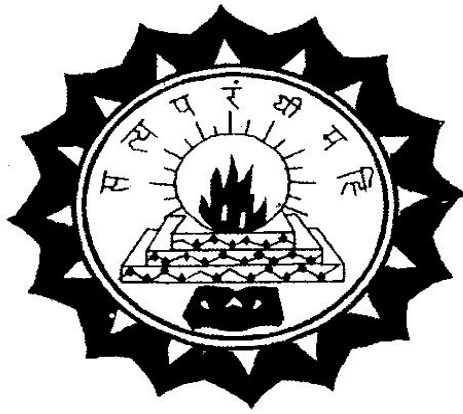


**DWARAKA DOSS GOVERDHAN DOSS VAISHNAV EVENING COLLEGE
(AUTONOMOUS)**

ARUMBAKKAM, CHENNAI-600 106



DEPARTMENT OF COMPUTER APPLICATIONS

Bachelor of Computer Applications

PROGRAMMING IN JAVA LAB

**DWARAKA DOSS GOVERDHAN DOSS VAISHNAV EVENING COLLEGE
(AUTONOMOUS)**

ARUMBAKKAM, CHENNAI-600 106



DEPARTMENT OF COMPUTER APPLICATIONS

PROGRAMMING IN JAVA LAB

Certified that this is a record work done by whose Register no.
..... of II B.C.A./ III Semester during the academic year 2018 - 2018.

Faculty In-charge

Head of the Department

Submitted for practical examination held on at Dwaraka Doss
Goverdhan Doss Vaishnav College, Chennai-106.

Internal Examiner

External Examiner

SL.NO.	DATE	PROGRAM NAME	PAGE	SIGNATURE
JAVA CLASS AND OBJECTS				
1		Class, Methods, Constructors and Objects	1	
STRINGS				
2		String Class and its Methods	3	
3		String Buffer and its Methods	5	
BUILT-IN UTILITY PACKAGES				
4		Random Class	6	
5		Vector Class	8	
INHERITANCE				
6		Multilevel Inheritance	10	
7		Hierarchical Inheritance	13	
8		Multiple Inheritance using Interface	15	
USER-DEFINED PACKAGES				
9		Basic Arithmetic Operations using package	17	
EXCEPTION HANDLING				
10		Built-in Exception Handling	19	
11		User-Defined Exception Handling	21	
THREADS				
12		Thread Manipulation using Thread Class	23	
13		Thread Synchronization	25	
FILES				
14		Copy one file into another file	27	
15		Random Access File	29	
JAVA APPLET AND AWT				
16		Fonts And Colors	31	
17		Parameter Passing Technique	33	
18		Drawing various Shapes	35	
19		Temperature Conversion	37	
20		Simple Calculator	39	

EX. NO: 1

DATE:

CLASS, METHODS, CONSTRUCTORS AND OBJECTS

AIM:

To write a Java program to illustrate class, method, object and constructors.

ALGORITHM:

Step1: Define a class "Student" with name, register number and 3 marks.

Step 2: Declare three constructors – Default, Parameterized and Copy constructor to initialize the data members of the class.

Step 3: In the display() method, print all the details.

PROGRAM:

```
class Student
{
String name;
int regno;
int mark1,mark2,mark3;

// Default constructor
Student()
{
    name="Raju";
    regno=12345;
    mark1=56;
    mark2=47;
    mark3=78;
}

// Parameterized Constructor
Student(String name,int regno,int mark1,int mark2,int mark3)
{
    this.name=name;
    this.regno=regno;
    this.mark1=mark1;
    this.mark2=mark2;
    this.mark3=mark3;
}

//Copy Constructor
Student(Student s)
{
    name=s.name;
```

```

        regno=s.regno;
        mark1=s.mark1;
        mark2=s.mark2;
        mark3=s.mark3;
    }
    void display()
    {
        System.out.println(name + "\t" + regno+ "\t" + mark1+ "\t" + mark2+ "\t" + mark3);
    }
    public static void main(String args[])
    {
        Student s1=new Student();
        Student s2=new Student("Bala",34266,58,96,84);
        Student s3=new Student(s1);
        System.out.println("Name" + "\t" + "Regno"+ "\t" + "Marks1"+ "\t" + "Mark2"+ "\t" +
        "Mark3");
        s1.display();
        s2.display();
        s3.display();
    }
}

```

OUTPUT:

<u>Name</u>	<u>Regno</u>	<u>Marks1</u>	<u>Mark2</u>	<u>Mark3</u>
<u>Raju</u>	<u>12345</u>	<u>56</u>	<u>47</u>	<u>78</u>
<u>Bala</u>	<u>34266</u>	<u>58</u>	<u>96</u>	<u>84</u>
<u>Raju</u>	<u>12345</u>	<u>56</u>	<u>47</u>	<u>78</u>

RESULT:

Hence Class, Methods, Constructors and Objects are verified.

EX. NO: 2

DATE:

STRING CLASS AND ITS METHODS

AIM:

To write a Java program using the methods of String class.

ALGORITHM:

- Step 1: Initialize strings using constructor and "=" operator.
- Step 2: Calculate the length of string using length() method.
- Step 3: Predict the occurrence of a character in the string using indexOf() method.
- Step 4: Convert the case of strings using toUpperCase() and toLowerCase() methods.
- Step 5: Check the equality of strings using equals() method.
- Step 6: Compare the strings using compareTo() method and display the result.

