    float s1,s2,a;
    final float pi=3.14f;
    public abstract void get_Input();
    public abstract void Cal_Area();
    public void show_Area()
    {
        System.out.println("Area is:"+a);
    }
}
class Rect extends Shape
{
    public void get_Input()
    {
        System.out.println("Enter L and B values");
        s1=sc.nextFloat();
        s2=sc.nextFloat();
    }
    public void Cal_Area()
    {
        a=s1*s2;
    }
}
class Circle extends Shape
{
    public void get_Input()
    {
        System.out.println("Enter radius of the Circle");
        s1=sc.nextFloat();
    }
    public void Cal_Area()
    {
        a=pi*s1*s1;
    }
}
```

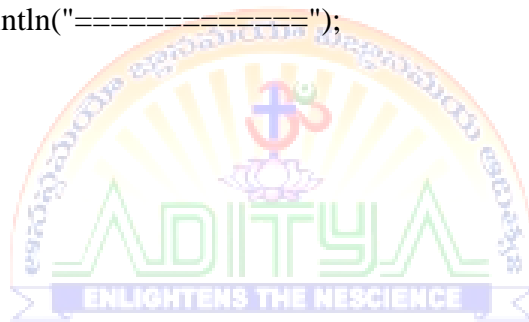


```
class Mainclass
{
    public static void main(String args[])throws Exception
    {
        Shape s;

        s=new Rect();
        System.out.println("Rectangle:");
        s.get_Input();
        s.Cal_Area();
        s.show_Area();
        System.out.println("=====");

        s=new Circle();
        System.out.println("Circle:");
        s.get_Input();
        s.Cal_Area();
        s.show_Area();
        System.out.println("=====");
    }
}
```

**Output:**



**Exercise : 6****Date:****Aim: 6) Inheritance - continued****a) Write a Java program give example for “super” keyword.****Source Code:**

```
class A
{
    int x=10;
    public void show()
    {
        System.out.println("A: X = "+x);
    }
}
class B extends A
{
    int x=20;
    public void show()
    {
        super.show();
        System.out.println("B: X = "+x);
    }
}
class C extends B
{
    int x=30;
    public void show()
    {
        super.show();
        System.out.println("C: X = "+x);
    }
}
class MainDemo
{
    public static void main(String args[])
    {
        C c=new C();
        c.show();
    }
}
```

**Output:**

b) Write a Java program to implement interface. What kind of Inheritance can be achieved?

**Source Code:**

```
import java.util.*;
interface Internal
{
    void get_InternalMarks();
}
interface External
{
    void get_ExternalMarks();
}
interface Marks extends Internal, External
{
    void show_Marks();
}
class Result implements Marks
{
    // s1_i -> sub1 internal, s1_e -> sub1 external
    float s1_i, s1_e, s2_i, s2_e, s3_i, s3_e;
    Scanner sc = new Scanner(System.in);
    public void get_InternalMarks()
    {
        System.out.println("Enter 3 subjects internal marks (0 - 40)");
        s1_i = sc.nextFloat();
        s2_i = sc.nextFloat();
        s3_i = sc.nextFloat();
    }
    public void get_ExternalMarks()
    {
        System.out.println("Enter 3 subjects External marks (0 - 60)");
        s1_e = sc.nextFloat();
        s2_e = sc.nextFloat();
        s3_e = sc.nextFloat();
    }
    public void show_Marks()
    {
        System.out.println("Subject \t Internal \t External \t Total_Marks: ");
        System.out.println(" Sub1 \t\t " + s1_i + "\t\t" + s1_e + "\t\t" + (s1_i + s1_e));
        System.out.println(" Sub2 \t\t " + s2_i + "\t\t" + s2_e + "\t\t" + (s2_i + s2_e));
        System.out.println(" Sub3 \t\t " + s3_i + "\t\t" + s3_e + "\t\t" + (s3_i + s3_e));
    }
}

class Mainclass
{

```

```
public static void main(String args[])
{
    Marks m=new Result();
    m.get_InternalMarks();
    m.get_ExternalMarks();
    m.show_Marks();
}
}
```

**Output:**





**Exercise : 7****Date:****Aim: 7) (Exceptions)****a) Write a Java program that describes exception handling mechanism.****Source Code:**

```
import java.util.Scanner;
class Exception_handle
{
    public static void main(String args[])
    {
        Exception ex=new Exception();
    }
}
class Exception
{
    Exception()
    {
        Scanner s=new Scanner(System.in);
        System.out.print("Enter a and b values:");
        int a=s.nextInt();
        int b=s.nextInt();
        try
        {
            int c=a/b;
            System.out.print(c);
        }
        catch(ArithmeticException ae)
        {
            System.out.println("Arthmatic Exception occured");
        }
    }
}
```

**Output:**

**b) Write a Java program Illustrating Multiple catch clauses.****Source Code:**

```
class ExceptionDemo
{
    public static void main(String[] args)
    {
        int m, n, o=0;
        try
        {
            m = Integer.parseInt(args[0]);
            n = Integer.parseInt(args[1]);
            o = m/n;
        }
        catch(ArrayIndexOutOfBoundsException ae)
        {
            System.out.println(ae.getMessage());
        }
        catch(NumberFormatException ne)
        {
            System.out.println(ne.getMessage());
        }
        catch(ArithmeticException are)
        {
            are.printStackTrace();
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
        finally
        {
            System.out.println("Cleanup code");
            System.out.println(o);
        }
    }
}
```

**Output:**

**Exercise : 8****Date:****Aim: 8) Runtime Polymorphism****a) Write a Java program that implements Runtime polymorphism.****Source Code:**

