

**AURORA'S PG COLLEGE
MOOSARAMBAGH
MCA DEPARTMENT**

**JAVA LAB
MANUAL**

Introduction to JAVA:

JAVA was developed by Sun Microsystems Inc in 1991, later acquired by Oracle Corporation. It was developed by James Gosling and Patrick Naughton. It is a simple programming language. Writing, compiling and debugging a program is easy in java. It helps to create modular programs and reusable code.

Java terminology

Before we start learning Java, let's get familiar with common java terms.

Java Virtual Machine (JVM)

This is generally referred to as JVM. Before we discuss about JVM, let's see the phases of program execution. Phases are as follows: we write the program, then we compile the program and at last we run the program.

- 1) Writing of the program is of course done by java programmer like you and me.
- 2) Compilation of program is done by javac compiler, javac is the primary java compiler included in java development kit (JDK). It takes java program as input and generates java bytecode as output.
- 3) In third phase, JVM executes the bytecode generated by compiler. This is called program run phase.

So, now that we understand that the primary function of JVM is to execute the bytecode produced by compiler. **Each operating system has different JVM, however the output they produce after execution of bytecode is same across all operating systems.** That is why we call java as platform independent language.

bytecode

As discussed above, javac compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. The bytecode is saved in a .class file by compiler.

Java Development Kit(JDK)

While explaining JVM and bytecode, I have used the term JDK. Let's discuss about it. As the name suggests this is complete java development kit that includes JRE (Java Runtime Environment), compilers and various tools like JavaDoc, Java debugger etc.

In order to create, compile and run Java program you would need JDK installed on your computer.

Java Runtime Environment(JRE)

JRE is a part of JDK which means that JDK includes JRE. When you have JRE installed on your system, you can run a java program however you won't be able to compile it. JRE includes JVM, browser plugins and applets support. When you only need to run a java program on your computer, you would only need JRE.

These are the basic java terms that confuse beginners in java. For complete java glossary refer this link:

<https://docs.oracle.com/javase/tutorial/information/glossary.html>

Main Features of JAVA

Java is a platform independent language

Compiler(javac) converts source code (.java file) to the byte code(.class file). As mentioned above, JVM executes the bytecode produced by compiler. This byte code can run on any platform such as Windows, Linux, Mac OS etc. Which means a program that is compiled on windows can run on Linux and vice-versa. Each operating system has different JVM, however the output they produce after execution of bytecode is same across all operating systems. That is why we call java as platform independent language.

Java is an Object Oriented language

Object oriented programming is a way of organizing programs as collection of objects, each of which represents an instance of a class.

4 main concepts of Object Oriented programming are:

1. [Abstraction](#)
2. [Encapsulation](#)
3. [Inheritance](#)
4. [Polymorphism](#)

Simple

Java is considered as one of simple language because it does not have complex features like Operator overloading, [Multiple inheritance](#), pointers and Explicit memory allocation.

Robust Language

Robust means reliable. Java programming language is developed in a way that puts a lot of emphasis on early checking for possible errors, that's why java compiler is able to detect errors that are not easy to detect in other programming languages. The main features of java that makes it robust are garbage collection, Exception Handling and memory allocation.

Secure

We don't have pointers and we cannot access out of bound arrays (you get `ArrayIndexOutOfBoundsException` if you try to do so) in java. That's why

several security flaws like stack corruption or buffer overflow is impossible to exploit in Java.

Java is distributed

Using java programming language we can create distributed applications. RMI(Remote Method Invocation) and EJB(Enterprise Java Beans) are used for creating distributed applications in java. In simple words: The java programs can be distributed on more than one systems that are connected to each other using internet connection. Objects on one JVM (java virtual machine) can execute procedures on a remote JVM.

Multithreading

Java supports [multithreading](#). Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilisation of CPU.

Portable

As discussed above, java code that is written on one machine can run on another machine. The platform independent byte code can be carried to any platform for execution that makes java code portable.

PROGRAMS:

1.Creating Class and Objects

```
/* Program to demonstrate class and objects */
```

```
class Box  
{  
    double l,w,h;  
}
```

```
public class Boxdemo  
{
```

```
public static void main(String[] args)
{
    Box b1 = new Box();
    Box b2 = new Box();
    b1.l=30;
    b1.h=20;
    b1.w=10;
    b2.l=50;
    b2.h=30;
    b2.w=40;

    System.out.println("BOX b1 Details:");
    System.out.println("Height" + b1.h);
    System.out.println("Width" + b1.w);
    System.out.println("Length" + b1.l);

    System.out.println("BOX b2 Details");
    System.out.println("Height" + b2.h);
    System.out.println("Width" + b2.w);
    System.out.println("Length" + b2.l);
}
}
```

Output:

```
BOX b1 Details:
Height 20.0
Width 10.0
Length 30.0
BOX b2 Details
Height 30.0
Width 40.0
Length 50.0
```

2. Program To Demonstrate Class With Methods

```
class Rectangle
{
    double h,w;
    void setValues(double x, double y)
    {
        h=x;
        w=y;
    }

    void area(double x,double y)
    {
        setValues(x,y);
        System.out.println("Area of rectangle is " + (h*w));
    }

    double perimeter(double x, double y)
    {
        setValues(x,y);
        return(2*(h+w));
    }
}

public class Demo
{
    public static void main(String[] args)
    {
        Rectangle r = new Rectangle();
        r.area(30,20);
        double p = r.perimeter(30,20);
        System.out.println("Perimeter is " + p);
    }
}
```

Output:

Area of rectangle is 600.0
Perimeter is 100.0

3. Program to demonstrate Objects as Parameters to Methods

```
class Test
{
    int x;
    void increment(Test o)
    {
        o.x=x*10;
    }
}

public class Demo1
{
    public static void main(String args[])
    {
        Test t = new Test();
        t.x=10;
        Test t1 = new Test();
        t1.x=20;

        System.out.println("t.x=" + t.x);
        System.out.println("t1.x=" + t1.x);
        t.increment(t1);
        System.out.println("After calling increment() t1.x=" + t1.x);
    }
}
```

Output:

```
t.x=10
t1.x=20
After calling increment() t1.x=100
```

4. Program to demonstrate Constructor Overloading

```
class Student
{
    int rollno;
    String name;
    double m1,m2,m3,avg;

    Student()
    {
        rollno=1;
        name="xyz";
        m1=75;
        m2=80;
        m3=70;
        avg=0;
    }

    Student(int r, String n, double m11, double m12, double m13)
    {
        rollno=r;
        name=n;
        m1 = m11;
        m2 = m12;
        m3 = m13;
        avg = 0;
    }

    double calculate()
    {
        avg = (m1+m2+m3)/3;
        return(avg);
    }

    void details()
    {
        System.out.println("Name: " + name);
        System.out.println("Roll No: " + rollno);
        System.out.println("Average: " + calculate());
    }
}

public class Demo2
{
    public static void main(String[] args)
    {
        Student s1 = new Student();
        Student s2 = new Student();
    }
}
```



```
s1.details();  
s2.details();  
}  
}
```

Output:

Name: xyz
Roll No: 1
Average: 75.0
Name: xyz
Roll No: 1
Average: 75.0

5.Program to implement Stack operations using Interface

```
interface Stack
{
    void push(int i);
    int pop();
}

class Stack1
{
    int stk[];
    Stack1()
    {
        stk = new int[10];
    }
}

public class StackDemo extends Stack1 implements Stack
{
    int tos;
    StackDemo()
    {
        tos=-1;
    }

    public void push(int i)
    {
        if(tos>9)
            System.out.println("Stack is full");
        else
            stk[++tos]=i;
    }

    public int pop()
    {
        if(tos<0)
        {
            System.out.println("Stack is empty");
            return 0;
        }
        else
            return stk[tos--];
    }

    public static void main(String[] args)
    {
        StackDemo s = new StackDemo();
    }
}
```

```
    for(int i=1; i<=9; i++)
        s.push(i);
    for(int i=0; i<10; i++)
        System.out.println(s.pop());
    }
}
```

Output:

```
9
8
7
6
5
4
3
2
1
Stack is empty
0
```

6. Program to demonstrate 'this' keyword

```
class ThisDemo
{
    int x,y;
    ThisDemo(int x,int y)
    {
        this.x=x;
        this.y=y;
    }

    void display()
    {
        System.out.println("x=" + x);
        System.out.println("y=" + y);
    }
}

public class Demo3
{
    public static void main(String[] args)
    {
        ThisDemo d = new ThisDemo(1,2);
        d.display();
    }
}
```

Output:

```
x=1
y=2
```

7.Program to demonstrate 'static' keyword

```
class StatDemo
{
    static int x;
    int y;
    static
    {
        x=4;
        System.out.println("x=" + x);
    }

    static void display()
    {
        x=x*2;
        StatDemo d = new StatDemo();
        d.y=10;
        System.out.println("y=" + d.y);
    }
}

class StaticDemo
{
    public static void main(String[] args)
    {
        StatDemo d1 = new StatDemo();
        d1.y=10;
        System.out.println("d1.y=" + d1.y);
        System.out.println("x=" + StatDemo.x);
        StatDemo.display();
    }
}
```

OUTPUT:

```
x=4
d1.y=10
x=4
y=10
```

8. Program to demonstrate Command Line Arguments

```
class Demo5
{
    public static void main(String[] args)
    {
        String str = args[0];
        int i = Integer.parseInt(args[1]);
        float f = Float.parseFloat(args[2]);
        System.out.println("Adding " + i + " and " + f + " gives " + (f+i));
        System.out.println("Multiplying " + i + " and " + f + " gives " + (f*i));
        System.out.println("String " + str + " length is " + str.length());
        System.out.println("For String " + str + " character at e is" + str.charAt(3));
    }
}
```

Output:

```
D:\SREE\RECORD>java Demo5 Hello 10 20.1
Adding 10 and 20.1 gives 30.1
Multiplying 10 and 20.1 gives 201.0
String Hello length is 5
For String Hello character at e is l
```

9. Program to demonstrate inner class

```
class Outer
{
    int ox;
    Outer()
    {
        ox = 10;
    }

    class inner
    {
        int ix;
        inner()
        {
            ox = 30;
            ix = 40;
        }

        void display()
        {
            System.out.println();
            System.out.println("Inner Class display");
            System.out.println("ox=" + ox);
            System.out.println("ix=" + ix);
        }
    }

    void displayOuter()
    {
        System.out.println("Outer class display");
        System.out.println("ox=" + ox);
        inner k = new inner();
        k.display();
    }

    public static void main(String[] args)
    {
        Outer l = new Outer();
        l.displayOuter();
    }
}
```

Output:

Outer class display

ox=10

Inner Class display

ox=30

ix=40

10. Program to demonstrate Inheritance

```
class Inher1
{
    int i;
    Inher1()
    {
        i = 10;
    }

    Inher1(int i)
    {
        this.i = i;
    }

    void show()
    {
        System.out.println("In Superclass i = " + i);
    }
}
```

```
public class Inher2 extends Inher1
{
    int k;
    Inher2()
    {
        i = 10;
        k = 20;
    }

    Inher2(int i, int k)
    {
        this.i = i;
        this.k = k;
    }

    void showK()
    {
        show();
        System.out.println("k=" + k);
    }

    public static void main(String[] args)
    {
        Inher2 i = new Inher2();
        Inher2 i1 = new Inher2(30,40);
        i.showK();
    }
}
```

```
    i1.showK();  
  }  
}
```

Output:

```
In Superclass i = 10  
k=20  
In Superclass i = 30  
k=40
```

11. Program to demonstrate 'super' keyword

```
class Figure
{
    int l,b;
    Figure()
    {
        l=10;
        b=5;
        System.out.println("Super class constructor");
    }

    Figure(int i, int j)
    {
        System.out.println("Super class Constructor");
        l=i;
        b=j;
    }

    void area()
    {
        System.out.println("There is no area for the figure");
    }
}

class Cube extends Figure
{
    int h;
    Cube()
    {
        l=5;
        b=7;
        h=9;
    }

    Cube(int l, int b, int h)
    {
        super(l,b);
        this.h=h;
    }

    void area()
    {
        System.out.println("area=" + (l*b));
    }

    void volume()
    {
```

```
        System.out.println("volume=" + (l*b*h));
    }
}
```

```
public class SuperDemo extends Figure
{
    void area()
    {
        System.out.println("Area of rectangle" + (l*b));
    }

    void perimeter()
    {
        System.out.println("Perimeter of rectangle" + (2*(l+b)));
    }

    SuperDemo(int l, int b)
    {
        super(l,b);
    }

    public static void main(String[] args)
    {
        Figure f = new Figure(30,40);
        Cube c = new Cube(40,50,60);
        SuperDemo s = new SuperDemo(30,20);
        Figure f1;

        f.area();
        c.area();
        s.area();
        c.volume();
        s.perimeter();

        f1=f;
        f1.area();
        f1=c;
        f1.area();
        f1=s;
        f1.area();
    }
}
```

Output:

Super class Constructor
Super class Constructor
Super class Constructor
There is no area for the figure
area=2000
Area of rectangle600
volume=120000
Perimeter of rectangle100
There is no area for the figure
area=2000
Area of rectangle600

12. Program to demonstrate Abstract class

```
abstract class Figure
{
    int l,b;
    abstract void area();
    void showlb()
    {
        System.out.println("l=" + l + "b=" + b);
    }
}

public class AbstractDemo extends Figure
{
    int h;
    AbstractDemo(int l,int b, int h)
    {
        this.l=l;
        this.b=b;
        this.h=h;
    }

    void area()
    {
        System.out.println("Area of cube "+ (l*b*h));
    }

    public static void main(String[] args)
    {
        AbstractDemo a = new AbstractDemo(10,20,30);
        a.area();
        a.showlbh();
    }

    void showlbh()
    {
        showlb();
        System.out.println("h=" + h);
    }
}
```

Output:

```
Area of cube 6000
l=10b=20
h=30
```

13. Program to demonstrate Packages

```
package mypack;

class Employee
{
    int eid;
    String name;
    double hra,da,sal,basic;
    Employee(int id,String n, double hr, double da)
    {
        eid=id;
        name=n;
        hra=hr;
        basic=3000.50;
        this.da=da;
        sal=50;
    }

    void calculate()
    {
        sal=hra+basic+da;
        System.out.println("empid: " + eid);
        System.out.println("Name: " + name);
        System.out.println("HRA: " + hra);
        System.out.println("DA: " + da);
        System.out.println("Basic: " + basic);
        System.out.println("Sal: " + sal);
    }

    public static void main(String[] argS)
    {
        Employee e = new Employee(1,"abc",30.1,3000);
        e.calculate();
    }
}
```

Output:

```
empid1
Nameabc
HRA: 30.1
DA: 3000.0
Basic: 3000.5
Sal: 6030.6
```

14. Program to demonstrate Interface

```
interface myintf
{
    int x=10;
    void call();
    void callMe(int x);
}

public class InterfaceDemo implements myintf
{
    public void call()
    {
        System.out.println("This is implemented method");
    }

    public void callMe(int x)
    {
        System.out.println("Second method is implemented" + x);
    }

    public static void main(String[] args)
    {
        InterfaceDemo d = new InterfaceDemo();
        d.call();
        d.callMe(4);
        System.out.println("x=" + x);
    }
}
```

Output:

```
This is implemented method
Second method is implemented4
x=10
```


15. Program to demonstrate Exception Handling

```
class ExcepDemo
{
    public static void main(String[] args)
    {
        int a=0;
        int d=30;

        try
        {
            System.out.println("a=" + a + " d=" + d);
            d=d/a;
            System.out.println("This will not be executed");
        }

        catch(ArithmeticException e)
        {
            System.out.println("Exception raised "+ e);
            a=5;
            d=d/a;
            System.out.println("This will be executed");
        }

        System.out.println("This also will be executed");
    }
}
```

Output:

```
a=0 d=30
Exception raised java.lang.ArithmeticException: / by zero
This will be executed
This also will be executed
```

16. Program to demonstrate MultiCatch

```
class Multicatch
{
    public static void main(String[] args)
    {
        int a=args.length;
        int d=40;
        int c[]={1,2,3,4};

        try
        {
            if(a==0)
                d=d/a;
            if(a==1)
                d=d/(a-a);
            if(a==2)
                System.out.println("Value of c[7] = " + c[7]);
            System.out.println("This will never be executed");
        }

        catch(ArithmeticException e)
        {
            System.out.println(e);
        }

        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println("Array problem" + e);
        }

        catch(Exception e)
        {
            System.out.println("All kinds of exceptions " + e);
        }

        System.out.println("This will be executed");
    }
}
```

Output:

```
java.lang.ArithmeticException: / by zero
This will be executed
```

17. Program to demonstrate 'throws' clause and finally block

```
class ThrowDemo
{
    void throwDemometh()throws Exception
    {
        try
        {
            System.out.println("This is within ThrowDemo");
            throw new NullPointerException("From my code");
        }

        catch(Exception e)
        {
            System.out.println(e);
        }
    }

    public static void main(String[] args)
    {
        ThrowDemo d1 = new ThrowDemo();
        try
        {
            d1.throwDemometh();
        }

        catch(Exception e)
        {
            System.out.println("Recaught" + e);
        }

        finally
        {
            System.out.println("In finally");
        }
    }
}
```

OUTPUT:

```
This is within ThrowDemo
java.lang.NullPointerException: From my code
In finally
```

18. Program to demonstrate User Defined Exceptions

```
class BankExcep extends Exception
{
    String str;
    BankExcep(String str)
    {
        this.str=str;
    }

    public String toString()
    {
        return str+ "Exception raised as amt<500";
    }
}

public class UserExcep
{
    double amt;
    UserExcep()
    {
        amt=500;
    }

    void withdraw(double d)throws BankExcep
    {
        if((amt-d)<500)
            throw new BankExcep("In Withdraw");
        else
        {
            amt=amt-d;
            System.out.println("Amount Withdrawn is " + d);
        }
    }

    void deposit(double dep)
    {
        amt=amt+dep;
        System.out.println("Amount deposited. Updated amount is " + amt);
    }

    public static void main(String[] args)
    {
        UserExcep d = new UserExcep();
        d.deposit(1000);

        try
        {
```

```
        d.withdraw(500);
        d.withdraw(2000);
    }

    catch(BankExcep e)
    {
        System.out.println(e);
    }
}
}
```

Output:

Amount deposited. Updated amount is 1500.0

Amount Withdrawn is 500.0

In WithdrawException raised as amt<500

19. Program to demonstrate Arrays

```
class ArrDemo
{
    public static void main(String[] args)
    {
        int a[][] = new int[3][3];
        int i,j,k=0;

        for(i=0; i<3; i++)
            for(j=0; j<3; j++)
            {
                a[i][j] = k;
                k++;
            }

        for(i=0; i<3; i++)
        {
            for(j=0; j<3; j++)
                System.out.print(a[i][j] + " ");
            System.out.println();
        }
    }
}
```

OUTPUT:

```
0 1 2
3 4 5
6 7 8
```

THREADS

/* Program to demonstrate Thread */

```
class ThreadDemo
{
    public static void main(String[] args)
    {
        System.out.println("Within main thread");
        try
        {
            Thread t = Thread.currentThread();
            System.out.println(t);
            t.setName("Demo Thread");
            System.out.println(t);
        }

        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

Output:

```
Within main thread
Thread[main,5,main]
Thread[Demo Thread,5,main]
```



```
/* Program to demonstrate Threads(Main and Child Threads) */
```

```
class Childth extends Thread
{
String str;
Childth(String str)
{
super(str);
this.str=str;
start();
}

public void run()
{
for(int i=65; i<91; i++)
{
System.out.println((char)i);
try
{
Thread.sleep(500);
}

catch(InterruptedExcepion e)
{
System.out.println(e);
}
}
}

public class MainThread
{
public static void main(String[] argS)
{
System.out.println("In Main");
Childth c = new Childth("child");

for(int i=0; i<25; i++)
{
System.out.println("Main" + i);
try
{
Thread.sleep(1000);
}

catch(Exception e)
{
```

```
        System.out.println(e);
    }
    System.out.println("Main existing");
}
}
```

Output:

In Main
Main0
A
B
Main existing
Main1
C
D
Main existing
E
Main2
F
Main existing
Main3
G
H
Main existing
Main4
I
J
Main existing

```
/* Program to demonstrate Thread using Runnable Interface */
```

```
class chrn implements Runnable
{
    Thread t;
    chrn(String str)
    {
        t=new Thread(this,str);
        t.start();
    }

    public void run()
    {
        for(int i=10; i<36; i++)
        {
            System.out.println((char)i + "child");
            try
            {
                Thread.sleep(500);
            }

            catch(InterruptedException e)
            {
                System.out.println(e);
            }
        }
    }
}
```

```
public class RunnableDemo
{
    public static void main(String[] args)
    {
        System.out.println("Entered main");
        chrn c = new chrn("child");

        for(int i=0; i<25; i++)
        {
            System.out.println("Main: " + i);
            try
            {
                Thread.sleep(1000);
            }

            catch(InterruptedException e)
            {
                System.out.println(e);
            }
        }
    }
}
```

```
    }  
  }  
}
```

Output:

Main: 0

child

♂child

Main: 1

♀child

child

Main: 2

♪child

☀child

Main: 3

▶child

◀child

Main: 4

```
/* Program to demonstrate Join and isalive */
```

```
class Childth implements Runnable
```

```
{  
    int x;  
    Thread t;  
    Childth(String str, int x)  
    {  
        this.x=x;  
        t=new Thread(this,str);  
        t.start();  
    }  
}
```

```
public void run()
```

```
{  
    if(x==1)  
        world();  
    if(x==2)  
        hello();  
    if(x==3)  
        hai();  
}
```