PROGRAM:

```
class Strings
{
    public static void main(String args[])
    {
        String s1=new String("Welcome to Java");
        String s2="WELCOME TO JAVA";
        System.out.println(" The string s1 is : " +s1);
        System.out.println(" The string s2 is : " +s2);
        System.out.println(" Length of the string s1 is : " +s1.length());
        System.out.println(" The first occurrence of o is : " +s1.indexOf('o'));
        System.out.println("String in Upper Case : " +s1.toUpperCase());
        System.out.println("String in Lower Case : " +s2.toLowerCase());
        System.out.println("s1 equals to s2 : " +s1.equals(s2));
        System.out.println("s1 equals ignore case to s2 : " +s1.equalsIgnoreCase(s2));
        int result=s1.compareTo(s2);
        System.out.println("After compareTo()");
        if(result==0)
            System.out.println( s1 + " is equal to "+s2);
        else
            if(result>0)
                System.out.println( s1 + " is greater than to "+s2);
            else
                System.out.println( s1 + " is smaller than to "+s2);
        System.out.println(" Character at an index of 6 is : " +s1.charAt(6));
        String s3=s1.substring(4,12);
        System.out.println(" Extracted substring is :"+s3);
        System.out.println(" After Replacing o with x in s1 : " + s1.replace('o','x'));
        String s4=" This is a book ";
        System.out.println(" The string s4 is :"+s4+"with length "+s4.length());
    }
}
```

```

        s4=s4.trim();
        System.out.println(" After trim() :"+s4+" with length "+s4.length());
    }
}

```

OUTPUT:

The string s1 is : Welcome to Java
The string s2 is : WELCOME TO JAVA
Length of the string s1 is : 15
The first occurrence of o is : 4
String in Upper Case : WELCOME TO JAVA
String in Lower Case : welcome to java
s1 equals to s2 : false
s1 equals ignore case to s2 : true
After compareTo()
Welcome to Java is greater than to WELCOME TO JAVA
Character at an index of 6 is :e
Extracted substring is :ome to J
After Replacing o with x in s1 : Welcxme tx Java
The string s4 is : This is a book with length 21
After trim() :This is a book with length 14

RESULT:

Hence String Class and its Methods are verified .

EX. NO: 3

DATE:

STRING BUFFER CLASS AND ITS METHODS

AIM:

To write a Java program using the methods of StringBuffer class.

ALGORITHM:

- Step 1: Initialize a string using StringBuffer constructor.
- Step 2: Calculate the length using length() method.
- Step 3: Print the character at the given position using charAt() method.
- Step 4: Replace a character using setCharAt() method.
- Step 5: Append a string at the end using append() method.
- Step 6: Insert a string in the middle using insert() method.
- Step 7: Delete a substring from the main string using delete() method.

PROGRAM:

```
class Str
{
    public static void main(String arg[])
    {
        StringBuffer sb=new StringBuffer("This is my college");
        System.out.println("This string sb is : " +sb);
        System.out.println("The length of the string sb is : " +sb.length());
        System.out.println("The character at index 6 is : " +sb.charAt(6));
        sb.setCharAt(3,'x');
        System.out.println("After setting x at position 3 : " +sb);
        System.out.println("After appending : " +sb.append(" in arumbakkam "));
        System.out.println("After inserting : " +sb.insert(19,"DGVC "));
        System.out.println("After deleting : " +sb.delete(19,22));
    }
}
```

OUTPUT:

The length of the string sb is : 18
 The character at index 6 is : s
 After setting x at position 3 : Thix is my college
 After appending : Thix is my college in arumbakkam
 After inserting : Thix is my college DGVC in arumbakkam
 After deleting : Thix is my college C in arumbakkam

RESULT:

Hence String Buffer and its Methods are verified .

EX. NO: 4

DATE:

RANDOM CLASS

AIM:

To generate random numbers using java utility package.

ALGORITHM:

Step 1: Declare an object for random class.

Step 2: Initialize an array of size 10.

Step 3: Using nextInt() method, generate random numbers and assign to array a.

Step 4: Arrange the random values in ascending order and display maximum and minimum number in an array.

PROGRAM:

```
// Random number generation
import java.util.*;
public class ArrayRandom
{
    public static void main(String args[])
    {
        Random r = new Random();
        int a[] = new int[10];
        for(int i = 0; i <10; i++)
            a[i] = r.nextInt(50) + 1;
        System.out.println("Random Array generated is .....");
        for(int i = 0; i <10; i++)
            System.out.println(a[i]);
        for( int i = 0; i < 10; i++ )
            for (int j=i+1;j<10;j++)
            {
                if(a[i] > a[j])
                {
                    int t=a[i];
                    a[i]=a[j];
                    a[j]=t;
                }
            }
        System.out.println("The sorted array is.....");
        for (int i=0;i<10;i++)
            System.out.println(a[i]);
        System.out.println("Maximum Number in the above list is: "+ a[9]);
        System.out.println("Minimum Number in the above list is:"+a[0]);
    }
}
```

OUTPUT:

Random Array generated is

22

32

26

15

44

17

28

37

3

35

The sorted array isâ??â??

3

15

17

22

26

28

32

35

37

44

Maximum Number in the above list is: 44

Minimum Number in the above list is:3

RESULT:

Hence Random Class are verified.