```
class Notification
{
    String message;
    Notification(String message) {
        this.message = message;
    }
    void send() {
        System.out.println(message);
    }
}

class EmailNotification extends Notification {
    String emailaddress;
    EmailNotification(String message, String emailaddress) {
        super(message); // Call the constructor of the superclass
        this.emailaddress = emailaddress;
    }
    void send()
    {
        System.out.println(message + ": an email notification from " + emailaddress);
    }
}

class RuntimePolymorphism {
    public static void main(String[] args) {
        Notification n = new Notification("This is a normal message");
        EmailNotification e = new EmailNotification("Important email", "sravya@acet.ac.in");
        n.send();
        e.send();
    }
}
```

**Output:**

**b) Write a Case study on run time polymorphism, inheritance that implements in above problem.**

**Case Study:**

**Polymorphism:-**

Polymorphism is the process of performing the single task in different ways. Polymorphism is the Greek word . In which 'poly' means many and 'morphs' means forms. It means many forms .

**Polymorphism is classified into two types they are-**

1. Static or Compile time Polymorphism
2. Dynamic or Run time Polymorphism

**Runtime Polymorphism:-**

Runtime Polymorphism is also called as Dynamic polymorphism. It is the process of calling the Overriding methods in runtime rather than In compile time.

In this process the overriding methods are called with reference of super class.

**Explanation:-**

In the above program we use single inheritance two achieve the runtime polymorphism in our example.

We use two classes here. We call the base class with the reference of the super class.

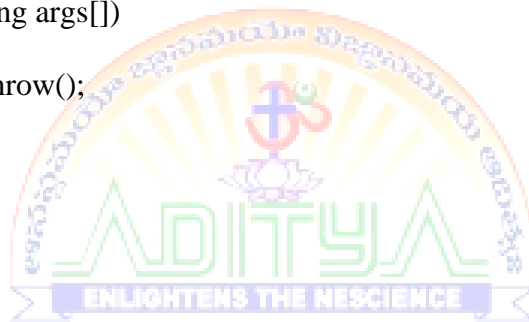
We create different objects to the two classes and call the both base class as well as super class with reference of super class only.

So, Runtime polymorphism is achieved here.



**Exercise : 9****Date:****Aim: 9) User defined Exception****a) Write a Java program for creation of Illustrating throw.****Source Code:**

```
public class TestThrow
{
    void validate()
    {
        try
        {
            System.out.println("Inside of the try block");
            throw new ArithmeticException("Not valid,throwing Exception");
        }
        catch(ArithmeticException ae)
        {
            System.out.println(ae);
        }
    }
    public static void main(String args[])
    {
        TestThrow t=new TestThrow();
        t.validate();
    }
}
```

**Output:**

b) Write a Java program for creation of Illustrating finally.

**Source Code:**

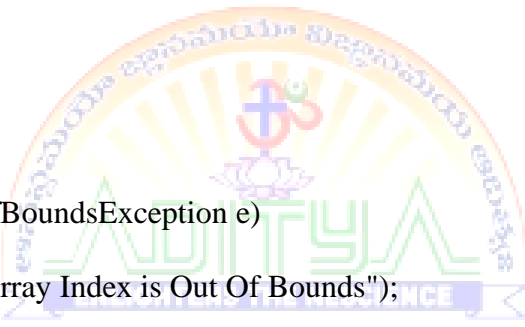
```
class TestFinallyBlock
{
    public static void main(String args[])
    {
        try
        {
            int data=25/5;
            System.out.println(data);
        }
        catch(NullPointerException e)
        {
            System.out.println(e);
        }
        finally
        {
            System.out.println("finally block is defaultly  executed");
        }
        System.out.println("rest of the code...");
    }
}
```

**Output:**



**c) Write a Java program for creation of Java Built-in Exception.****Source Code:**

```
import java.util.Scanner;
class Demo
{
    public static void main(String args[])
    {
        ArrayIndex_Demo obj=new ArrayIndex_Demo();
        obj.display();
    }
}
class ArrayIndex_Demo
{
    void display()
    {
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter the element you want to insert:");
        int n=sc.nextInt();
        System.out.print("Enter the index of the array:");
        try
        {
            int a[] = new int[5];
            int i=sc.nextInt();
            a[i]=n;
        }
        catch (ArrayIndexOutOfBoundsException e)
        {
            System.out.println("Array Index is Out Of Bounds");
        }
    }
}
```

**Output:**

**d) Write a Java program for creation of User Defined Exception.**

**Problem Statement:** Write a Program to take care of NumberFormatException if user enters values other than integer for calculating average marks of a student. The name of the student and marks in 3 subjects are taken from user while executing the program. In the same program write your own Exception classes to take care of **Negative values** and values out of range (i.e other than in the range of 0 -100).

**Source Code:**

```
class NegativeValException extends Exception
{
    public NegativeValException(String msg)
    {
        super(msg);
    }
}
class Excep_Example
{
    public static void main(String args[])
    {
        String name=null;
        int m1=0,m2=0,m3=0;
        try
        {
            name=args[0];
            m1=Integer.parseInt(args[1]);
            m2=Integer.parseInt(args[2]);
            m3=Integer.parseInt(args[3]);
            if(m1<0 || m2<0 || m3<0)
                throw new NegativeValException("Marks should be greater than 0");
        }
        catch(ArrayIndexOutOfBoundsException aoe)
        {
            System.out.println("Minimum of 4 arguments you need to pass");
        }
        catch(NumberFormatException ne)
        {
            System.out.println("Marks should be Integers only");
        }
        catch(NegativeValException nve)
        {
            System.out.println("Marks should be greater than zero");
            System.exit(0);
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
        System.out.println("Name = "+name);
        System.out.println("Average Marks="+ (m1+m2+m3)/3);
    }
}
```



**Output:**



**Exercise : 10****Date:****Aim: 10) Threads**

- a) that creates threads by extending Thread class. First thread display “Good Morning” every 1 sec, the second thread displays “Hello” every 2 seconds and the third display “Welcome” every 3 seconds, (Repeat the same by implementing Runnable).

**Source Code:**