```
void world()
```

```
{  
    try  
    {  
        for(int i=0; i<5; i++)  
        {  
            System.out.println("world");  
            Thread.sleep(100);  
        }  
    }  
}
```

```
catch(InterruptedException e)
```

```
{  
    System.out.println(e);  
}  
}
```

```
void hello()
```

```
{  
    try  
    {  
        for(int i=0; i<5; i++)  
        {  
            System.out.println("hello");  
            Thread.sleep(100);  
        }  
    }  
}
```

```

    }
}

catch(InterruptedException e)
{
    System.out.println(e);
}

}

void hai()
{
    try
    {
        for(int i=0; i<5; i++)
        {
            System.out.println("hai");
            Thread.sleep(50);
        }
    }

    catch(InterruptedException e)
    {
        System.out.println(e);
    }
}

}

public class JoinDemo
{
    public static void main(String[] argS)
    {
        Childth c1 = new Childth("One",1);
        Childth c2 = new Childth("Two",2);
        Childth c3 = new Childth("Three", 3);
        System.out.println("c1 is alive: " + c1.t.isAlive());
        System.out.println("c2 is alive: " + c2.t.isAlive());
        System.out.println("c3 is alive: " + c3.t.isAlive());

        try
        {
            c1.t.join();
            c2.t.join();
            c3.t.join();
        }

        catch(Exception e)
        {
            System.out.println(e);

```

```
    }  
  
    System.out.println("Now c1 is alive: " + c1.t.isAlive());  
    System.out.println("Now c2 is alive: " + c2.t.isAlive());  
    System.out.println("Now c3 is alive: " + c3.t.isAlive());  
    }  
}
```

Output:

```
c1 is alive: true  
c2 is alive: true  
c3 is alive: true  
world  
hello  
hai  
hai  
world  
hello  
hai  
hai  
world  
hello  
hai  
world  
hello  
world  
hello  
Now c1 is alive: false  
Now c2 is alive: false  
Now c3 is alive: false
```

```
/* Program to demonstrate Synchronization */
```

```
class Shared
{
    synchronized public void call(String str)
    {
        System.out.print "[" + str;
        try
        {
            Thread.sleep(1000);
        }

        catch (InterruptedException e)
        {
            System.out.println(e);
        }

        System.out.println("]");
    }
}
```

```
class Threads extends Thread
{
    String str;
    Shared q;

    Threads(Shared s, String str)
    {
        this.str = str;
        q = s;
        start();
    }

    public void run()
    {
        q.call(str);
    }
}
```

```
class SyncDemo
{
    public static void main(String[] argS)
    {
        Shared s = new Shared();
        Threads t1 = new Threads(s, "Hello");
        Threads t2 = new Threads(s, "World");
    }
}
```



```
try
{
    t1.join();
    t2.join();
}

catch(Exception e)
{
    System.out.println(e);
}
}
```

Output:

```
[Hello]
[World]
```

/* Program to demonstrate Interthread Communication */

```
class Q
{
    int x;
    boolean vs = false;
    synchronized int get()
    {
        if(!vs)
        {
            try
            {
                wait();
            }

            catch(InterruptedException e)
            {
                System.out.println(e);
            }
        }
        System.out.println("got: " + x);
        notify();
        vs=false;
        return x;
    }

    synchronized void put(int x)
    {
        if(vs)
        {
            try
            {
                wait();
            }
            catch(Exception e)
            {
                System.out.println(e);
            }
        }

        this.x=x;
        System.out.println("put:x " + x);
        notify();
        vs=true;
    }
}
```

```
class Consumer extends Thread
```

```
{  
    Q q;  
    Consumer(Q q)  
    {  
        this.q=q;  
        start();  
    }  
  
    public void run()  
    {  
        while(true)  
        {  
            q.get();  
        }  
    }  
}
```

```
class Producer extends Thread
```

```
{  
    Q q;  
    Producer(Q q)  
    {  
        this.q=q;  
        start();  
    }  
  
    public void run()  
    {  
        int i=0;  
        while(true)  
        {  
            q.put(i++);  
        }  
    }  
}
```

```
class InterThread
```

```
{  
    public static void main(String[] args)  
    {  
        Q q = new Q();  
        Consumer c = new Consumer(q);  
        Producer p = new Producer(q);  
        try  
        {  
            c.join();  
            p.join();  
        }  
    }  
}
```

```
    }  
    catch(Exception e)  
    {  
        System.out.println(e);  
    }  
}  
}
```

Output:

```
got: 527  
put:x 528  
got: 528  
put:x 529  
got: 529  
put:x 530  
got: 530  
put:x 531  
got: 531  
put:x 532  
got: 532  
put:x 533  
got: 533  
put:x 534  
got: 534  
put:x 535  
got: 535  
put:x 536  
got: 536  
put:x 537  
got: 537  
put:x 538  
got: 538  
put:x 539  
got: 539
```

APPLETS

/* Program to demonstrate Applet */

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

public class FirstApp extends Applet
{
    int st=0, in=0, so=0, p=0, d=0;
    public void init()
    {
        System.out.println("In Init" + (in++));
    }

    public void start()
    {
        System.out.println("In Start" + (st++));
    }

    public void paint(Graphics g)
    {
        System.out.println("In paint" + (p++));
    }

    public void stop()
    {
        System.out.println("In Stop" + (so++));
    }

    public void destroy()
    {
        System.out.println("In destroy" + (d++));
    }
}
```

<HTML>

<applet code="FirstApp.class" height=400 width=400>

</applet>

</HTML>

```
/* Program to demonstrate Methods in Graphics class */
```

```
import java.applet.*;
```

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

```
public class GraphicsDemo extends Applet  
{  
    public void paint(Graphics g)  
    {  
        g.setColor(Color.blue);  
        g.drawString("First applet Program" ,10,10);  
        g.setColor(Color.red);  
        g.drawLine(10,20,100,40);  
        g.setColor(Color.blue);  
        g.fillRect(10,60,100,100);  
    }  
}
```

```
<HTML>
```

```
<applet code="GraphicsDemo.class" height=400 width=400>
```

```
</applet>
```

```
</HTML>
```

OUTPUT:



Applet started.

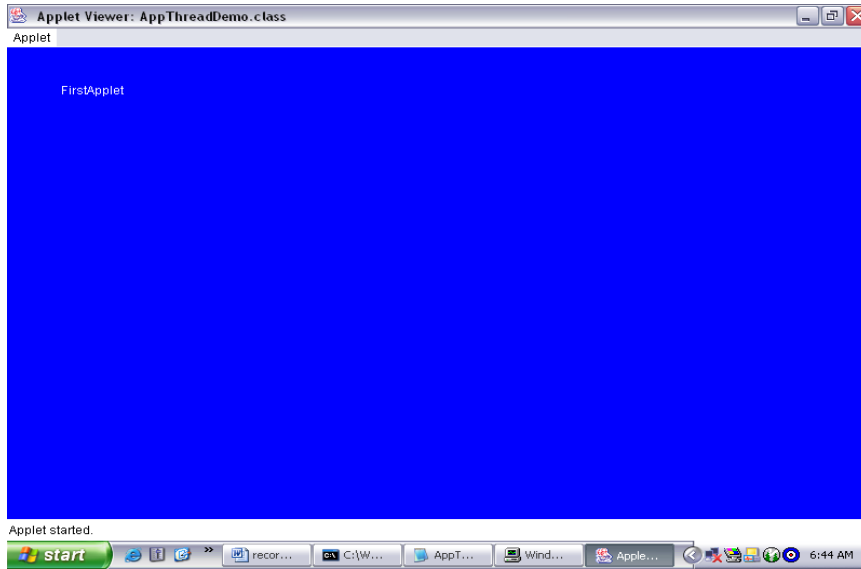
```
/* Program to demonstrate Threads using Applets */
```

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

```
public class AppThreadDemo extends Applet  
{  
    public void paint(Graphics g)  
    {  
        setBackground(Color.blue);  
        setForeground(Color.white);  
        g.drawString("FirstApplet",50,50);  
        AppletContext ac = getAppletContext();  
  
        try  
        {  
            Thread.sleep(5000);  
            ac.showDocument(new URL(getDocumentBase(),"FirstApp.html"));  
        }  
  
        catch(Exception e)  
        {  
            System.out.println(e);  
        }  
    }  
}
```

```
<HTML>  
<applet code="AppThreadDemo.class" height=400 width=400>  
</applet>  
</HTML>
```

OUTPUT:



/* Program to demonstrate Image within Applet*/

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

public class DrawImageApp extends Applet
{
    Image i;
    public void start()
    {
        i = getImage(getDocumentBase(),"2002106ke.jpg");
    }

    public void paint(Graphics g)
    {
        g.drawImage(i,0,0,this);
    }
}
```

```
<HTML>
<applet code="DrawImageApp.class" height=400 width=400>
</applet>
</HTML>
```

OUTPUT:



Applet started.

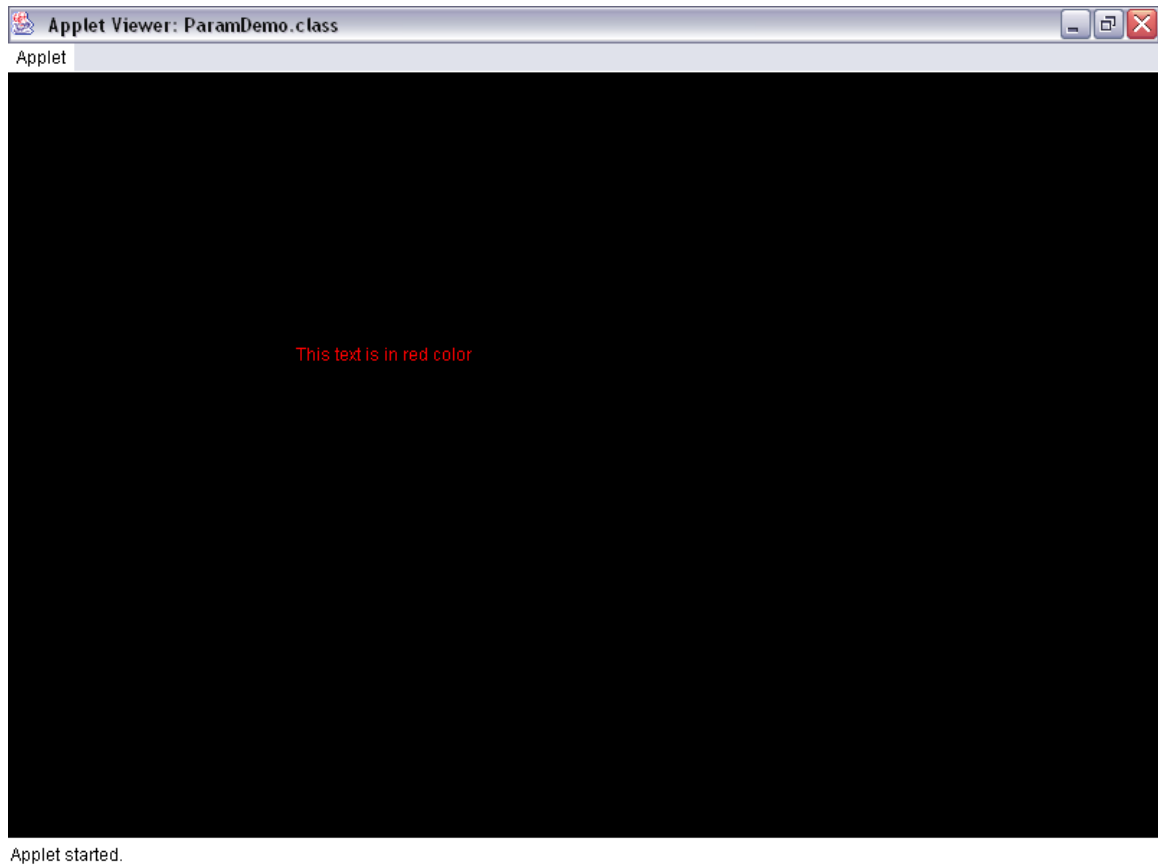
/* Program to demonstrate Param Tag */

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

public class ParamDemo extends Applet
{
    String st="This text is in red color";
    public void paint(Graphics g)
    {
        String fg = getParameter("fgcolor");
        String bg = getParameter("bgcolor");
        if(fg.equals("red"))
            setForeground(Color.red);
        if(bg.equals("black"))
            setBackground(Color.black);
        g.drawString(st,200,200);
    }
}
```

```
<HTML>
<applet code="ParamDemo.class" height=400 width=400>
  <param name="fgcolor" value="red">
  <param name="bgcolor" value="black">
</applet>
</HTML>
```