EX. NO: 5

DATE:

VECTOR CLASS

AIM:

To manipulate vector objects with its methods.

ALGORITHM:

- Step 1: Declare an object V for a Vector class.
- Step 2: Using addElement() method, add the values to the vector.
- Step 3: Using nextLine() method read the values of x.
- Step 4: Check whether 'x' is in V by using contains() method.
- Step 5: Create an object for enumeration and assign the values of V to it using elements() method.
- Step 6: Print the vector elements using nextElements() method.
- Step 7: Display first & last element of vector using firstElement() and lastElement() methods.

PROGRAM:

```
import java.util.*;
class VectorDemo
{
    public static void main(String args[])
    {
        Vector v = new Vector(3, 2); // initial size is 3, increment is 2
        System.out.println("Initial size: " + v.size());
        System.out.println("Initial capacity: " + v.capacity());
        v.addElement(1);
        v.addElement(2);
        v.addElement('a');
        v.addElement(4);
        System.out.println("Now the size of the vector: " + v.size());
        v.addElement(5.45);
        System.out.println("Current capacity: " + v.capacity());
        v.addElement(7);
        System.out.println("Current capacity: " + v.capacity());
        v.addElement(9.4);
        System.out.println("Current capacity: " + v.capacity());
        v.addElement('b');
        v.addElement(12);
        System.out.println("Now the size of the vector: " + v.size());
        System.out.println("First element: " + v.firstElement());
        System.out.println("Last element: " + v.lastElement());
        // Searching an integer element
        System.out.println("Enter the Integer element to be checked:");
        Scanner s=new Scanner(System.in);
```

```

int x=Integer.parseInt(s.nextLine());
if(v.contains(x))
    System.out.println("Vector contains "+x);
else
    System.out.println("Vector doesn't contains "+x);
// Print the elements in the vector
Enumeration obj = v.elements();
System.out.println("\nElements in vector:");
while(obj.hasMoreElements())
    System.out.println(obj.nextElement());
}
}

```

OUTPUT:

Initial size: 0

Initial capacity: 3

Now the size of the vector: 4

Current capacity: 5

Current capacity: 7

Current capacity: 7

Now the size of the vector: 9

First element: 1

Last element: 12

Enter the Integer element to be checked:

5

Vector doesn't contains 5

Elements in vector:

1

2

a

4

5.45

7

9.4

b

12

RESULT:

Hence Vector Class are verified.

EX. NO: 6

DATE:

MULTILEVEL INHERITANCE

AIM:

To write a Java Program to show the concept of multilevel inheritance.

ALGORITHM:

- Step 1: Create a class name as Student & declare variable name & rollno.
- Step 2: Declare a method get1() and get the information about the Student.
- Step 3: Create a sub class Marks which extends super class students.
- Step 4: Read 3 marks using get2() method.
- Step 5: Create a sub class Result which extends super class Marks.
- Step 6: Declare put() method and total by adding 3 marks.
- Step 7: if mark1 and mark2 and mark3 >=35
 Print result as 'pass'
 else print 'fail'
 and
 Print all other student information according.
- Step 8: Create an object for Result and call the methods using that object.

PROGRAM:

```
// Multilevel Inheritance
import java.util.*;
class Student
{
    int rollno;
    String name;
    void get1()
    {
        Scanner s=new Scanner(System.in);
        System.out.println("Enter the student roll number");
        rollno=Integer.parseInt(s.nextLine());
        System.out.println("Enter the student name");
        name=s.nextLine();
    }
}
class Marks extends Student
{
    int marks[]=new int[3];
    void get2()
    {
        Scanner s=new Scanner(System.in);
        System.out.println("Enter the 3 marks of the student" );
        for(int i=0;i<3;i++)
```

```

        marks[i]=Integer.parseInt(s.nextLine());
    }
}

class Result extends Marks
{
    int total;
    String result;
    void put()
    {
        total=marks[0]+marks[1]+marks[2];
        if (marks[0]>=35 && marks[1]>=35 && marks[2]>=35)
            result="pass";
        else
            result="fail";
        System.out.println("STUDENT'S ROLL NUMBER="+rollno);
        System.out.println("STUDENT'S NAME    "+name.toUpperCase());
        System.out.println("STUDENT'S MARK 1   "+marks[0]);
        System.out.println("STUDENT'S MARK 2   "+marks[1]);
        System.out.println("STUDENT'S MARK 3   "+marks[2]);
        System.out.println("STUDENT'S TOTAL    "+total);
        System.out.println("STUDENT'S RESULT  "+result.toUpperCase());
    }
}

public class Multilevel
{
    public static void main(String args[])
    {
        Result r=new Result();
        r.get1();
        r.get2();
        r.put();
    }
}

```

OUTPUT:

Enter the student roll number

34

Enter the student name

Kumar

Enter the 3 marks of the student

89

85

82

STUDENT'S ROLL NUMBER=34

STUDENT'S NAME =KUMAR

STUDENT'S MARK 1 =89

STUDENT'S MARK 2 =85

STUDENT'S MARK 3 =82

STUDENT'S TOTAL =256

STUDENT'S RESULT =PASS

RESULT:

Hence Multilevel Inheritance are verified.

EX. NO: 7

DATE:

HIERARCHICAL INHERITANCE

AIM:

To illustrate the concept of hierarchical inheritance in Java.