```
class MyThread1 extends Thread
```

```
{
    public void run()
    { try{
        while(true)
        {
            System.out.println(Thread.currentThread().getName()+": Good Morning");
            Thread.sleep(1000);
        }
    }
    catch(InterruptedException ie)
    {
    }
}
```

```
class MyThread2 extends Thread
```

```
{
    public void run()
    {
        try{
            while(true)
            {
                System.out.println(Thread.currentThread().getName()+": Hello");
                Thread.sleep(2000);
            }
        }
        catch(InterruptedException ie)
        {
        }
    }
}
```

```
class MyThread3 extends Thread
```

```
{
    public void run()
    {
        try{
            while(true)
```



```
        {
            System.out.println(Thread.currentThread().getName()+": Welcome");
            Thread.sleep(3000);
        }
    }
    catch(InterruptedException ie)
    {
    }
}
class MainDemo
{
    public static void main(String args[])
    {
        MyThread1 t1=new MyThread1();
        MyThread2 t2=new MyThread2();
        MyThread3 t3=new MyThread3();
        t1.setName("A");
        t2.setName("B");
        t3.setName("C");

        t1.start();
        t2.start();
        t3.start();

        System.out.println("MAIN CLOSED");
    }
}
```

**Output:**



**b) Write a Java program illustrating isAlive and join().**

**Source Code:**


```
public class AliveJoin extends Thread
{
    public void run()
    {
        System.out.println("ABDUL KALAM ");
        try
        {
            Thread.sleep(300);
        }
        catch (InterruptedException ie)
        { }
        System.out.println("SUBHASH CHANDRABOSH ");
    }
    public static void main(String[] args)
    {
        AliveJoin c1 = new AliveJoin();
        AliveJoin c2 = new AliveJoin();
        c1.start();
        c2.start();
        System.out.println(c1.isAlive());
        System.out.println(c2.isAlive());
        try
        {
            c1.join(); // Waiting for c1 to finish
        }
        catch (InterruptedException ie)
        { }
    }
}
```



**Output:**

**c) Write a Java program illustrating Daemon Threads.****Source Code:**

```
import java.io.*;
public class DaemonThread extends Thread
{
    public DaemonThread(String name)
    {
        super(name);
    }
    public void run()
    {
        if(Thread.currentThread().isDaemon())
        {
            System.out.println(getName() + " is Daemon thread");
        }
        else
        {
            System.out.println(getName() + " is User thread");
        }
    }
    public static void main(String[] args)
    {
        DaemonThread t1 = new DaemonThread("t1");
        DaemonThread t2 = new DaemonThread("t2");
        DaemonThread t3 = new DaemonThread("t3");
        t1.setDaemon(true);
        t1.start();
        t2.start();
        t3.setDaemon(true);
        t3.start();
    }
}
```

The logo of Aditya College of Engineering and Technology is a circular emblem. It features a central sun-like symbol with rays, surrounded by the college's name in Kannada and English. Below the emblem, a banner reads "ENLIGHTENS THE NESCIENCE".**Output:**

**Exercise : 11****Date:****Aim: 11) Thread continuity****a) Write a Java program Producer Consumer problem.****Source Code:**

```
import java.util.LinkedList;
public class Threadexample
{
    public static void main(String[] args) throws InterruptedException
    {
        final PC pc = new PC();
        Thread t1 = new Thread(new Runnable()
        {
            @Override
            public void run()
            {
                try
                {
                    pc.produce();
                }
                catch(InterruptedException e)
                {
                    e.printStackTrace();
                }
            }
        });
        Thread t2 = new Thread(new Runnable()
        {
            @Override
            public void run()
            {
                try
                {
                    pc.consume();
                }
                catch(InterruptedException e)
                {
                    e.printStackTrace();
                }
            }
        });
        t1.start();
        t2.start();
        t1.join();
        t2.join();
    }
    public static class PC
    {
        LinkedList<Integer> list = new LinkedList<>();
        int capacity = 4;
```



```
public void produce() throws InterruptedException
{
    int value = 0;
    while (true)
    {
        synchronized (this)
        {
            while (list.size()==capacity)
                wait();
            System.out.println("Producer produced-"+ value);
            list.add(value++);
            notify();
            Thread.sleep(1000);
        }
    }
}