OUTPUT:



```
/* Program to demonstrate Font Class*/
```

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

```
public class FontDemo extends Applet  
{  
    public void start()  
    {  
        Font f=new Font("TimesNewRoman",Font.BOLD,12);  
        setFont(f);  
    }  
  
    public void paint(Graphics g)  
    {  
        Color c = new Color(32,40,99);  
        g.setColor(c);  
        GraphicsEnvironment ge =  
GraphicsEnvironment.getLocalGraphicsEnvironment();  
        String str[] = ge.getAvailableFontFamilyNames();  
        int x=10, y=20;  
  
        for(int i=0; i<str.length; i++)  
        {  
            g.drawString(str[i],x,y);  
            y=y+20;  
        }  
    }  
}
```

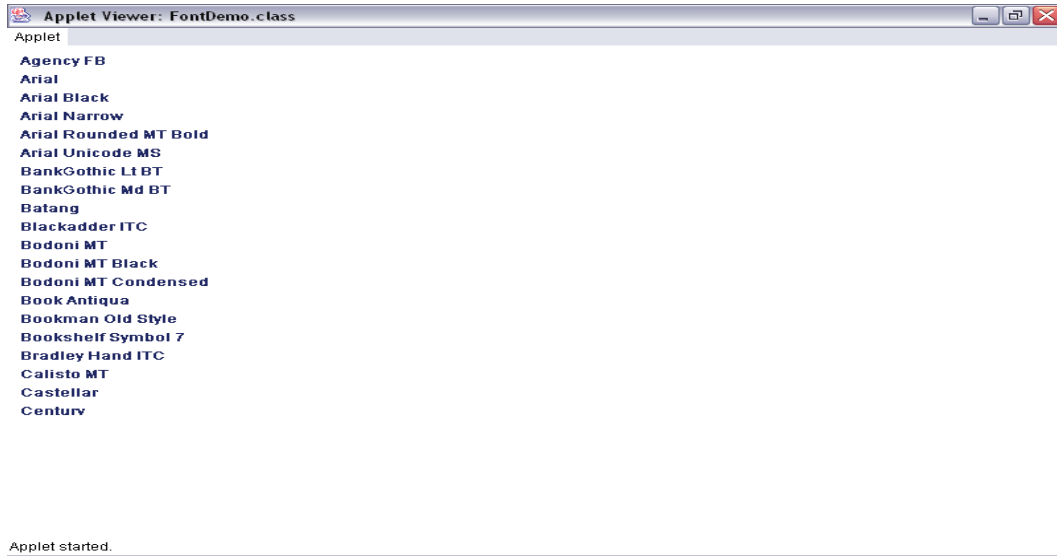
```
<HTML>
```

```
<applet code="FontDemo.class" height=400 width=400>
```

```
</applet>
```

```
</HTML>
```

OUTPUT:



```
/* Program to demonstrate Digital Clock */
```

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

```
public class TimeDemo extends Applet implements Runnable  
{  
    Thread t;  
    Calendar c;  
    String str;  
  
    public void init()  
    {  
        t=new Thread(this,"demo");  
        t.start();  
    }  
  
    public void run()  
    {  
        while(true)  
        {  
            c=Calendar.getInstance();  
            str=c.get(Calendar.HOUR)+ ":" +  
c.get(Calendar.MINUTE)+ ":"+c.get(Calendar.SECOND);  
            repaint();  
        }  
    }  
  
    public void paint(Graphics g)  
    {  
        g.drawString(str,10,50);  
    }  
}
```

```
<HTML>
```

```
<applet code="TimeDemo.class" height=400 width=400>
```

```
</applet>
```

```
</HTML>
```


OUTPUT:



7:1:25

Applet started.

AWT AND EVENT HANDLING

```
/* Program to create Frame*/
```

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

```
class MyFrame extends Frame
```

```
{  
    MyFrame(String str)  
    {  
        super(str);  
        setSize(300,300);  
        setVisible(true);  
    }  
}
```

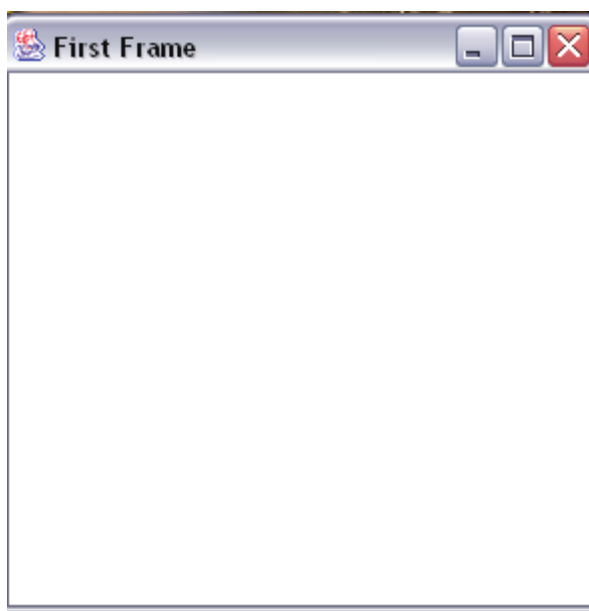
```
public static void main(String[] args)
```

```
{  
    new MyFrame("First Frame");  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
    return super.handleEvent(e);  
}
```

OUTPUT:



/* Program to demonstrate Frame with Labels and Buttons */

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

```
class FrameDemo extends Frame
```

```
{  
    Label l1,l2,l3;  
    Button b1,b2;  
    FrameDemo()  
    {  
        super("Second Frame");  
        l1=new Label("ONE");  
        l2=new Label("TWO");  
        l3=new Label("THREE",Label.CENTER);  
        l2.setBackground(Color.red);  
        l3.setBackground(Color.red);  
        b1=new Button("OK");  
        b2=new Button("Cancel");  
        setLayout(new FlowLayout());  
        add(l1);  
        add(l2);  
        add(l3);  
        add(b1);  
        add(b2);  
        setSize(300,300);  
        setVisible(true);  
    }  
}
```

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

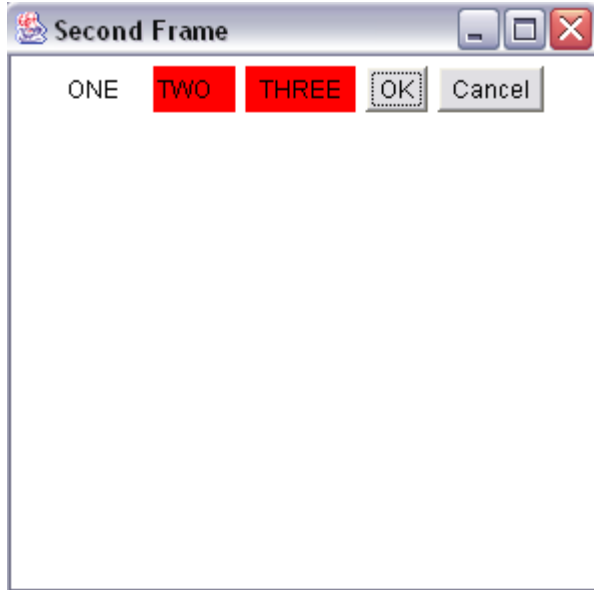
```
{  
    new FrameDemo();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
    return super.handleEvent(e);  
}
```

```
}
```

OUTPUT:



```
/* Program to demonstrate Frames using button , label, textfield.*/
```

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

```
class FrameDemo5 extends Frame
```

```
{  
    Label l,u,p,d;  
    Button o,c;  
    TextField tf1,tf2;  
    TextArea ta;
```

```
    FrameDemo5()
```

```
{  
    u=new Label("Username   ");  
    p=new Label("Password   ");  
    d=new Label("Details");  
    o=new Button("OK");  
    c=new Button("Cancel");  
    tf1=new TextField(30);  
    tf2=new TextField(30);  
    ta=new TextArea("Enter your Address",4,5);  
    tf2.setEchoChar('*');  
    setLayout(new FlowLayout());
```

```
    add(u);  
    add(tf1);  
    add(p);  
    add(tf2);  
    add(d);  
    add(ta);  
    add(o);  
    add(c);  
    setSize(400,400);  
    setVisible(true);  
}
```