ALGORITHM:

Step 1: Design a class called "Shape" with d1,d2 as its members. Initialize them by a method "setvalues".

Step 2: Derive a class "Triangle" from "Shape" and calculate the area using d1 and d2 and display the value.

Step 3: Derive a class "Rectangle" from "Shape" and calculate the area using d1 and d2 and display the value.

Step 4: Create objects for these subclasses in main method accordingly and invoke them.

PROGRAM:

```
// Program showing Hierarchical Inheritance
class Shape
{
    int d1;
    int d2;
    public void setvalues(int dimen1, int dimen2 )
    {
        d1 = dimen1;
        d2 = dimen2;
    }
}
class Triangle extends Shape
{
    int ar ;
    public void area()
    {
        ar = (d1 * d2 )/ 2;
        System.out.println("Area of triangle:"+ar);
    }
}
class Rectangle extends Shape
{
    int ar;
    public void area()
    {
        ar = d1* d2;
```



```
        System.out.println("Area of rectangle :"+ar);
    }
}
public class Inher
{
    public static void main(String args [ ])
    {
        Triangle t = new Triangle();
        t.setvalues(10,10);
        Rectangle r = new Rectangle();
        r.setvalues(5, 10);
        t.area();
        r.area();
    }
}
```

OUTPUT:

Area of triangle:50
Area of rectangle :50

RESULT:

Hence Hierarchical Inheritance are verified.

EX. NO: 8

DATE:

MULTIPLE INHERITANCE USING INTERFACE

AIM:

To write a Java program for multiple inheritance using the concept of interface.

ALGORITHM:

Step 1: Define an interface "Area" with a constant pi and abstract method calc().

Step 2: Define a class "Shape" with a member radius.

Step 3: Initialize radius using constructor Shape().

Step 4: Define a sub-class "Circle" with two super classes- Area and Shape.

Step 5: Compute the area and circumference of the circle and display the values.

PROGRAM:

```
public class Multiple
{
    public static void main(String args[])
    {
        Circle c=new Circle(15.4);
        c.calc();
    }
}

interface Area
{
    final double pi=3.14;
    void calc();
}

class Shape
{
    double radius;
    Shape(double r)
    {
        radius=r;
    }
}

class Circle extends Shape implements Area
{
    Circle(double r)
    {
        super(r);
    }
}
```

```
public void calc()
{
    double area=pi*radius*radius;
    double circum=2.0*pi*radius;
    System.out.println ("Radius:"+radius);
    System.out.println ("Area of circle:"+area);
    System.out.println("Circumference of circle:"+circum);
}
}
```

OUTPUT:

Radius:15.4
Area of circle:744.6824
Circumference of circle:96.712

RESULT:

Hence Multiple Inheritance Using Interface are verified.

EX. NO: 9

DATE:

BASIC ARITHMETIC OPERATIONS USING PACKAGE

AIM:

To write a Java program using packages.

ALGORITHM:

- Step 1: Define a package “pack” with methods performing basic arithmetic operations.
- Step 2: Save the package in the folder “pack” and compile it.
- Step 3: Write the main program importing the package “pack” and call the methods defined within the package.

PROGRAM:

```
import pack.*;
import java.util.*;
class MainPackage
{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        int a,b;
        System.out.println("Enter 2 numbers....");
        a=Integer.parseInt(s.nextLine());
        b=Integer.parseInt(s.nextLine());
        Arithmetic obj = new Arithmetic();
        System.out.println(a + " + " + b + " = " +obj.add(a,b));
        System.out.println(a + " - " + b + " = " +obj.sub(a,b));
        System.out.println(a + " * " + b + " = " +obj.mul(a,b));
        System.out.println(a + " / " + b + " = " +obj.div(a,b));
    }
}

package pack;
public class Arithmetic
{
    public int add(int a,int b)
    {
        return(a+b);
    }
    public int sub(int a, int b)
    {
        return(a-b);
    }
    public int mul(int a, int b)
```

```
        {  
            return(a*b);  
        }  
    public int div(int a, int b)  
    {  
        return(a/b);  
    }  
}
```

OUTPUT:**Enter 2 numbers....****5****6****5 + 6 = 11****5 - 6 = -1****5 * 6 = 30****5 / 6 = 0****RESULT:**

Hence Basic Arithmetic Operations Using Package are verified.

EX. NO: 10

DATE:

BUILT-IN EXCEPTION HANDLING

AIM:

To write a Java program to show the working of built-in exception handling.

ALGORITHM:

Step 1: Divide an integer by 0 and raise "Divide by zero" exception.

Step 2: Initialize the value of array greater than its size and raise "Array Index Out of Bounds" exception.

Step 3: Instead of entering number, enter alphabetical characters and raise "Number Format" exception.

Step 4: Define a class and methods for invoking above exceptions.