public void consume() throws InterruptedException
{
    while (true)
    {
        synchronized (this)
        {
            while (list.size()==0)
                wait();
            int val = list.removeFirst();
            System.out.println("Consumer consumed-"+ val);
            notify();
            Thread.sleep(1000);
        }
    }
}
}
```



**Output:**

**b) Write a case study on thread Synchronization after solving the above producer consumer problem.**

**Case study:**

In computing, the producer-consumer problem (also known as the bounded-buffer problem) is a classic example of a multi-process synchronization problem. The problem describes two processes, the producer and the consumer, which share a common, fixed-size buffer used as a queue.

- The producer's job is to generate data, put it into the buffer, and start again.
- At the same time, the consumer is consuming the data (i.e. removing it from the buffer), one piece at a time.

**Problem**

To make sure that the producer won't try to add data into the buffer if it's full and that the consumer won't try to remove data from an empty buffer.

**Solution**

The producer is to either go to sleep or discard data if the buffer is full. The next time the consumer removes an item from the buffer, it notifies the producer, who starts to fill the buffer again. In the same way, the consumer can go to sleep if it finds the buffer to be empty. The next time the producer puts data into the buffer, it wakes up the sleeping consumer.

**Implementation of Producer Consumer Class**

- A LinkedList list – to store list of jobs in queue.
- A Variable Capacity – to check for if the list is full or not
- A mechanism to control the insertion and extraction from this list so that we do not insert into list if it is full or remove from it if it is empty.

In PC class (A class that has both produce and consume methods), a linked list of jobs and a capacity of the list is added to check that producer does not produce if the list is full.

In Producer class, the value is initialized as 0. Also, we have an infinite outer loop to insert values in the list. Inside this loop, we have a synchronized block so that only a producer or a consumer thread runs at a time. An inner loop is there before adding the jobs to list that checks if the job list is full, the producer thread gives up the intrinsic lock on PC and goes on the waiting state. If the list is empty, the control passes to below the loop and it adds a value in the list.

In the Consumer class, we again have an infinite loop to extract a value from the list. Inside, we also have an inner loop which checks if the list is empty. If it is empty then we make the consumer thread give up the lock on PC and passes the control to producer thread for producing more jobs. If the list is not empty, we go round the loop and removes an item from the list.

In both the methods, we use notify at the end of all statements. The reason is simple, once you have something in list, you can have the consumer thread consume it, or if you have consumed something, you can have the producer produce something. sleep() at the end of both methods just make the output of program run in step wise manner and not display everything all at once so that you can see what actually is happening in the program.



**Exercise: 12****Date:****Aim: 12) (Packages)****a) Write a Java program to illustrate class Path.****Source Code:**

```
import java.net.URL;

import java.net.URLClassLoader;
public class App
{
    public static void main(String[] args)
    {
        ClassLoader sysClassLoader = ClassLoader.getSystemClassLoader();
        URL[] urls = ((URLClassLoader)sysClassLoader).getURLs();
        for(int i=0; i< urls.length; i++)
        {
            System.out.println(urls[i].getFile());
        }
    }
}
```

**Output:**

**b) Write a case study on including in class path in your os environment of your package****Source Code:****COPEING THE PATH OF JAVA:-**

First of all go to MYCOMPUTER and go to the drive where the java is installed. In that go to

PROGRAM FILES and then double click on java folder.

In that we have observed there is a folder with name java jdk, double click on the java jdk folder and then go into the bin folder.

At this time we have to copy the path of the bin folder.

**SETTING THE JAVA PATH:-**

In order to set the path of the java in our system, first of all we need to open CONTROL PANEL in our system and go to SYSTEM SETTINGS .

In system settings we need to go into the ADVANCED SYSTEM SETTINGS settings. In

advanced system settings we just click on ENVIRONMENT VARIABLES option.

In USER VARIABLES click on NEW button and type the " path" at VARIABLE NAME.

We need to paste the previously copied path of the bin folder at the place of VARIABLE VALUE.

Finally click on OK and then OK, then close the MYCOMPUTER window. Now the java path is set. We are ready to use java facilities in our computer.



c) Write a Java programs that import and use the defined your package in the previous problem.

### User defined packages

User-defined packages are those which are developed by users in order to group related classes, interfaces and sub packages. With the help of an example program, let's see how to create packages, compile Java programs inside the packages and execute them.

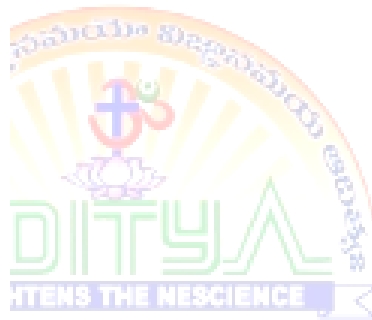
#### Steps involved in user defined package creation:

1. Creation of user defined package file
2. Compilation of user defined package file
3. Setting of class path
4. Importing of user defined package in another application.

#### Source Code:

##### Step – 1:

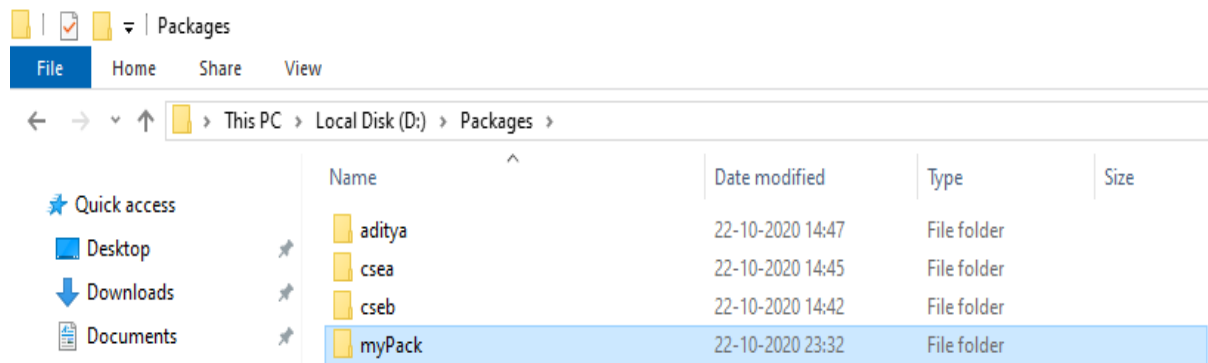
```
package myPack;
public class Compare
{
    public int getMax(int n, int m)
    {
        if(n>m)
            return n;
        else
            return m;
    }
    public int getMin(int n,int m)
    {
        if(n<m)
            return n;
        else
            return m;
    }
    public void getEqual(int n,int m)
    {
        if(n==m)
            System.out.println("Equal");
        else
            System.out.println("Not Equal");
    }
}
```



##### Step – 2:

```
D:\Java_Programs>javac -d D:\Packages Compare.java
```

```
D:\Java_Programs>
```

**Step – 3 & 4:**

```

1 import java.util.Scanner;    // built in package
2 import myPack.Compare;      // user defined package
3 class NumberDemo
4 {
5     public static void main(String args[])
6     {
7         int n1,n2;
8         Scanner sc=new Scanner(System.in);
9         Compare c=new Compare();
10        System.out.println("Enter any two numbers");
11        n1=sc.nextInt();
12        n2=sc.nextInt();
13
14        int max=c.getMax(n1,n2);
15        int min=c.getMin(n1,n2);
16        c.getEqual(n1,n2);
17
18        System.out.println("Maximu: "+max);
19        System.out.println("Minimum: "+min);
20    }
21 }

```

**Output:**

**Exercise: 13****Date:****Aim: 11) Applet****a) Write a Java program to paint like a paint brush in applet.****Source Code:**

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class MouseDrag extends Applet implements MouseMotionListener
{
    public void init(){
        addMouseMotionListener(this);
        setBackground(Color.red);
    }
    public void mouseDragged(MouseEvent me) {
        Graphics g=getGraphics();
        g.setColor(Color.white);
        g.fillOval(me.getX(),me.getY(),10,10); // (x-position, y-postion, width, height)
    }
    public void mouseMoved(MouseEvent me)
    {}
}
/*
<applet code="MouseDrag.class" height=300 width=400>
</applet>
*/
```

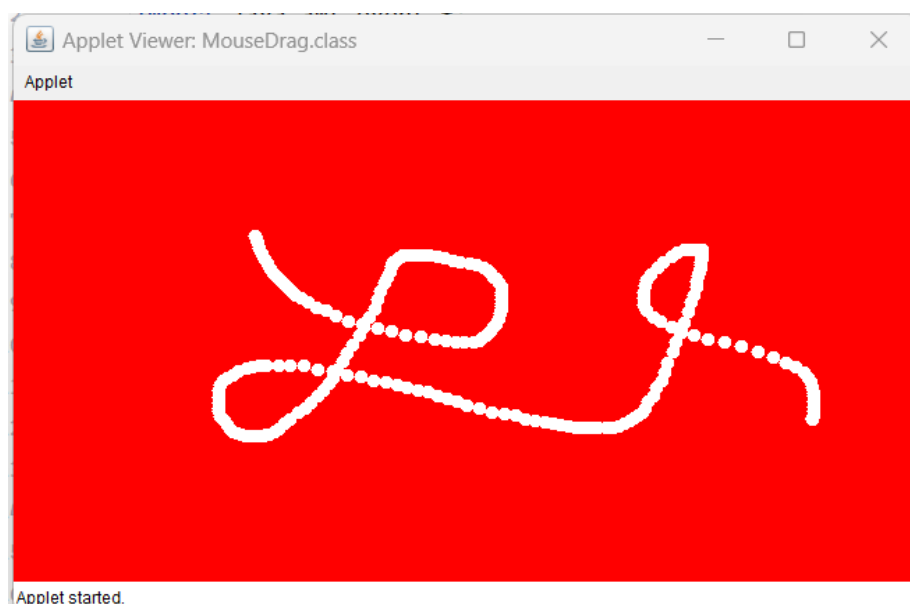
**Output:**

Running the Applet using **appletviewer** as most of the browsers stopped applet support in recent versions.

**Commands:**

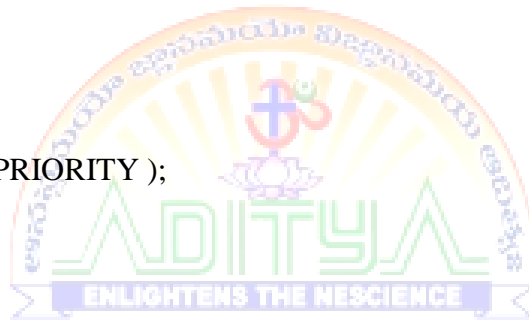
To Compile: **javac** MouseDrag.java

To View: **appletviewer** MouseDrag.java



**b) Write a Java program to display analog clock using Applet.****Source Code:**

```
import java.applet.*;
import java.awt.*;
import java.util.*;
import java.text.*;
public class MyClock extends Applet implements Runnable
{
    int width, height;
    Thread t = null;
    boolean threadSuspended;
    int hours=0, minutes=0, seconds=0;
    String timeString = "";
    public void init()
    {
        width = getSize().width;
        height = getSize().height;
        setBackground( Color.black );
    }
    public void start()
    {
        if ( t == null )
        {
            t = new Thread( this );
            t.setPriority( Thread.MIN_PRIORITY );
            threadSuspended = false;
            t.start();
        }
        else
        {
            if ( threadSuspended )
            {
                threadSuspended = false;
                synchronized( this )
                {
                    notify();
                }
            }
        }
    }
    public void stop()
    {
        threadSuspended = true;
    }
    public void run()
    {
        try
        {
            while (true)
            {
```



```

Calendar cal = Calendar.getInstance();
hours = cal.get( Calendar.HOUR_OF_DAY );
if ( hours > 12 ) hours -= 12;
minutes = cal.get( Calendar.MINUTE );
seconds = cal.get( Calendar.SECOND );

SimpleDateFormat formatter = new SimpleDateFormat( "hh:mm:ss", Locale.getDefault()
);

Date date = cal.getTime();
timeString = formatter.format( date );

// Now the thread checks to see if it should suspend itself
if ( threadSuspended )
{
    synchronized( this )
    {
        while ( threadSuspended )
        {
            wait();
        }
    }
}
repaint();
t.sleep( 1000 ); // interval specified in milliseconds
}
catch (Exception e)
{
}

void drawHand( double angle, int radius, Graphics g )
{
    angle -= 0.5 * Math.PI;
    int x = (int)( radius*Math.cos(angle) );
    int y = (int)( radius*Math.sin(angle) );
    g.drawLine( width/2, height/2, width/2 + x, height/2 + y );
}

void drawWedge( double angle, int radius, Graphics g )
{
    angle -= 0.5 * Math.PI;
    int x = (int)( radius*Math.cos(angle) );
    int y = (int)( radius*Math.sin(angle) );
    angle += 2*Math.PI/3;
    int x2 = (int)( 5*Math.cos(angle) );
    int y2 = (int)( 5*Math.sin(angle) );
    angle += 2*Math.PI/3;
    int x3 = (int)( 5*Math.cos(angle) );
    int y3 = (int)( 5*Math.sin(angle) );
    g.drawLine( width/2+x2, height/2+y2, width/2 + x, height/2 + y );
    g.drawLine( width/2+x3, height/2+y3, width/2 + x, height/2 + y );
    g.drawLine( width/2+x2, height/2+y2, width/2 + x3, height/2 + y3 );
}

```



```
}  
public void paint( Graphics g )  
{  
    g.setColor( Color.white );  
    drawWedge( 2*Math.PI * hours / 12, width/5, g );  
    drawWedge( 2*Math.PI * minutes / 60, width/3, g );  
    drawHand( 2*Math.PI * seconds / 60, width/2, g );  
    g.setColor( Color.white );  
    g.drawString( timeString, 10, height-10 );  
}  
}
```

Applet Code: Applet Code save as .html file.

```
/*  
<applet code="MyClock.class" height=300 width=400>  
</applet>  
*/
```

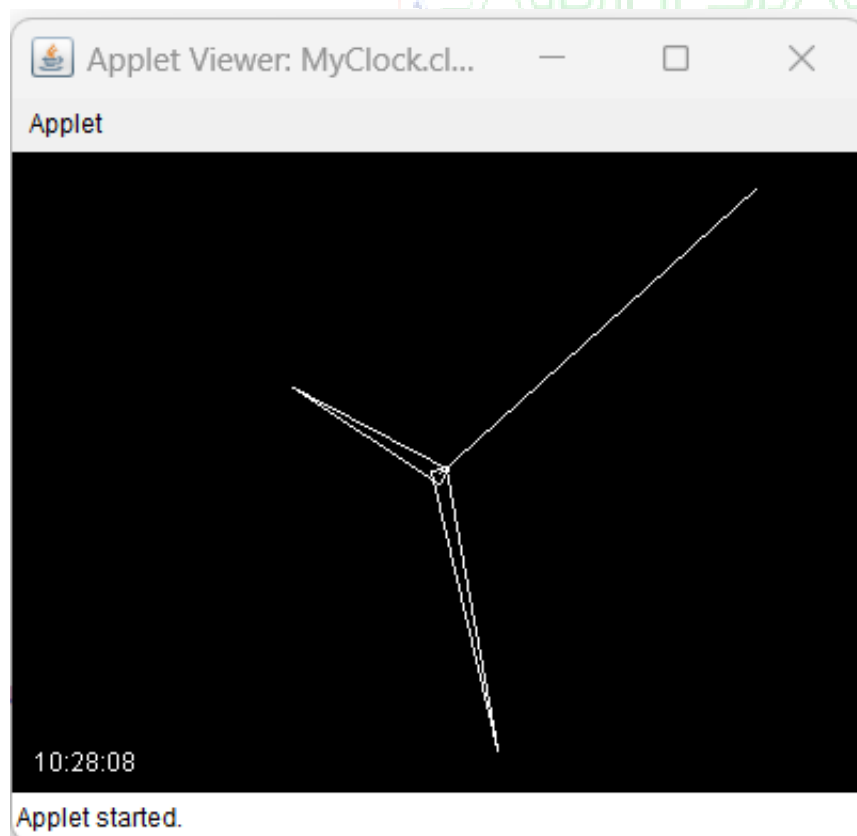
### Output:

Running the Applet using **appletviewer** as most of the browsers stopped applet support in recent versions.

### Commands:

To Compile: **javac** MyClock.java

To View: **appletviewer** MyClock.java





c) Write a JAVA program to create different shapes and fill colours using Applet.

**Source Code:**

```
import java.applet.*;
import java.awt.*;
public class ShapeColor extends Applet {
    int x=300,y=100,r=50;
    public void paint(Graphics g) {
        g.setColor(Color.red); //Drawing line color is red
        g.drawLine(3,300,200,10);
        g.setColor(Color.magenta);
        g.drawString("Line",100,100);
        g.drawOval(x-r,y-r,100,100);
        g.setColor(Color.yellow); //Fill the yellow color in circle
        g.fillOval( x-r,y-r, 100, 100 );
        g.setColor(Color.magenta);
        g.drawString("Circle",275,100);
        g.drawRect(400,50,200,100);
        g.setColor(Color.yellow); //Fill the yellow color in rectangel
        g.fillRect( 400, 50, 200, 100 );
        g.setColor(Color.magenta);
        g.drawString("Rectangel",450,100);
    }
}

/*
<applet code="ShapeColor.class" height=300 width=400>
</applet>
*/
```

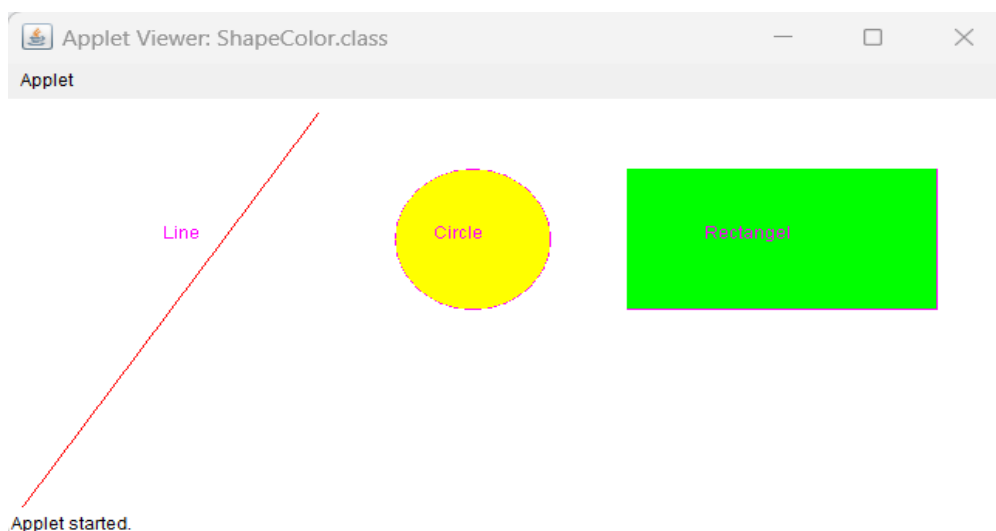
**Output:**

Running the Applet using **appletviewer** as most of the browsers stopped applet support in recent versions.

**Commands:**

To Compile: **javac ShapeColor.java**

To View: **appletviewer ShapeColor.java**



**Exercise: 14****Date:****Aim: 14) Event Handling**

- a) Write a Java program that display the x and y position of the cursor movement using mouse.

**Source Code:**

```
import java.awt.*;

import java.awt.event.*;
import java.applet.Applet;
public class AppletMouse extends Applet implements MouseListener, MouseMotionListener
{
    int x, y;
    String str="";
    public void init()
    {
        addMouseListener(this);
        addMouseMotionListener(this);
    }
    // override ML 5 abstract methods
    public void mousePressed(MouseEvent e)
    {
        x = e.getX();
        y = e.getY();
        str = "Mouse Pressed";
        repaint();
    }
    public void mouseReleased(MouseEvent e)
    {
        x = e.getX();
        y = e.getY();
        str = "Mouse Released";
        repaint();
    }
    public void mouseClicked(MouseEvent e)
    {
        x = e.getX();
        y = e.getY();
        str = "Mouse Clicked";
        repaint();
    }
    public void mouseEntered(MouseEvent e)
    {
        x = e.getX();
        y = e.getY();
        str = "Mouse Entered";
        repaint();
    }
    public void mouseExited(MouseEvent e)
    {
        x = e.getX();
```

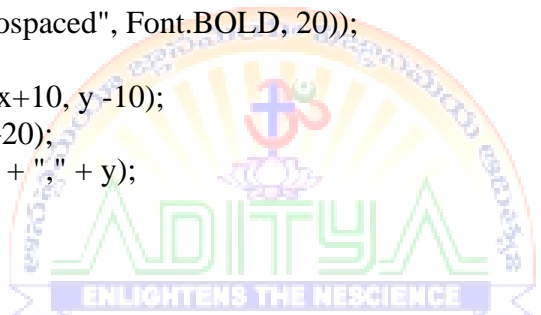


```
y = e.getY();
str = "Mouse Exited";
repaint();
}

// override two abstract methods of MouseMotionListener
public void mouseMoved(MouseEvent e)
{
    x = e.getX();
    y = e.getY();
    str = "Mouse Moved";
    repaint();
}
public void mouseDragged(MouseEvent e)
{
    x = e.getX();
    y = e.getY();
    str = "Mouse dragged";
    repaint();
}

// called by repaint() method
public void paint(Graphics g)
{
    g.setFont(new Font("Monospaced", Font.BOLD, 20));
    g.fillOval(x, y, 10, 10);
    g.drawString(x + "," + y, x+10, y -10);
    g.drawString(str, x+10, y+20);
    showStatus(str + " at " + x + "," + y);
}
}

/*
<applet code="AppletMouse.class" width="300" height="300">
</applet>
*/
```

The logo of Aditya College of Engineering and Technology is a circular emblem. It features a central sun-like symbol with rays, surrounded by a green border. The word 'ADITYA' is written in large, stylized green letters across the middle. Below it, a blue banner contains the text 'ENLIGHTENS THE NESCIENCE' in white capital letters. The entire logo is semi-transparent and overlaid on the code.