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

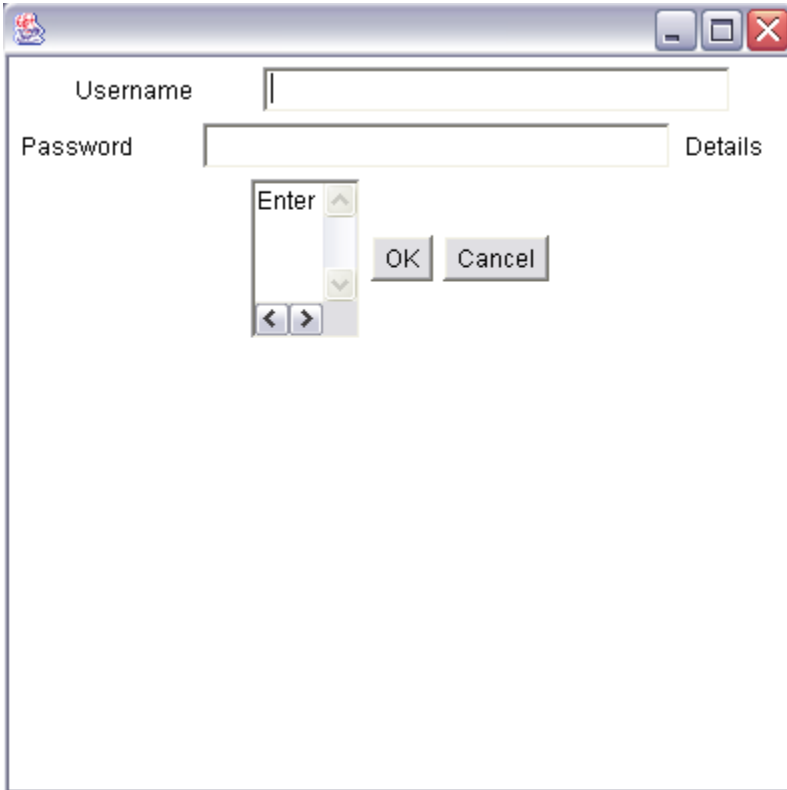
```
{  
    new FrameDemo5();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
    return super.handleEvent(e);
```

```
}  
}
```

OUTPUT:



```
/* Program to demonstrate Frame with CheckBox and CheckBoxGroup */
```

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

```
class FrameDemo3 extends Frame
```

```
{  
    Label l1,l2,l3;  
    Checkbox cb1,cb2,cb3,cb4;  
    CheckboxGroup cbg;  
    Choice c;
```

```
FrameDemo3()
```

```
{  
    super("Hello");  
    l1=new Label("QUALIFICATION");  
    l2=new Label("GENDER");  
    l3=new Label("SUBJECT");  
    cb1=new Checkbox("MCA");  
    cb2=new Checkbox("BSC");  
    cbg=new CheckboxGroup();  
    cb3=new Checkbox("M",true,cbg);  
    cb4=new Checkbox("F",false,cbg);  
    c=new Choice();  
    c.add("Java");  
    c.add("C++");  
    c.add("C");  
    c.add("Cobol");  
    setLayout(new FlowLayout());  
    add(l1);  
    add(cb1);  
    add(cb2);  
    add(l2);  
    add(cb3);  
    add(cb4);  
    add(l3);  
    add(c);  
    setSize(400,400);  
    setVisible(true);  
}
```

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

```
{  
    new FrameDemo3();  
}
```

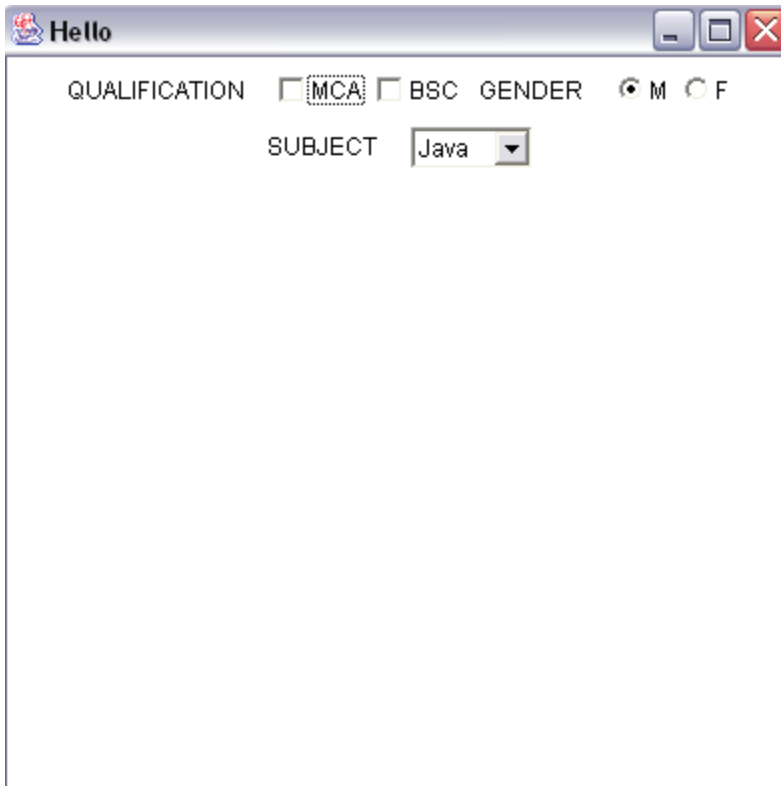
```
public boolean handleEvent(Event e)
```

```
{
```



```
if(e.id==Event.WINDOW_DESTROY)
{
    System.exit(0);
}
return super.handleEvent(e);
}
}
```

OUTPUT:



The screenshot shows a Java Swing window titled "Hello". The window contains a form with the following elements:

- QUALIFICATION: MCA BSC
- GENDER: M F
- SUBJECT:

```
/* Program to demonstrate Complex Frame( Checkboxes) */
```

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

```
class FrameDemo4 extends Frame
```

```
{  
    Label la;  
    Checkbox o,w,d,l,s,j,cp,c,p;  
    CheckboxGroup cbg1,cbg2;
```

```
    FrameDemo4()
```

```
{  
    super("Hello");  
    la=new Label("Skill Set");  
    o=new Checkbox("O.S");  
    s=new Checkbox("Software");  
    cbg1=new CheckboxGroup();  
    w=new Checkbox("Windows",true,cbg1);  
    d=new Checkbox("Dos",false,cbg1);  
    l=new Checkbox("Linux",false,cbg1);  
    p=new Checkbox("Unix",false,cbg1);  
    cbg2=new CheckboxGroup();  
    j=new Checkbox("Java",true,cbg2);  
    cp=new Checkbox("C++", false,cbg2);  
    c=new Checkbox("Cobol",false,cbg2);
```

```
    setLayout(new FlowLayout());  
    add(la);  
    add(o);  
    add(w);  
    add(d);  
    add(l);  
    add(p);  
    add(s);  
    add(j);  
    add(cp);  
    add(c);  
    setSize(400,400);  
    setVisible(true);  
}
```

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

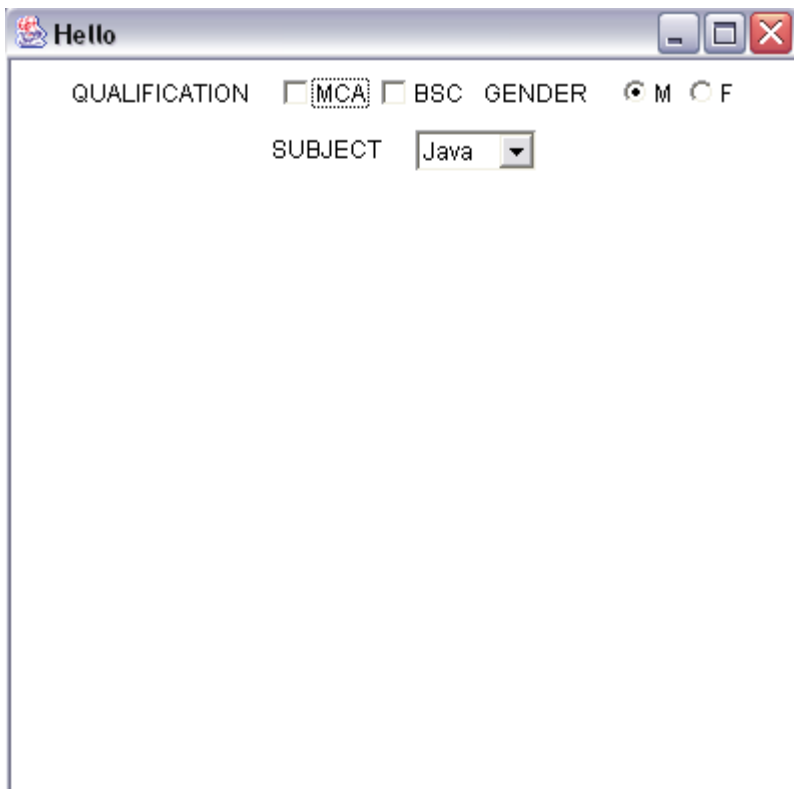
```
{  
    new FrameDemo4();  
}
```

```
public boolean handleEvent(Event e)
```

```
{
```

```
if(e.id==Event.WINDOW_DESTROY)
{
    System.exit(0);
}
return super.handleEvent(e);
}
}
```

OUTPUT:



The screenshot shows a Java Swing window titled "Hello". The window contains a form with the following elements:

- QUALIFICATION: MCA BSC
- GENDER: M F
- SUBJECT:

```
/* Program to demonstrate DialogBox*/
```