PROGRAM:

```
import java.util.*;
public class Builtin
{
    void arith()
    {
        int a,b;
        try
        {
            System.out.println("1. Inside Arithmetic");
            a=0;
            b=10/a;
            System.out.println("a="+a+" b="+b);
        }
        catch(ArithmeticException e)
        {
            System.out.println("\t"+"Divide by zero is an arithmetic exception\n"+" \t"+e);
        }
    }
    void array()
    {
        try
        {
            System.out.println("2. Inside Array");
            int a[]={1,2,3,4,5};
            a[10]=78;
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println("\t"+"Wrong index in the given array"+" \n"+" \t"+e);
        }
    }
}
```

```

void num()
{
    Scanner s=new Scanner(System.in);
    int x[]=new int[5];
    System.out.println("Enter 5 elements:");
    try
    {
        for (int i=0;i<5;i++)
            x[i]=Integer.parseInt(s.nextLine());
    }
    catch(NumberFormatException e)
    {
        System.out.println("\t"+"Wrong input"+"\\n"+"\\t"+e);
    }
}
public static void main(String args[])
{
    Builtin obj=new Builtin();
    obj.arith();
    obj.array();
    obj.num();
}
}

```

OUTPUT:

1. Inside Arithmetic

Divide by zero is an arithmetic exception

java.lang.ArithmeticException: / by zero

2. Inside Array

Wrong index in the given array

java.lang.ArrayIndexOutOfBoundsException: 10

Enter 5 elements:

5
6
6
2
3

RESULT:

Hence Built-in Exception Handling are verified.

EX. NO: 11

DATE:

USER-DEFINED EXCEPTION HANDLING

AIM:

To write a Java program for user-defined exception handling.

ALGORITHM:

- Step 1: Define a sub-class "MyException" with super class as "Throwable".
- Step 2: Declare arrays for account number, account holder names and balance maintained by them.
- Step 3: Define an user-defined exception "MyException" to raise exception for the account holders whose balance is less than 2000.

PROGRAM:

```
class MyException extends Throwable
{
    int accno[] = {1001,1002,1003,1004,1005};
    String name[] = {"Hari","Siva","Bhanu","Rama","Chandu"};
    double bal[] = {2500,3500,1500,1000,6000};
    MyException(String S)
    {
        super(S);
    }
    public static void main(String args[])
    {
        MyException me = new MyException("");
        System.out.println("AccNo \t Name \t Balance ");
        for(int i=0;i<5;i++)
        {
            try
            {
                System.out.println(me.accno[i]+ "\t" + me.name[i] + "\t" +me.bal[i] );
                if( me.bal[i] < 2000 )
                    throw new MyException("InsufficientBalance");
            }
            catch(MyException e)
            {
                System.out.println(e);
                continue;
            }
        }
    }
}
```


OUTPUT:

AccNo	Name	Balance
1001	Hari	2500.0
1002	Siva	3500.0
1003	Bhanu	1500.0

MyException: InsufficientBalance

1004	Rama	1000.0
------	------	--------

MyException: InsufficientBalance

1005	Chandu	6000.0
------	--------	--------

RESULT:

Hence User-Defined Exception Handling are verified.

EX. NO: 12

DATE:

THREAD MANIPULATION USING THREAD CLASS

AIM:

To illustrate the concept of threads in Java.

ALGORITHM:

Step 1: Create a class A which extends Thread. In the run() method, use the yield() method, to relinquish the control to other thread.

Step 2: Create a class B which extends Thread. In the run() method, use the stop() method, to abort the thread.

Step 3: Create a class C which extends Thread that runs normally.

Step 4: Create objects for classes A,B,C and start them accordingly.

PROGRAM:

// Illustration about threads

class A extends Thread

{

public void run()

{

for (int i=1;i<=5;i++)

{

if (i==1)

Thread.yield();

System.out.println("From thread A:i="+i);

}

System.out.println("Exited from A");

}

}

class B extends Thread

{

public void run()

{

for (int i=1;i<=5;i++)

{

if (i==3) stop();

System.out.println("From thread B:i="+i);

}

System.out.println("Exited from B");

}

}

class C extends Thread

{

public void run()

{

```

        for (int i=1;i<=5;i++)
            System.out.println("From thread C:i="+i);
        System.out.println("Exited from C");
    }
}
class Threads
{
    public static void main(String args[])
    {
        A a=new A();
        B b=new B();
        C c=new C();
        a.start();
        b.start();
        c.start();
    }
}

```

OUTPUT:

From thread A:i=1
From thread C:i=1
From thread C:i=2
From thread C:i=3
From thread C:i=4
From thread C:i=5
Exited from C
From thread B:i=1
From thread B:i=2
From thread A:i=2
From thread A:i=3
From thread A:i=4
From thread A:i=5
Exited from A

RESULT:

Hence Thread Manipulation Using Thread Class are verified.

EX. NO: 13

DATE:

THREAD SYNCHRONIZATION

AIM:

To implement synchronization method among threads.

ALGORITHM:

Step 1: Create 2 threads acting on a single object.

Step 2: Reserve tickets for the person whose wanted satisfies the availability of tickets.

Step 3: Print results accordingly.

PROGRAM:

class Reserve implements Runnable

```
{
    int available = 3;
    int wanted;
    Reserve (int i)
    { wanted = i; }
    public void run()
    {
        synchronized (this)
        {
            String name = Thread.currentThread ().getName ();
            System.out.println ("Number of berths available: " + available);
            if ( available >= wanted)
            {
                System.out.println (wanted + " berths allotted to: " + name);
                available = available - wanted;
            }
            else
                System.out.println ("Sorry, no berths available for "+name);
        }
    }
}
```

class Safe

```
{
    public static void main(String args[])
    {
        Reserve obj = new Reserve (2);
        Thread t1 =new Thread (obj);
        Thread t2 = new Thread (obj);
        t1.setName ("First Person");
```

```
        t2.setName ("Second Person");  
        t1.start ();  
        t2.start ();  
    }  
}
```

OUTPUT:

Number of berths available: 3

3 berths allotted to: First Person

Number of berths available: 0

Sorry, no berths available for Second Person

RESULT:

Hence Thread Synchronization are verified.

EX. NO: 14

DATE:

COPY ONE FILE CONTENTS INTO ANOTHER FILE

AIM:

To copy the content of one file into another file using file streams.

ALGORITHM:

Step1: Read a file name from the user.

Step 2: Read a target file name from the user.

Step 3: Read the content of the source file name & copy the content as byte into target file name.

Step 4: Repeat step 3 until the source file reaches its end.

PROGRAM:

```
import java.io.*;
import java.util.*;
public class FileCopy
{
    public static void main(String a[]) throws IOException
    {
        int d=0;
        FileInputStream fi=null;
        FileOutputStream fo=null;
        Scanner s=new Scanner(System.in);
        System.out.println("Enter existing file name:");
        String s1=s.nextLine();
        System.out.println("Enter new file name:");
        String s2=s.nextLine();
        try
        {
            fi=new FileInputStream(s1);
            fo=new FileOutputStream(s2);
            System.out.println("Copying file");
            while((d=fi.read())!=-1)
                fo.write((byte)d);
            System.out.println("File is copied");
        }
        catch(IOException e)
        { System.out.println(e); }
        finally
        {
            fi.close();fo.close();}
    }
}
```

OUTPUT:

Enter existing file name:

new 1

Enter new file name:

new 2

Copying file

File is copied

RESULT:

Hence Copy One File Into Another File are verified.

EX. NO: 15

DATE:

RANDOM ACCESS FILES

AIM:

To create, write and read the contents of random access files.

ALGORITHM:

Step 1: Create a random access file "rand.txt" by opening in read write mode.

Step 2: Write character, integer, double and Boolean values into the file.

Step 3: Read those file contents.

Step 4: Display the file length using file.length() method.

Step 5: Place the file pointer to any position in the file using seek() method.

PROGRAM:

```
import java.io.*;
class RandRW
{
    public static void main(String[] args)
    {
        RandomAccessFile file = null;
        try
        {
            file = new RandomAccessFile("rand.txt","rw");
            file.writeChar('V');           //Writing to the file
            file.writeInt(999);
            file.writeDouble(99.99);
            file.seek(0);                  //Go to the beginning and read from the file
            System.out.println(file.readChar());
            System.out.println(file.readInt());
            System.out.println(file.readDouble());
            file.seek(2);                  //Go to the Second Item
            System.out.println(file.readInt());
            file.seek(file.length());      //Go to the end and append true to the file
            file.writeBoolean(true);
            file.seek(4);                  //Go to the fourth Item
            System.out.println(file.readBoolean());
            file.close();
        }
        catch(IOException e)
        { System.out.println(e); }
    }
}
```


OUTPUT:

V
999
99.99
999
true

RESULT:

Hence Random Access File are verified.

EX. NO: 16

DATE:

WORKING WITH FONTS AND COLORS

AIM:

To execute an applet program in show the various fonts and colors.

ALGORITHM:

Step 1: Write the applet tag with code , height , width.

Step 2: In the paint method , do the following.

A: Set font f1,f2,f3,f4 with various font faces, styles & size.

B: Set color as gray , font as f1 & print string "Be happy & hopeful".

C: Set color as blue , font as f2 & print string " Be self confident".

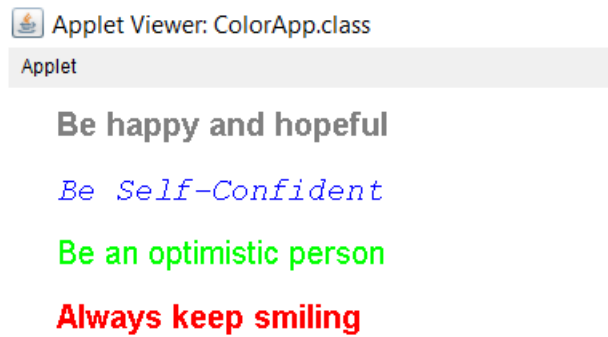
D: Set color as green , font as f3 & print string "Be an optimistic Person".

E: Set color as red , font as f4 & print string "Always keep smiling".

Step 3: Run the applet using the command "appletviewer program.java".

PROGRAM:

```
// Changing fonts and colors
import java.awt.*;
import java.applet.*;
public class ColorApp extends Applet
{
    public void paint(Graphics g)
    {
        Font f1=new Font("TimesNewRoman", Font.BOLD,20);
        Font f2=new Font("Courier", Font.ITALIC,20);
        Font f3=new Font("Helvotica", Font.PLAIN,20);
        Font f4=new Font("Arial", Font.BOLD,20);
        g.setColor(Color.gray);
        g.setFont(f1);
        g.drawString("Be happy and hopeful",30,30);
        g.setColor(Color.blue);
        g.setFont(f2);
        g.drawString("Be Self-Confident",30,70);
        g.setColor(Color.green);
        g.setFont(f3);
        g.drawString("Be an optimistic person",30,110);
        g.setColor(Color.red);
        g.setFont(f4);
        g.drawString("Always keep smiling",30,150);
    }
}
/*<applet code="ColorApp.class" height="30" width="300"></applet>*/
```

OUTPUT:**RESULT:**

Hence Fonts And Colors are verified.

EX. NO: 17

DATE:

PARAMETER PASSING TECHNIQUE

AIM:

To write an applet program incorporating parameter passing technique.

ALGORITHM:

Step 1: In the applet tag, define two parameters.

Step 2: In the init() method, get those two parameters and compare them.

Step 3: In the paint() method, display whether two values are equal or which is greater and smaller.


PROGRAM:

```

/*
<applet code="MyApplet2.class" width = 600 height= 450>
<param name = "t1" value="102">
<param name = "t2" value ="101">
</applet>
*/

import java.applet.*;
import java.awt.*;
public class MyApplet2 extends Applet
{
    int n1,n2,n;
    public void init()
    {
        n1 = Integer.parseInt(getParameter("t1"));
        n2 = Integer.parseInt(getParameter("t2"));
    }
    public void paint(Graphics g)
    {
        g.drawString("First Value:"+n1,100,50);
        g.drawString("Second Value:"+n2,100,70);
        if (n1==n2)
            g.drawString("Both values are equal",100,90);
        else if (n1>n2)
            g.drawString(n1+ " is greater than "+n2,100,90);
        else
            g.drawString(n2+ " is greater than "+n1,100,90);
    }
}

```

OUTPUT: Applet Viewer: MyApplet2.class

Applet

First Value:102
Second Value:101
102 is greater than 101

RESULT:

Hence Parameter Passing Technique are verified.

EX. NO: 18

DATE:

DRAWING VARIOUS SHAPES

AIM:

To write a Java Applet program for incorporating graphics for drawing shapes.

ALGORITHM:

Step 1: Write the applet tag with code, height , width.

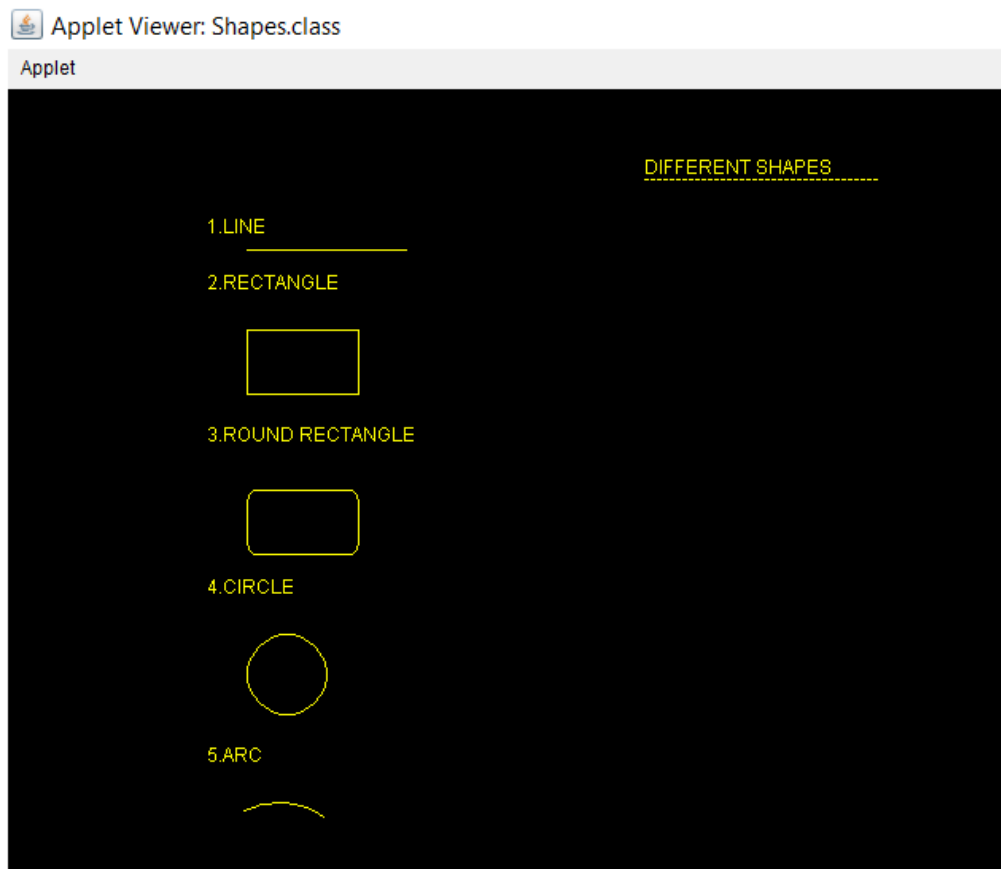
Step 2: In the paint method , draw shapes like line, rectangle , rounded rectangle , oval, circle, arc, polygon.

Step 3: Set different colors for the shapes.

Step 4: Run the applet using the command “appletviewer program.java”.

PROGRAM:

```
// Drawing various shapes
import java.awt.*;
import java.applet.*;
//<applet code="Shapes.class" height=1000 width=1000> </applet>
public class Shapes extends Applet
{
    public void init()
    {
        setForeground(Color.yellow);
        setBackground(Color.black);
    }
    public void paint(Graphics g)
    {
        g.drawString("DIFFERENT SHAPES",400,53);
        g.drawString("-----",400,60);
        g.drawString("1.LINE",125,90);
        g.drawLine(150,100,250,100);
        g.drawString("2.RECTANGLE",125,125);
        g.drawRect(150,150,70,40);
        g.drawString("3.ROUND RECTANGLE",125,220);
        g.drawRoundRect(150,250,70,40,10,20);
        g.drawString("4.CIRCLE",125,315);
        g.drawOval(150,340,50,50);
        g.drawString("5.ARC",125,420);
        g.drawArc(125,445,90,80,50,70);
    }
}
```

OUTPUT:**RESULT:**

Hence Drawing Various Shapes are verified.

EX. NO: 19

DATE:

TEMPERATURE CONVERSION

AIM:

To write a Java applet program for converting fahrenheit to Celsius and viceversa using AWT controls like textbox, label, button etc.

ALGORITHM:

Step 1: Write the applet tag with code=name of program, height & width of the browser.

Step 2: Create 2 textboxes & label them accordingly.

Step 3: Create 2 buttons, one for F2C (Fahrenheit to Celcius) & other for C2F (Celcius to Fahrenheit).

Step 4: Write action listeners for 2 buttons using actionPerformed method.

Step 5: If we press F2C, compute $F = 5/9(C - 32)$ & display ' f ' as result.

Step 6: If we press C2F, compute $C = 9 * F / 5 + 32$ & display ' c ' as result.

PROGRAM:

```
// Temperature conversion
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
//<applet code="Farcel.class" width=500 height=500></applet>
public class Farcel extends Applet implements ActionListener
{
    Label l1,l2;
    TextField t1,t2;
    Button b1,b2;
    public void init()
    {
        l1=new Label("Input");
        l2=new Label("Output");
        b1=new Button("F2C");
        b2=new Button("C2F");
        t1=new TextField(5);
        t2=new TextField(5);
        add(l1);add(t1);
        add(l2);add(t2);
        add(b1);add(b2);
        b1.addActionListener(this);
        b2.addActionListener(this);
    }
    public void actionPerformed(ActionEvent ae)
    {
        float ans=0,i=0;
        String s=ae.getActionCommand();
```

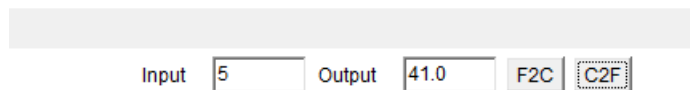


```

try
{
    i=Float.parseFloat(t1.getText());
    if (s.equals("F2C"))
        ans=5*(i-32)/9;
    else
        if (s.equals("C2F"))
            ans=(9*i)/5+32 ;
        t2.setText(Float.toString(ans));
    }
    catch(NumberFormatException e)
    {
        System.out.println("Please enter only numbers");
    }
}
}

```

OUTPUT:



Input Output

RESULT:

Hence Temperature Conversion are verified.

EX. NO: 20

DATE:

SIMPLE CALCULATOR

AIM:

To write a Java Applet to design and run simple calculator operations.

ALGORITHM:

- Step 1: Write the applet tag with code=name of program, height & Width of the browser.
- Step 2: Create a text field.
- Step 3: Create buttons for addition, subtraction, multiplication, division, equal and clear.
- Step 4: Write action listeners for 6 buttons using actionPerformed method.
- Step 5: If we press add, subtract, multiplication or division button, the number entered first will be saved in n1 and the textfield is cleared to get the second number as n2.
- Step 6: If we press equal button, add or subtract or multiply or divide n1 and n2 and display the result in textfield.
- Step 7: If we press clear button, clear the textbox.

PROGRAM:

```
// Calculator
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
//<applet code="Calci.class" width=500 height=500></applet>
public class Calci extends Applet implements ActionListener
{
    TextField t;
    String op,option;
    Button b1,b2,b3,b4,b5,b6;
    int n1,n2,n;
    public void init()
    {
        t=new TextField(10);
        b1=new Button("+");
        b2=new Button("-");
        b3=new Button("*");
        b4=new Button("/");
        b5=new Button("=");
        b6=new Button("CLS");
        add(t);
        add(b1);add(b2);add(b3);add(b4);add(b5);add(b6);
        b1.addActionListener(this);
        b2.addActionListener(this);
```

```

        b3.addActionListener(this);
        b4.addActionListener(this);
        b5.addActionListener(this);
        b6.addActionListener(this);
    }
    public void actionPerformed(ActionEvent ae)
    {
        op=ae.getActionCommand();
        if (op.equals("CLS"))
        {
            t.setText("");
            t.requestFocus();
        }
        else
        if (op.equals("+") || op.equals("-") || op.equals("/") || op.equals("*"))
        {
            option = op;
            n1=Integer.parseInt(t.getText());
            t.setText("");
            t.requestFocus();
        }
        else if (op.equals("="))
        {
            n2=Integer.parseInt(t.getText());
            if (option.equals("+"))
                n=n1+n2;
            if (option.equals("-"))
                n=n1-n2;
            if (option.equals("*"))
                n=n1*n2;
            if (option.equals("/"))
                n=n1/n2;
            t.setText( Integer.toString(n));
        }
    }
}

```

OUTPUT:**RESULT:**

Hence Simple Calculator are verified.