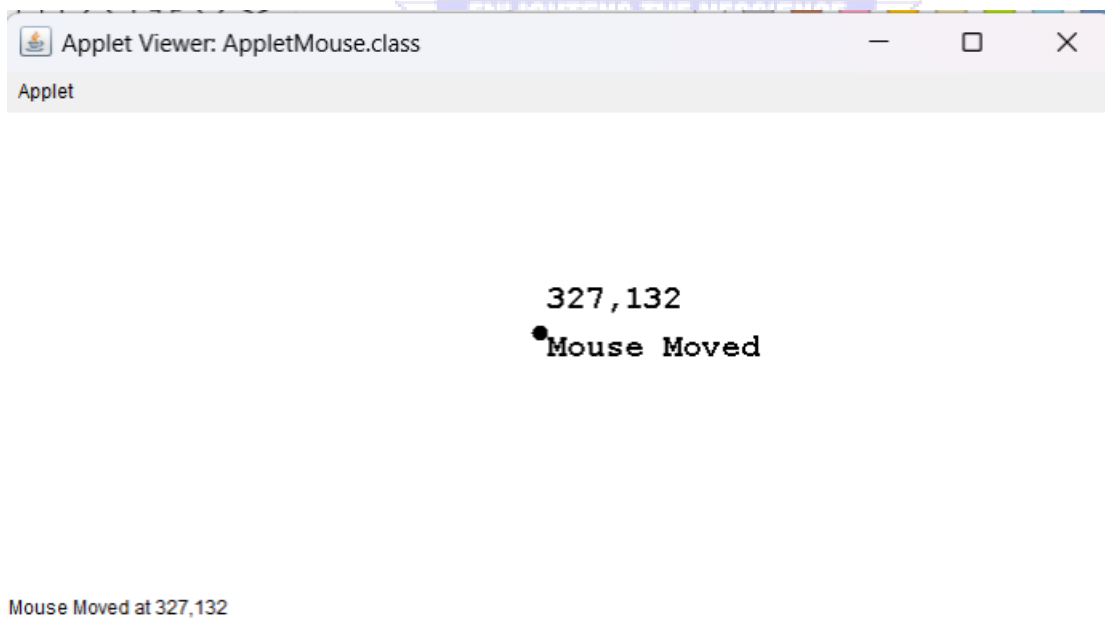
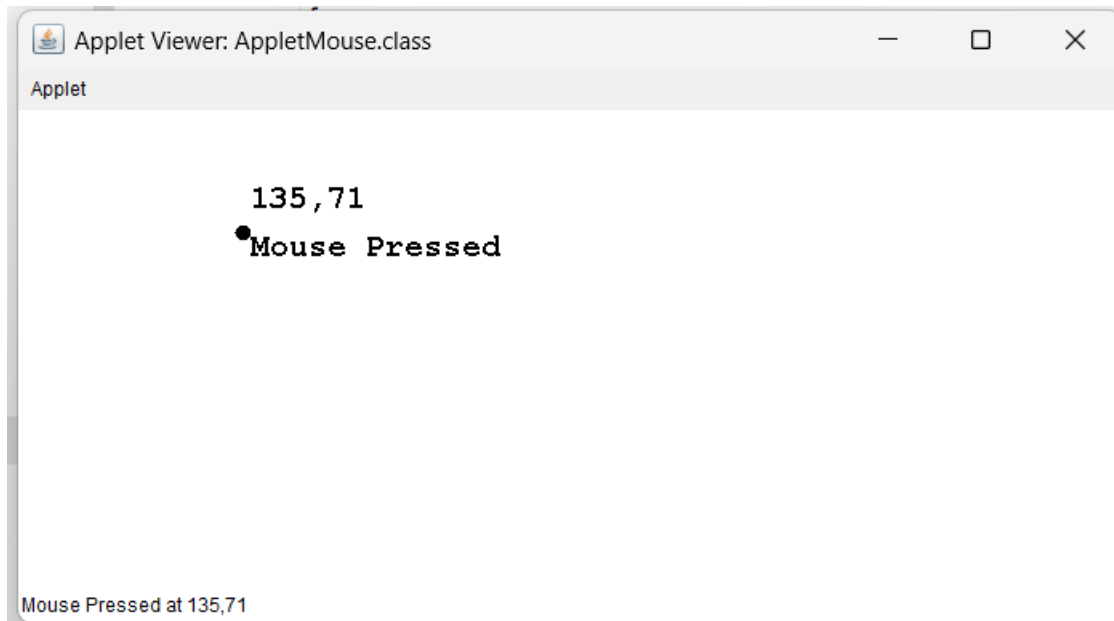
**Output:**

Running the Applet using **appletviewer** as most of the browsers stopped applet support in recent versions.

**Commands:**

To Compile: **javac** AppletMouse.java

To View: **appletviewer** AppletMouse.java



**b) Write a Java program that identifies Key-up key-down event user entering text in an Applet**

**Source Code:**

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

public class KeyUpDown1 extends Applet {
    private Font f;
    private String letter;
    private boolean first;

    public void init()
    {
        f = new Font( "Courier", Font.BOLD, 72 );
        first = true;
    }

    public void paint( Graphics g )
    {
        g.setFont( f );

        if ( !first )
            g.drawString( letter, 75, 70 );
    }

    public boolean keyDown( Event e, int key )
    {
        showStatus( "keyDown: the " + ( char ) key +
            " was pressed." );

        letter = String.valueOf( ( char ) key );
        first = false;
        repaint();

        return true; // event has been handled
    }

    public boolean keyUp( Event e, int key )
    {
        showStatus( "keyUp: the " + ( char ) key +
            " was released." );

        return true; // event has been handled
    }
}
/*
<applet code="KeyUpDown1.class" width="300" height="300">
</applet>
*/
```

**Output:**

Running the Applet using **appletviewer** as most of the browsers stopped applet support in recent versions.

**Commands:**

To Compile: **javac** KeyUpDown1.java

To View: **appletviewer** KeyUpDown1.java