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

```
class DialogDemo extends Dialog
{
    Label l;
    Button b1,b2;
    DialogDemo(Frame f, String title)
    {
        super(f,title);
        l=new Label("Illegal Operation");
        b1=new Button("Ok");
        b2=new Button("Cancel");
        setLayout(new FlowLayout());
        add(l);
        add(b1);
        add(b2);
        setSize(400,400);
    }
}
```

```
class FramesDemo extends Frame
{
    Scrollbar vs,hs;
    FramesDemo()
    {
        vs=new Scrollbar(Scrollbar.VERTICAL);
        hs=new Scrollbar(Scrollbar.HORIZONTAL);
        setLayout(new FlowLayout());
        add(vs);
        add(hs);
        setVisible(true);
        setSize(500,500);
        try
        {
            Thread.sleep(100);
        }

        catch(Exception e)
        {
            System.out.println(e);
        }

        DialogDemo d = new DialogDemo(this,"Alert");
        d.setVisible(true);
        try
```

```
{
    Thread.sleep(1000);
}

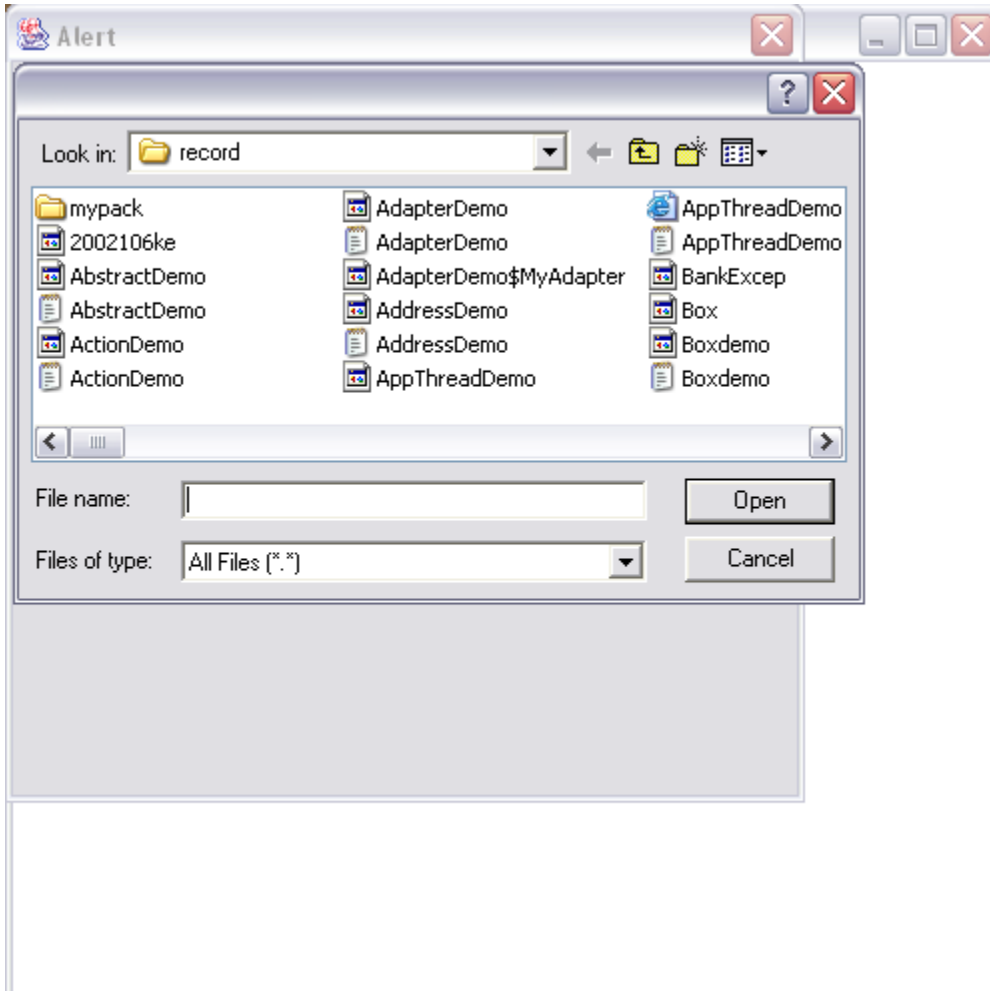
catch(Exception e)
{
    System.out.println(e);
}

FileDialog fd = new FileDialog(this);
fd.setVisible(true);
}

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

public boolean handleEvent(Event e)
{
    if(e.id==Event.WINDOW_DESTROY)
    {
        System.exit(0);
    }
    return super.handleEvent(e);
}
}
```

OUTPUT:



```
/* Program to demonstrate Menus*/
```

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

```
class MenuDemo extends Frame
```

```
{  
    Menu f,e,s;  
    MenuItem o,c,cu,co,u,l;  
    CheckboxMenuItem fo;  
    MenuBar mb;  
    MenuDemo()  
    {  
        f=new Menu("File");  
        e=new Menu("Edit");  
        s=new Menu("Select");  
        o=new MenuItem("Open");  
        c=new MenuItem("Close");  
        cu=new MenuItem("Cut");  
        co=new MenuItem("Copy");  
        u=new MenuItem("UpperCase");  
        l=new MenuItem("LowerCase");  
        fo=new CheckboxMenuItem("Font");  
        f.add(o);  
        f.add(c);  
        e.add(cu);  
        e.add(co);  
        s.add(u);  
        s.add(l);  
        e.add(s);  
        mb=new MenuBar();  
        mb.add(f);  
        mb.add(e);  
        setMenuBar(mb);  
        setVisible(true);  
        setSize(300,300);  
    }  
}
```

```
public static void main(String[] argS)
```

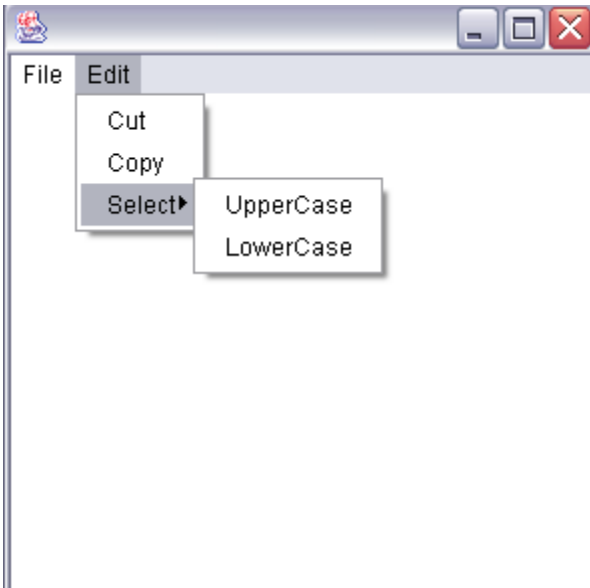
```
{  
    new MenuDemo();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
}
```

```
    return super.handleEvent(e);  
  }  
}
```

OUTPUT:




```
/* Program to demonstrate Border Layout */
```

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

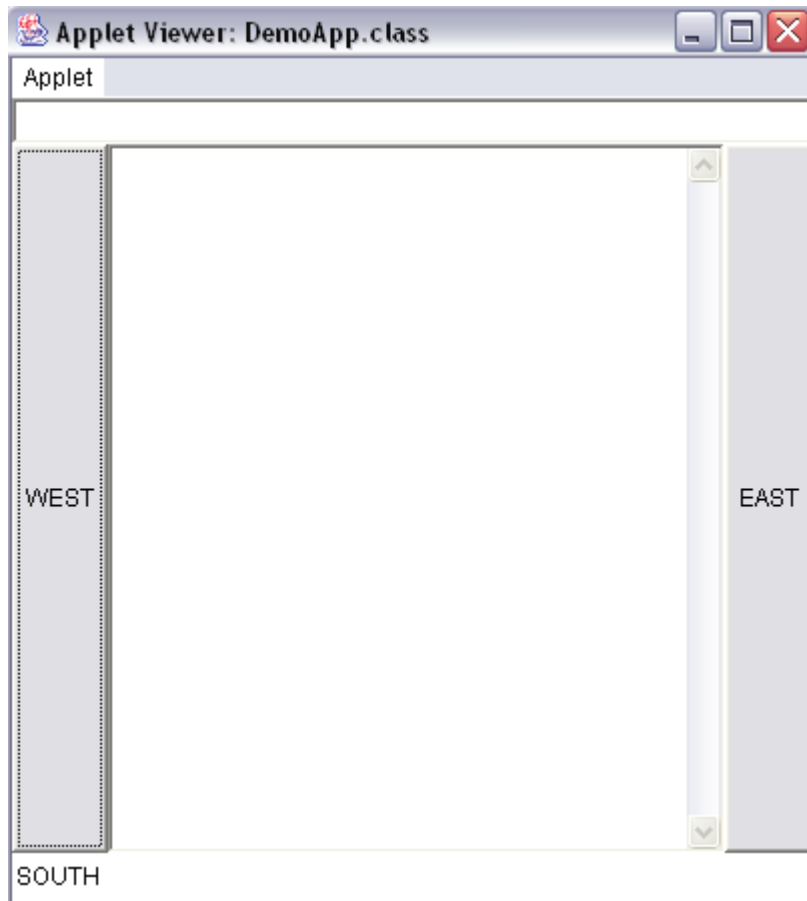
public class DemoApp extends Applet
{
    Button b1,b2;
    Label l1;
    TextField tf;
    TextArea ta;

    public void init()
    {
        b1=new Button("WEST");
        b2=new Button("EAST");
        l1=new Label("SOUTH");
        tf=new TextField(20);
        ta=new TextArea(5,4);
        setLayout(new BorderLayout());
    }

    public void start()
    {
        add(b1,BorderLayout.WEST);
        add(b2,BorderLayout.EAST);
        add(l1,BorderLayout.SOUTH);
        add(tf,BorderLayout.NORTH);
        add(ta);
    }
}
```

```
<HTML>
<applet code="DemoApp.class" height=400 width=400>
</applet>
</HTML>
```

OUTPUT:



```
/* Program to demonstrate Grid Layout */
```

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

```
class GridDemo extends Frame
```

```
{  
    Button b[];  
    GridDemo()  
    {  
        b=new Button[16];  
        setLayout(new GridLayout(4,4));  
  
        for(int i=0; i<16; i++)  
        {  
            b[i]=new Button((i+1)+" ");  
            add(b[i]);  
        }  
  
        setVisible(true);  
        setSize(300,300);  
    }  
}
```

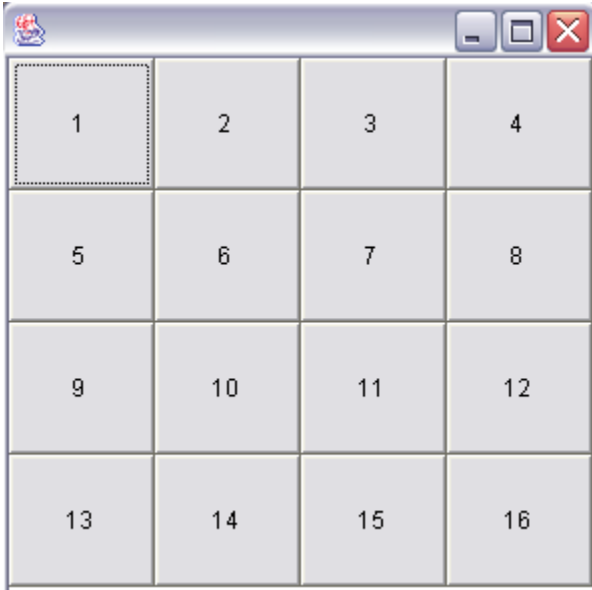
```
public static void main(String[] args)
```

```
{  
    new GridDemo();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
    return super.handleEvent(e);  
}
```

OUTPUT:



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

```
/* Program to demonstrate Null Layout */
```

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

```
class NullDemo extends Frame
```

```
{  
    Label l1,l2;  
    Button o,c;  
    TextField tf1,tf2;  
    NullDemo()  
    {  
        super("NULL LAYOUT DEMO");  
        l1=new Label("Username");  
        l2=new Label("Password");  
        o=new Button("Ok");  
        c=new Button("Cancel");  
        tf1=new TextField(30);  
        tf2=new TextField(30);  
        setLayout(null);  
        o.setBounds(20,120,30,20);  
        c.setBounds(90,120,50,20);  
        l1.setBounds(10,40,60,20);  
        l2.setBounds(10,80,60,20);  
        tf1.setBounds(80,40,60,20);  
        tf2.setBounds(80,80,60,20);  
        add(l1);  
        add(tf1);  
        add(l2);  
        add(tf2);  
        tf2.setEchoChar('*');  
        add(o);  
        add(c);  
        setVisible(true);  
        setSize(300,300);  
    }  
}
```

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

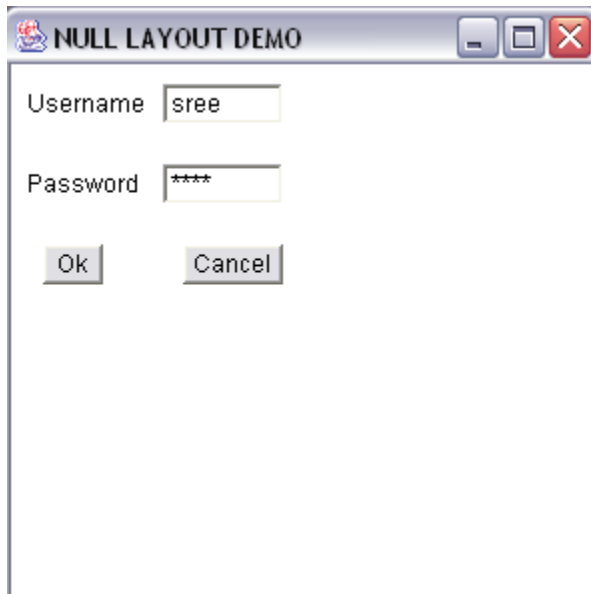
```
{  
    new NullDemo();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
}
```

```
    return super.handleEvent(e);  
  }  
}
```

OUTPUT:



```
/* Program to demonstrate Implementation of Action Listener*/
```

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

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

```
class ActionDemo extends Frame implements ActionListener
```

```
{  
    Button b1,b2;  
    Label l;  
    ActionDemo()  
    {  
        b1=new Button("One");  
        b2=new Button("Two");  
        l=new Label(" ");  
        add(b1,BorderLayout.NORTH);  
        add(b2,BorderLayout.SOUTH);  
        add(l);  
        setSize(300,300);  
        setVisible(true);  
        b1.addActionListener(this);  
        b2.addActionListener(this);  
    }  
}
```

```
public void actionPerformed(ActionEvent ae)
```

```
{  
    if(ae.getSource()==b1)  
    {  
        l.setText("One");  
    }  
}
```

```
    if(ae.getSource()==b2)
```

```
    {  
        l.setText("Two");  
    }  
}
```

```
public static void main(String[] argS)
```

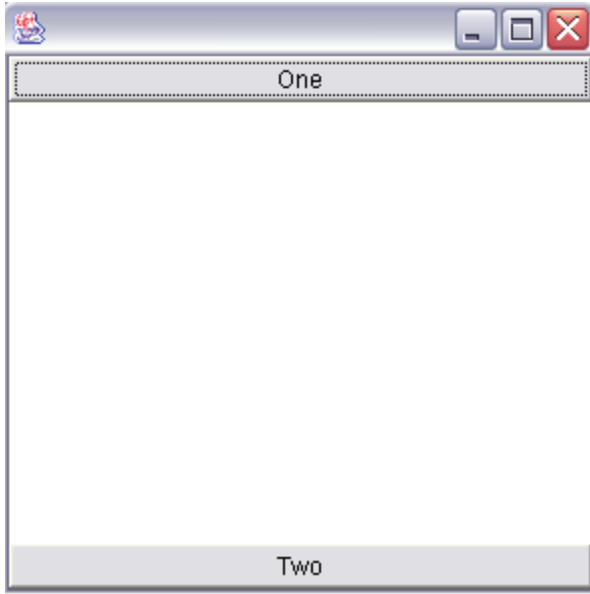
```
{  
    new ActionDemo();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
    return super.handleEvent(e);  
}
```

```
}  
}
```

OUTPUT:




```
/* Program to graphs using Action Listener */
```

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

```
public class GraphDemo extends Applet implements ActionListener
```

```
{  
    Button o,s,l;  
    int k;
```

```
    public void start()
```

```
    {  
        o=new Button("Oval");  
        s=new Button("Square");  
        l=new Button("Line");  
        add(o);  
        add(s);  
        add(l);  
        k=0;  
        o.addActionListener(this);  
        s.addActionListener(this);  
        l.addActionListener(this);  
    }
```

```
    public void actionPerformed(ActionEvent e)
```

```
    {  
        if(e.getSource()==o)  
        {  
            k=1;  
            repaint();  
        }
```

```
        if(e.getSource()==s)
```

```
        {  
            k=2;  
            repaint();  
        }
```

```
        if(e.getSource()==l)
```

```
        {  
            k=3;  
            repaint();  
        }
```

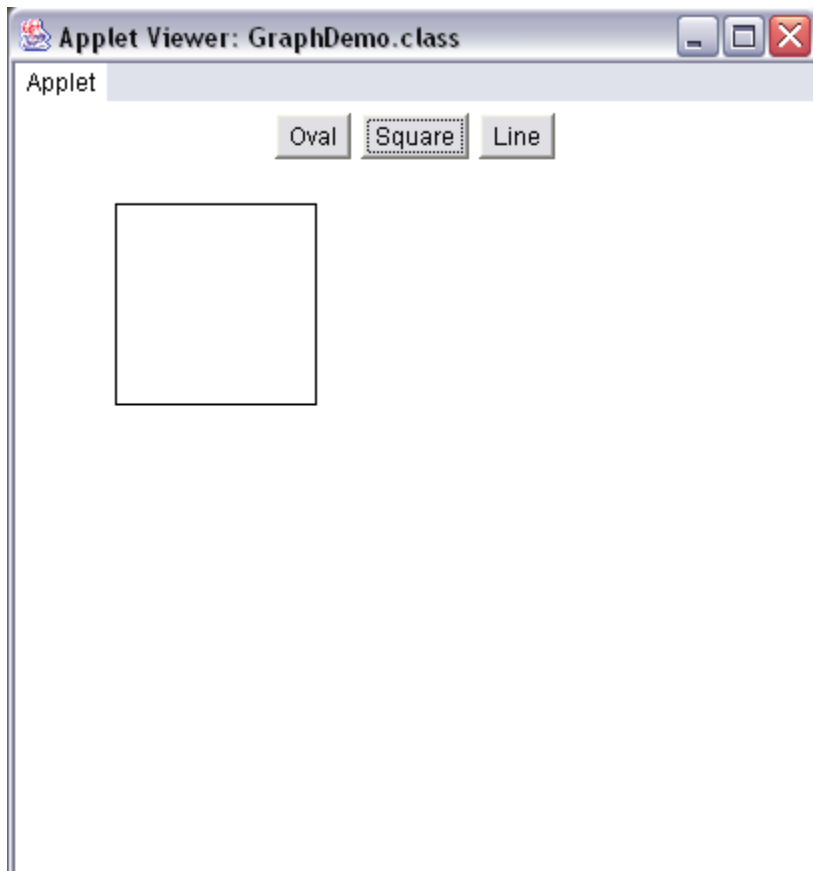
```
    }
```

```
    public void paint(Graphics g)
```

```
    {
```

```
if(k==1)
    g.drawOval(10,20,50,60);
if(k==2)
    g.drawRect(50,50,100,100);
if(k==3)
    g.drawLine(80,20,160,90);
}
}
```

OUTPUT:



```
/* Program to demonstrate implementation of ItemListener */
```

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

```
public class ItemDemo extends Frame implements ItemListener
```

```
{  
    Choice c;  
    Panel p;  
    ItemDemo()  
    {  
        c=new Choice();  
        c.add("Red");  
        c.add("Green");  
        c.add("Blue");  
        c.add("Pink");  
        p=new Panel();  
        setLayout(new GridLayout(1,2));  
        p.add(c);  
        add(p);  
        c.addItemListener(this);  
        setVisible(true);  
        setSize(300,300);  
    }  
}
```

```
public void itemStateChanged(ItemEvent e)
```

```
{  
    if(c.getSelectedItem().equals("Red"))  
        p.setBackground(Color.red);  
    if(c.getSelectedItem().equals("Green"))  
        p.setBackground(Color.green);  
    if(c.getSelectedItem().equals("Blue"))  
        p.setBackground(Color.blue);  
    if(c.getSelectedItem().equals("Pink"))  
        p.setBackground(Color.pink);  
}
```

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

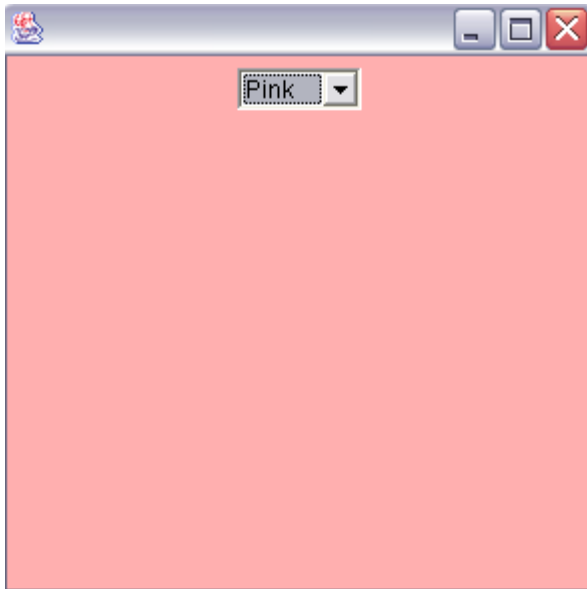
```
{  
    new ItemDemo();  
}
```

```
public boolean handleEvent(Event e)
```

```
{  
    if(e.id==Event.WINDOW_DESTROY)  
    {  
        System.exit(0);  
    }  
}
```

```
    }  
    return super.handleEvent(e);  
  }  
}
```

OUTPUT:



/* Program to show the use of Item Listener */

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

```
public class ItemDemo2 extends Frame implements ItemListener  
{  
    Choice c;  
    Checkbox bsc,msc,mca;  
    TextArea ta;  
    Panel p;  
    ItemDemo2()  
    {  
        setLayout(new GridLayout(1,2));  
        c=new Choice();  
        c.add("One");  
        c.add("Two");  
        c.add("Three");  
        bsc=new Checkbox("BSC");  
        msc=new Checkbox("MSC");  
        mca=new Checkbox("MCA");  
        p=new Panel();  
        p.setLayout(new GridLayout(4,1));  
        p.add(c);  
        p.add(bsc);  
        p.add(msc);  
        p.add(mca);  
        ta=new TextArea(15,30);  
        add(ta);  
        add(p);  
        c.addItemListener(this);  
        bsc.addItemListener(this);  
        msc.addItemListener(this);  
        mca.addItemListener(this);  
        setVisible(true);  
        setSize(300,300);  
    }  
}
```

```
public void itemStateChanged(ItemEvent e)  
{  
    if(e.getSource()==c)  
        ta.append(c.getSelectedItem());  
    if(c.getSelectedItem().equals("One"));  
        ta.append("One");  
    if(c.getSelectedItem().equals("Two"));  
        ta.append("Two");  
    if(c.getSelectedItem().equals("Three"));  
        ta.append("Three");  
}
```

```
if(bsc.getState())
    ta.append("BSC");
if(msc.getState())
    ta.append("MSC");
if(mca.getState())
    ta.append("MCA");
}

public static void main(String[] args)
{
    new ItemDemo2();
}

public boolean handleEvent(Event e)
{
    if(e.id==Event.WINDOW_DESTROY)
    {
        System.exit(0);
    }
    return super.handleEvent(e);
}
}
```

OUTPUT:



```
/* Program to demonstrate implementation of MouseListener */
```

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

```
public class MouseDemo extends Applet implements MouseListener
```

```
{  
    int x,y;  
    public void start()  
    {  
        x=0;  
        y=0;  
        addMouseListener(this);  
    }  
}
```

```
public void mouseClicked(MouseEvent e)  
{  
    x=e.getX();  
    y=e.getY();  
    repaint();  
}
```

```
public void mousePressed(MouseEvent e)  
{}
```

```
public void mouseReleased(MouseEvent e)  
{}
```

```
public void mouseEntered(MouseEvent e)  
{}
```

```
public void mouseExited(MouseEvent e)  
{}
```

```
public void paint(Graphics g)  
{  
    g.drawOval(x,y,50,50);  
}  
}
```

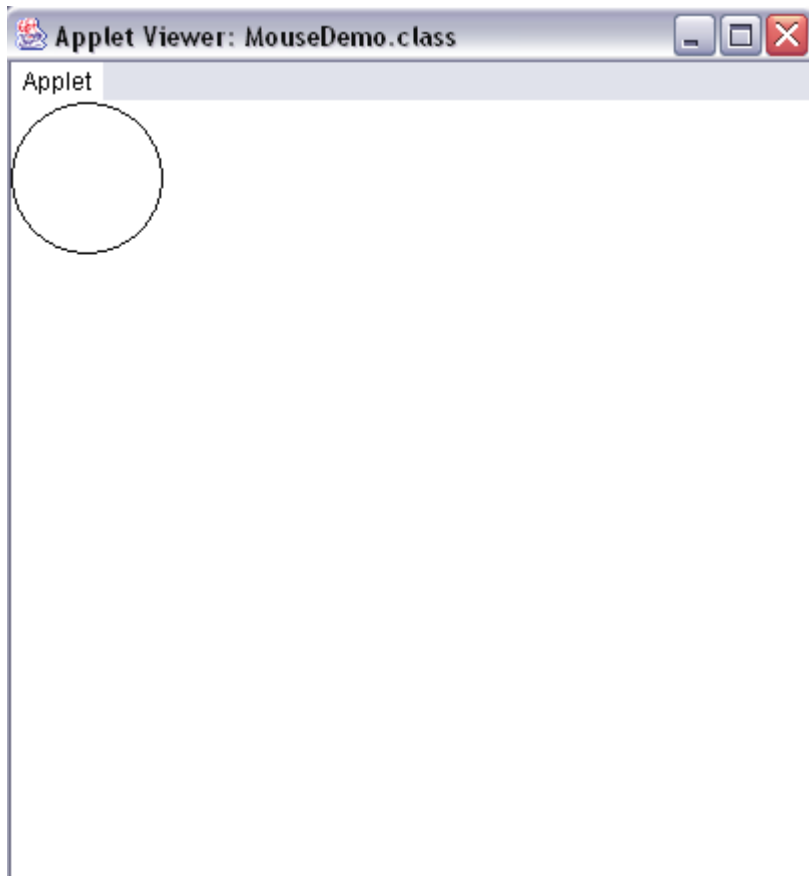
```
<HTML>
```

```
<applet code="MouseDemo.class" height=400 width=400>
```

```
</applet>
```

```
</HTML>
```

OUTPUT:



/* Program to show the use of MouseListener and increment the value of the button.

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

```
public class MouseDemo1 extends Frame implements MouseListener
```

```
{  
    int x,y,i=0;  
    MouseDemo1()  
    {  
        x=0;  
        y=0;  
        addMouseListener(this);  
        setLayout(null);  
        setVisible(true);  
        setSize(300,300);  
    }  
}
```

```
public void mouseClicked(MouseEvent e)  
{  
    Button b1=new Button((i++)+"_");  
    x=e.getX();  
    y=e.getY();  
    b1.setBounds(x,y,50,50);  
    add(b1);  
    validate();  
}
```

```
public void mousePressed(MouseEvent e)  
{}
```

```
public void mouseReleased(MouseEvent e)  
{}
```

```
public void mouseEntered(MouseEvent e)  
{}
```

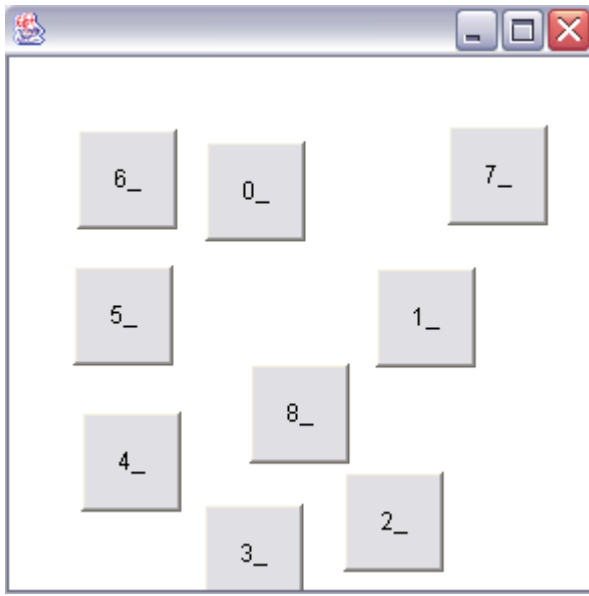
```
public void mouseExited(MouseEvent e)  
{}
```

```
public static void main(String[] args)  
{  
    new MouseDemo1();  
}
```

```
public boolean handleEvent(Event e)  
{
```

```
if(e.id==Event.WINDOW_DESTROY)
{
    System.exit(0);
}
return super.handleEvent(e);
}
}
```

OUTPUT:



```
/* Program to demonstrate FocusListener and WindowListener */
```

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

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

```
class FocusDemo extends Frame implements FocusListener, WindowListener
```

```
{  
    Button o,c,r;  
    Label l1,l2,l3;  
    FocusDemo()  
    {  
        setLayout(new FlowLayout());  
        o=new Button("OK");  
        c=new Button("Cancel");  
        r=new Button("Retry");  
        l1=new Label("Label1");  
        l2=new Label("Label2");  
        l3=new Label("Label3");  
        add(o); add(c); add(r);  
        add(l1);  
        add(l2);  
        add(l3);  
        o.addFocusListener(this);  
        c.addFocusListener(this);  
        r.addFocusListener(this);  
        addWindowListener(this);  
        setVisible(true);  
        setSize(700,700);  
    }  
}
```

```
public void focusGained(FocusEvent fe)
```

```
{  
    if(fe.getSource()==o)  
        l1.setText("Ok gained focus\n");  
    if(fe.getSource()==c)  
        l2.setText("Cancel gained focus \n");  
    if(fe.getSource()==r)  
        l3.setText("Retry gained focus \n");  
}
```

```
public void focusLost(FocusEvent fe)
```

```
{  
    if(fe.getSource()==o)  
        l1.setText("Ok lost focus\n");  
    if(fe.getSource()==c)  
        l2.setText("Cancel lost focus \n");  
    if(fe.getSource()==r)  
        l3.setText("Retry lost focus\n");  
}
```

```
}

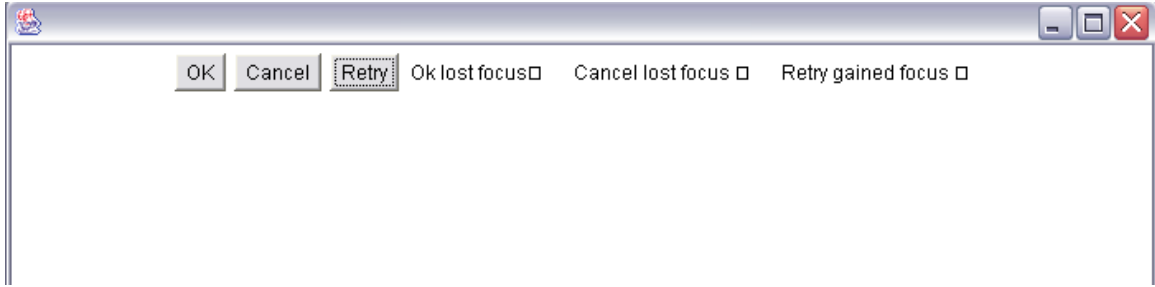
public void windowClosing(WindowEvent we)
{
    setVisible(false);
    System.exit(0);
}

public void windowClosed(WindowEvent we) {}
public void windowOpened(WindowEvent we) {}
public void windowActivated(WindowEvent we) {}
public void windowDeactivated(WindowEvent we) {}
public void windowIconified(WindowEvent we) {}
public void windowDeiconified(WindowEvent we) {}

public static void main(String[] argS)
{
    new FocusDemo();
}

public boolean handleEvent(Event e)
{
    if(e.id==Event.WINDOW_DESTROY)
    {
        System.exit(0);
    }
    return super.handleEvent(e);
}
}
```

OUTPUT:



```
/* Program to demonstrate Adapter class */
```

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

```
class AdapterDemo extends Frame implements ActionListener
```

```
{  
    Label u,p;  
    Button o,c;  
    TextField tf1,tf2;  
    dia d;
```

```
AdapterDemo()
```

```
{  
    u=new Label("Username");  
    p=new Label("Password");  
    tf1=new TextField(30);  
    tf2=new TextField(30);  
    o=new Button("Ok");  
    c=new Button("Reset");  
    tf2.setEchoChar('*');  
    setLayout(new FlowLayout());  
    add(u); add(tf1); add(p); add(tf2); add(o); add(c);  
    setVisible(true);  
    setSize(300,300);  
    o.addActionListener(this);  
    c.addActionListener(this);  
    addWindowListener(new MyAdapter());  
}
```

```
class MyAdapter extends WindowAdapter
```

```
{  
    public void windowClosing(WindowEvent we)  
    {  
        setVisible(false);  
        System.exit(0);  
    }  
}
```

```
public void actionPerformed(ActionEvent ae)
```

```
{  
    if(ae.getSource()==o)  
    {  
        if(tf1.getText().equals(tf2.getText()))  
            d=new dia(this,"Login Successful");  
        else  
            d=new dia(this,"Login Denied");  
    }  
}
```

```

        if(ae.getSource()==c)
        {
            tf1.setText(" ");
            tf2.setText(" ");
        }
    }

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

    public boolean handleEvent(Event e)
    {
        if(e.id==Event.WINDOW_DESTROY)
        {
            System.exit(0);
        }
        return super.handleEvent(e);
    }
}

class dia extends Dialog
{
    Label l;
    dia(Frame f,String s)
    {
        super(f,"Alert");
        l=new Label(s);
        setLayout(new FlowLayout());
        add(l);
        setVisible(true);
        setSize(300,300);
        addWindowListener(new winadp());
    }
}

class winadp extends WindowAdapter
{
    public void windowClosing(WindowEvent we)
    {
        setVisible(false);
        return;
    }
}
}

```

OUTPUT:

