# EXERCISE 1: CLASS AND OBJECTS

## AIM:

To write a program on class and objects

## ALGORITHM:

Step 1: Start
Step 2: Create a class
Step 3: Define a user define methods inside the class
Step 4: Create another class
Step 5: Create main method inside the new class
Step 6: Create objects for the first class and initialize it with default Constructor
Step 7: Call the user defined methods with the created objects
Step 8: End

## PROGRAM:

Student.java

```
import java.util.*;
class Details{
int id;
   String name;
int marks;
   // Creating Method with three parameters and void as return
type     void insert(int i, String n, int s) {        id=i;        name=n;
marks=s;
   }
   // Creating Method without parameters
void display()
   {
      System.out.println(id+" "+name+" "+marks);
   }
}
public class Student {     public static
void main(String[] args) {



      //Creating Objects   obj1,obj2,obj3 for Details class and initializing it with Default
constructor
      Details obj1=new Details();
      Details obj2=new Details();
      Details obj3=new Details();
      //calling insert method by passing three parameters
obj1.insert(1,"Arun",92);        obj2.insert(2,"Megha",50);
obj3.insert(3,"Vani",87);        //calling display method
obj1.display();        obj2.display();        obj3.display();
   }
}
```

**OUTPUT:**


## EXERCISE 2: INHERITANCE AND PACKAGES

### AIM:

To write a program on inheritance and packages

### ALGORITHM:

Step 1: Start

Step 2: Create 3 packages for single, multilevel and hierarchical inheritance

Step 3: Create 3 classes for single, multilevel and hierarchical inheritance in respective packages

Step 4: Implement Inheritance concepts

4.1 For Single create 2 classes extend sub class with Super Class

4.2 For Multilevel create 3 classes where 1st sub class extends super class and 2nd sub class extends 1st sub class

4.3 For Hierarchical create 3 classes where both 1st and 2nd sub class extends super class

Step 5: Create main class outside of all 3 packages

Step 6: Import all 3 packages we created in main class

Step 7: Create main method inside main class

Step 8: Call all 3 inheritance in main method

Step 9: End

**PROGRAM:**

SingleInheritance.java

```
package single;

//Creating SuperClass class
SuperClass {      public static
void printSuper() {
        System.out.println("Single Inheritance : Super Class");
    }
}

// Inheriting SuperClass in SingleInheritance Class
public class SingleInheritance extends SuperClass {
public static void startInheritance() {         // Calling
superclass method in sub class          printSuper();
        System.out.println("Single Inheritance : Sub Class");
    }
}
```

MultilevelInheritance.java

```
package multilevel;

// Creating SuperClass class
SuperClass {      public static
void printSuper() {
        System.out.println("Multilevel Inheritance : Super Class");
    }
}
// Inheriting SuperClass in SubClass1 class
SubClass1 extends SuperClass {     public static void printSub1()
{
        // Calling SuperClass method from SubClass1         printSuper();
        System.out.println("Multilevel Inheritance : Sub Class - 1");
    }
}
// Inheriting SubClass1 in MultilevelInheritance public class
MultilevelInheritance extends SubClass1 {

    public static void startInheritance() {
        // Calling SubClass1 method from MultilevelInheritance         printSub1();
        System.out.println("Multilevel Inheritance : Sub Class - 2");
    }
}
```

## HierarchicalInheritance.java

```
package hierarchical;

// Creating SuperClass class
SuperClass {      public static
void printSuper() {
        System.out.println("Hierarchical Inheritance : Super Class");
    }
}
// Inheriting SuperClass in SubClass1 class
SubClass1 extends SuperClass {     public static
void printSub1() {
        // Calling SuperClass method from SubClass1
```

```
        printSuper();
        System.out.println("Hierarchical Inheritance : Sub Class - 1");
    }
}
// Inheriting SuperClass in HierarchicalInheritance public
class HierarchicalInheritance extends SuperClass {      public
static void startInheritance() {
        // Calling SuperClass method from HierarchicalInheritance         printSuper();
        // Calling SubClass1 method from HierarchicalInheritance
        SubClass1.printSub1();
        System.out.println("Hierarchical Inheritance : Sub Class - 2");
    }
}
```

## Main.java

```
// Importing all 3 packages that contains inheritance import
hierarchical.HierarchicalInheritance; import
multilevel.MultilevelInheritance; import single.SingleInheritance;

public class Main {
    public static void main(String[] args) {
        // Method call which moves to Single Inheritance class
        System.out.println("\t\tSingle Inheritance");         SingleInheritance.startInheritance();

        // Method call which moves to Multilevel Inheritance class
        System.out.println("\n\t\tMultilevel Inheritance");
MultilevelInheritance.startInheritance();

        // Method call which moves to Hierarchical Inheritance class
        System.out.println("\n\t\tHierarchical Inheritance");
        HierarchicalInheritance.startInheritance();     }
}
```

**OUTPUT :**

```
Run:  Main
"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...
        Single Inheritance
Single Inheritance : Super Class
Single Inheritance : Sub Class

        Multilevel Inheritance
Multilevel Inheritance : Super Class
Multilevel Inheritance : Sub Class - 1
Multilevel Inheritance : Sub Class - 2

        Hierarchical Inheritance
Hierarchical Inheritance : Super Class
Hierarchical Inheritance : Super Class
Hierarchical Inheritance : Sub Class - 1
Hierarchical Inheritance : Sub Class - 2

Process finished with exit code 0
```

## EXERCISE 3: INTERFACES

**AIM:**

To write a program  on Interface

**ALGORITHM:**

Step 1: Start

Step 2: Create an interface

Step 3: Declare the methods with parameters inside interface

Step 4: Create main class and implement the interface in main class

Step 5: Give method definition to all the methods declared in interface

Step 6: Create main method

Step 7: call the methods

Step 8: End

## PROGRAM

Main.java

```
package com.company;
import java.util.*;
interface Polygon {
    //  Defining the methods in interface
void getArea(int length, int breadth);
}
// implement the Polygon interface
class Rectangle implements Polygon {


    // implementation of abstract method
public void getArea(int length, int breadth) {
        System.out.println("The area of the rectangle is " + (length * breadth));
    }
}
```

```java
class Main {
    public static void main(String[] args) {
        int a,b;
        //getting input
        Scanner sc=new Scanner(System.in);
        System.out.println("enter length and breadth of rectangle :");
        a=sc.nextInt();
        b=sc.nextInt();
        // Calling
        Rectangle r1 = new Rectangle();
        r1.getArea(a, b);
    }
}
```

## OUTPUT:

```
Main
"C:\Program Files\Java\jdk1.8.0_271\bin\java.exe" ...
enter length and breadth of rectangle :
2
3
The area of the rectangle is 6

Process finished with exit code 0
```

## EXERCISE 4: EXCEPTION HANDLING TECHNIQUE

## AIM:

To write a program on exception handling.

## ALGORITHM:

Step 1: Start
Step 2: Create class and main method
Step 3: Start Implementing exception with error conditions
3.1: ArrayIndexOutOfBoundExceptiom
3.1.1: Initialize an array
3.1.2: Open Try inside it get input in for loop for the index greater than array size
3.1.3: Print the error in catch block
3.2: ArithmeticExceptiom
3.2.1: Open Try inside it divide a number with 0
3.2.2: Print the error in catch block
3.3: NullPointerException
3.3.1: Initialize String with null value
3.3.2: Open Try inside it do operations with that string like getting length
3.3.3: Print the error in catch block
3.4: ClassCastException
3.4.1: Initialize a int or float value to object
3.4.2: Inside try try to convert the object to string
3.4.3: Print the error in catch block
3.5: NegativeArraySizeException
3.5.1: Open try inside it try to create an array with negative size
3.5.2: Print the error in catch block
3.6: ClassNotFoundException
3.6.1: Open try inside it use Class.forname and give a string value

3.6.2: Print the error in catch block

3.7: NumberFormatException

3.7.1: Open try inside it try to convert string type to int using Integer.parseInt()

3.7.2: Print the error in catch block

3.8: StringIndexOutOfBoundsException

3.8.1: Initialize a string

3.8.2: Inside try give get the character using charAt() whose position is greater than string size

3.8.3: Print the error in catch block

Step 4: End

**PROGRAM:**

ExceptionEG.java

```
package com.company;
import java.util.*;
/*ArrayIndexOutOfBounds
 Arithmetic Exception
 NullPointerException
 ClassCastException
 NegativeArraySizeException
 ClassNotFoundException
 Number format exception
 StringIndexOutOfBoundsException
 */ public class
ExceptionEG {
    public static void main(String args[]) {
Scanner sc = new Scanner(System.in);
int i;        int[] a = new int[2];        try
        {   //Trying to print the index th position value from array if it exceeds array size then
exception is thrown
            System.out.println("ArrayIndexOutOfBounds exception: is thrown if a program tries
to access an array index that is negative, greater than, or equal to the \nlength of the
array.The ArrayIndexOutOfBounds exception is a run-time exception. Java's compiler does
not check for this error during compilation.\n");
            for (i = 0; i <= 2; i++) {
                System.out.println("Enter value-" + (i + 1) + " : ");
a[i] = sc.nextInt();
            }
```

```
        }
        catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("\t\tValues exceeded the limit");
e.printStackTrace();
            System.out.println("\n**********************************\n");
        }
try {
            //code that may raise exception
            //Trying to divide two numbers if 0 is passed as divisor it will throw divided by zero
error
            System.out.println("Arithmetic Exception : arises when we trying to divide by
zero.Now Dividing first value with second value");            int data = a[0] / a[1];
        }
        catch (ArithmeticException e)
        {
            System.out.println("Arithmetic Exception caught " + e);
            System.out.println("The java.lang.ArithmeticException is an unchecked exception in
Java. Usually, one would come across java.lang.ArithmeticException: / by zero which \n
occurs when an attempt is made to divide two numbers and the number in the denominator
is zero....");
            System.out.println("\n**********************************\n");
        }
try
        {
            //  Trying to call method with a variable initialized as null
            String s = null;
            System.out.println(s.length());
        }
        catch(NullPointerException e)
        {
            e.printStackTrace();
            System.out.println("NullPointerException is a RuntimeException. In Java, a special
null value can be assigned to an object reference. NullPointerException is thrown when
program attempts to use an object reference that has the null value.");
            System.out.println("\n**********************************\n");
        }
try
        {
            //Trying to convert object into string if object contains int or float data it will throw
error
            Object s = 5.0f;
            System.out.println((String) s);
        }
        catch (ClassCastException e)
        {
            System.out.println("ClassCastException");
            System.out.println("Class Cast exception is showing here because we are trying to
cast the value of float to the class String");
            System.out.println(e.toString());
            System.out.println("\n**********************************\n");
```

13

15

```
        }
try  {
          //If we try to give negative array size it will go to catch block
            System.out.println(" NegativeArraySizeException: Thrown if an application tries
to create an array with negative size");            int[] b = new int[-1];
        }
        catch (NegativeArraySizeException e)
        {
            System.out.println("\t\tArray can't have negative index");
e.printStackTrace();
            System.out.println("\n**********************************\n");
        }
try{
            //If Invalid class name is passed then flow will go to catch block
            System.out.println("ClassNotFoundException :is thrown when the Java Virtual
Machine (JVM) tries to load a particular class and the specified class cannot be found in the
classpath.....");
            Class.forName("Hello World");
        }
        catch (ClassNotFoundException ex) {
ex.printStackTrace();
            System.out.println("\n**********************************\n");
        }
try  {
            //  If we try to parse string or other datatype as integer catch block will be executed
int u = Integer.parseInt("sruthi");
            System.out.println(u);
        }
        catch (NumberFormatException e)
        {
            System.out.println("\nNumberFormatException");
            System.out.println("Number format exception is showing here because we are trying
to store the string hello inside the int variable");
            System.out.println(e.toString());
            System.out.println("\n**********************************\n");
        }
try    {
            //Thrown by String methods to indicate that an index is either negative or greater
than the size of the string
            String s = "Exception Handling in Java!";
System.out.println("Length: " + s.length());
char c = s.charAt(100);
        }
        catch(StringIndexOutOfBoundsException e){
            System.out.println("String index is out of defined bounds");
            System.out.println("\n**********************************\n");
        }}}
```

# OUTPUT:

```
Exception6 :
"C:\Program Files\Java\jdk1.8.0_271\bin\java.exe" ...
ArrayIndexOutOfBounds exception: is thrown if a program tries to access an array index that is negative, greater than, or equal to the
length of the array.The ArrayIndexOutOfBounds exception is a run-time exception. Java's compiler does not check for this error during compilatio

Enter value-1 :

Enter value-2 :

Enter value-3 :

        Values exceeded the limit

***********************************

Arithmetic Exception : arises when we trying to divide by zero.Now Dividing first value with second value
NullPointerException is a RuntimeException. In Java, a special null value can be assigned to an object reference. NullPointerException is thrown
 when program attempts to use an object reference that has the null value.

***********************************

ClassCastException
Class Cast exception is showing here because we are trying to cast the value of float to the class String
java.lang.ClassCastException: java.lang.Float cannot be cast to java.lang.String

***********************************
```

```
 NegativeArraySizeException: Thrown if an application tries to create an array with negative size
        Array can't have negative index

************************************

ClassNotFoundException :is thrown when the Java Virtual Machine (JVM) tries to load a particular class and the specified class cannot be found in
  the classpath.....

************************************


NumberFormatException
Number format exception is showing here because we are trying to store the string hello inside the int variable
java.lang.NumberFormatException: For input string: "sruthi"

************************************

Length: 27
String index is out of defined bounds

************************************
```

```
java.lang.ArrayIndexOutOfBoundsException Create breakpoint : 2
    at com.company.ExceptionEG.main(ExceptionEG.java:22)
java.lang.NullPointerException Create breakpoint
    at com.company.ExceptionEG.main(ExceptionEG.java:47)
java.lang.NegativeArraySizeException Create breakpoint
    at com.company.ExceptionEG.main(ExceptionEG.java:72)
java.lang.ClassNotFoundException Create breakpoint : Hello World
    at java.net.URLClassLoader.findClass(URLClassLoader.java:382)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:418)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:355)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:351)
    at java.lang.Class.forName0(Native Method)
    at java.lang.Class.forName(Class.java:264)
    at com.company.ExceptionEG.main(ExceptionEG.java:84)
```

## EXERCISE 5 : INPUT/OUTPUT STREAMS

**AIM:**

To write a program on input / output streams

**ALGORITHM:**

Step 1: Start
Step 2: Create main class and main method
Step 3: Create Buffered Reader object and initialize it with input stream
Step 4: Get input from user using buffered reader object Step 5: Create FileOutputStream and FileInputStream object initialized with the file name
Step 6: Write the content user read to file using FileOutputStream object
Step 7: Read the contetn of the file using FileInputStream object
Step 8: End

**PROGRAM:**

Main.java

```
import java.io.*; public class Main {
public static void main(String[] args) {
        String  text;
int ch;        byte[]
bytes;
        // Opening InputStream in BufferedReader object to get input from user
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter  the  text  you  want  in  file");
try {
            //  Getting  input  for  file  content
text = br.readLine();
            // Opening MyFile.txt with write permission, in-case if there is no file in that name
it creates one file
            FileOutputStream fos = new FileOutputStream("MyFile.txt");
            // Opening MyFile.txt with read permission
            FileInputStream fis = new FileInputStream("MyFile.txt");
            // Convert to content string to byte array
```

```
            bytes = text.getBytes();
            // Writing the content to file in bytes
fos.write(bytes);            // Closing fos
fos.close();
            // Reads each character in file and prints in output
while ((ch=fis.read())!=-1) {
                System.out.print((char)ch);
            }
        }catch (IOException e) {
e.printStackTrace();
        }


    }
}
```

## OUTPUT:

```
Main (1)
"C:\Program Files\Java\jdk1.8.0_271\bin\java.exe" ...
Enter the text you want in file
this is the content that will be added
this is the content that will be added
Process finished with exit code 0
```

## EXERCISE 6 : THREADS

**AIM:**

To write a program on threading.

**ALGORITHM:**

Step 1: Start
Step 2: Create class that extend Thread class Step
3: Override run method and inside run method
print the thread name Step 4: Create main method
Step 5: Print main thread
Step 6: Start the thread using start method
Step 7: Create another class that implements Runnable Interface
Step 8: Override run method and inside run method print the thread name
Step 9: Create main method
Step 10: Create objects for thread class
Step 11: Print main thread
Step 12: Start the thread using start method
Step 13: End

**PROGRAM:**

Main.java

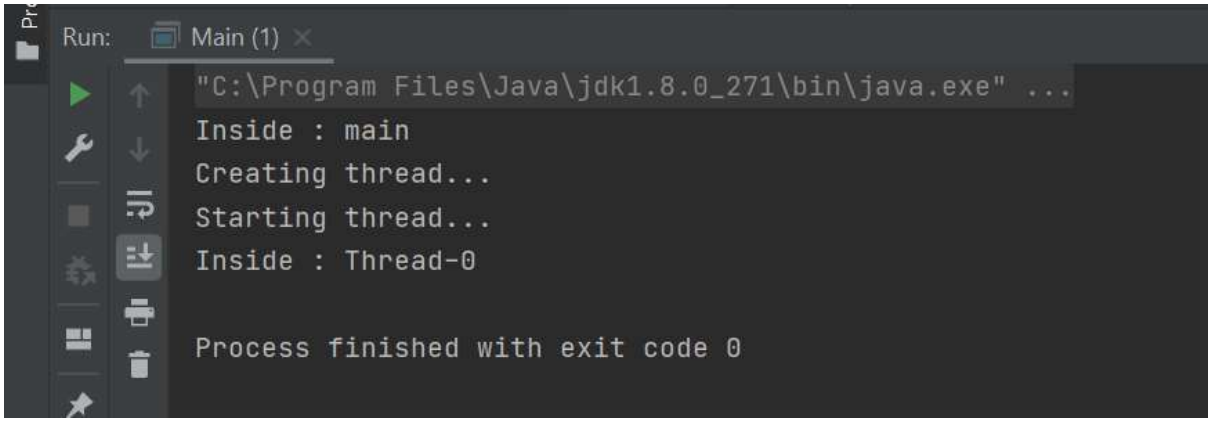
```
import java.util.*;
// Creating Thread by extending Thread class public
class Main extends Thread {
    // run() method contains the code that is executed by the thread.
    @Override
    public void run() {
        // Prints the currently exceuting thread name
```

```
        System.out.println("Inside : " + Thread.currentThread().getName());
    }
    public static void main(String[] args) {
        // Prints the Main Thread
        System.out.println("Inside : " + Thread.currentThread().getName());
        System.out.println("Creating thread...");
        // Creating a new Thread
        Thread thread = new Main();
        System.out.println("Starting thread...");
        //Starting the Thread
        thread.start();
    }
}
```

## Mains.java

```
package Oopassignment;
// Creating Thread by implementing Runnable interface
public class Mains implements Runnable {     public
static void main(String[] args) {
        // Prints the Main Thread
        System.out.println("Inside : " + Thread.currentThread().getName());
        System.out.println("Creating Runnable...");
        // Getting instance of the class and storing in runnable
        Runnable runnable = new Mains();
        System.out.println("Creating Thread...");
        // Creating new thread with runnable object
        Thread thread = new Thread(runnable);

        System.out.println("Starting Thread...");
        thread.start();
    }
// run() method contains the code that is executed by the thread.
    @Override
public void run() {
        // Prints the currently exceuting thread name
        System.out.println("Inside : " + Thread.currentThread().getName());
    }
}
```

**OUTPUT:**


Run:
Main (1)
"C:\Program Files\Java\jdk1.8.0_271\bin\java.exe" ...
Inside : main
Creating thread...
Starting thread...
Inside : Thread-0
Process finished with exit code 0

```
Run:  Main (1)
"C:\Program Files\Java\jdk1.8.0_271\bin\java.exe" ...
Inside : main
Creating Runnable...
Creating Thread...
Starting Thread...
Inside : Thread-0

Process finished with exit code 0
```

# EXERCISE 7 : AWT CONTROLS

## AIM:

To write a program on various AWT controls.

## ALGORITHM:

Step 1: Start
Step 2: Import awt packages
Step 3: Create class and declare the awt controls in class Step 4: Define a constructor and inside constructor initialize frame, setsize of frame and make it visible
Step 5: After initializing frame initialize the controls
Step 6: Place the elements in specified position using setBounds method
Step 7: Declare void main method and call the constructor in main
Step 8: End

## PROGRAM:

AWTEvents.java

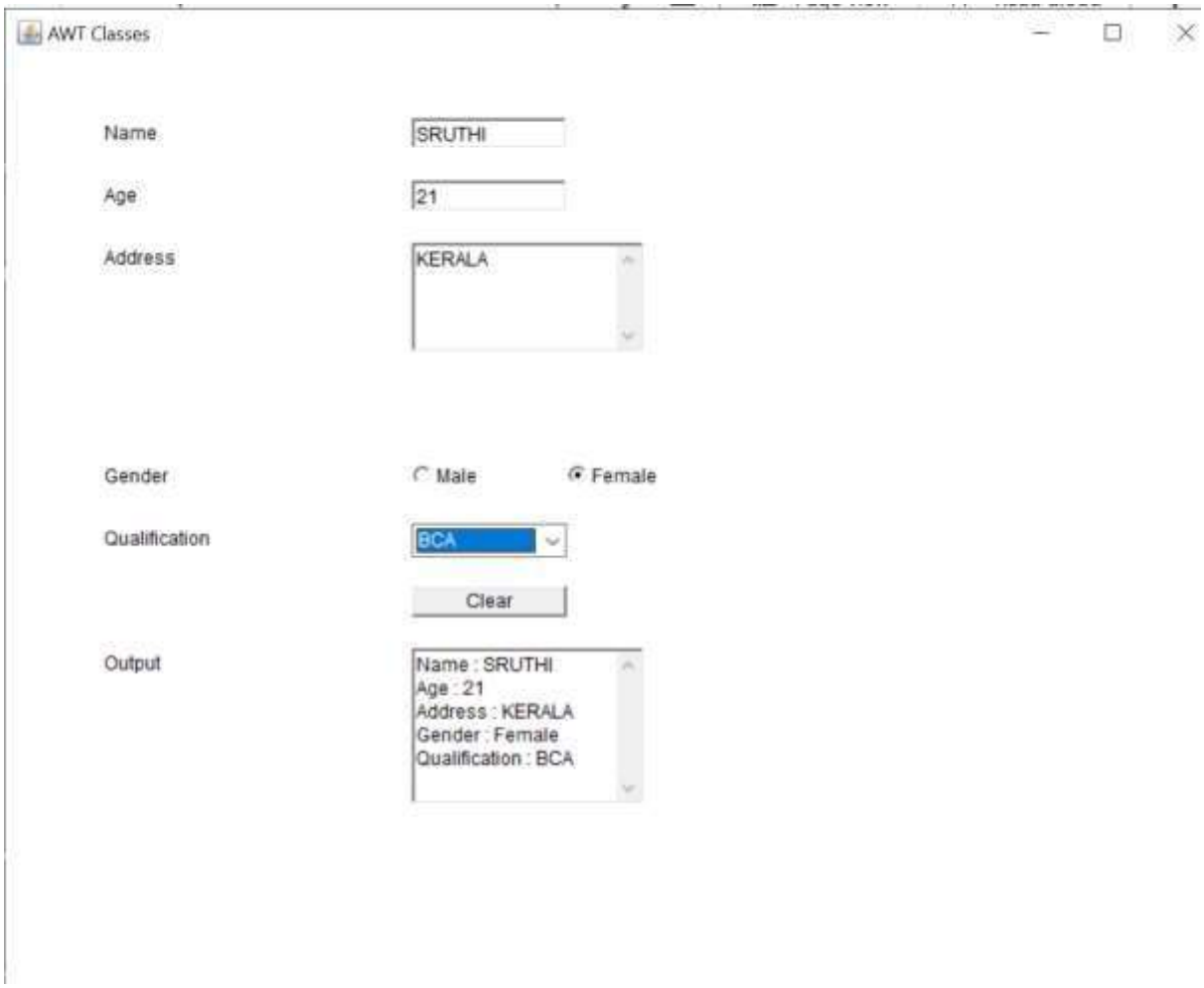
```
package Oopassignment;
import java.awt.*;
import java.awt.event.*;
public class AWTEvents extends Frame implements ActionListener, ItemListener,
TextListener {
    Label l1,l2,l3,l4,l5,l6;
    TextField t1,t2;
    TextArea ta1,ta2;
    Checkbox c1,c2;
    CheckboxGroup cg;
    Choice ch1;
    Button b1;
    String name = "",age = "",gender = "",qualification = "",address = "";
AWTEvents() {        super("AWT Classes");        setLayout(null); //
Make Frame visible in full screen
```

```
        setSize(800,800);
setVisible(true);
        //Initializing AWT Components
l1 = new Label("Name");        l2 = new
Label("Age");        l3 = new
Label("Address");        l4 = new
Label("Gender");        l5 = new
Label("Qualification");        l6 = new
Label("Output");        t1 = new TextField();
t2 = new TextField();        ta1 = new
TextArea();        ta2 = new TextArea();
b1 = new Button("Clear");        cg = new
CheckboxGroup();        c1 = new
Checkbox("Male",cg,false);        c2 = new
Checkbox("Female",cg,false);
      ch1 = new Choice();//Adding AWT Components to Frame

        ch1.add("BBA");
ch1.add("BCA");
ch1.add("MBA");
ch1.add("MCA");        add(l1);
l1.setBounds(70,70,100,20);
add(t1);
t1.setBounds(270,70,100,20);
add(l2);
l2.setBounds(70,110,100,20);
add(t2);
t2.setBounds(270,110,100,20);
add(l3);
l3.setBounds(70,150,100,20);
add(ta1);
ta1.setBounds(270,150,150,70);
add(l4);
l4.setBounds(70,290,100,20);
add(c1);
        c1.setBounds(270,290,80,20);
add(c2);
        c2.setBounds(370,290,80,20);
add(l5);
l5.setBounds(70,330,100,20);
add(ch1);
ch1.setBounds(270,330,100,20);
add(b1);
b1.setBounds(270,370,100,20);
add(l6);
l6.setBounds(70,410,100,20);
add(ta2);
        ta2.setBounds(270,410,150,100);
```

```
        //Set ActionPerformed, itemStateChanged and TextValueChanged Listener
      b1.addActionListener(this);
c1.addItemListener(this);
c2.addItemListener(this);
ch1.addItemListener(this);
t1.addTextListener(this);        t2.addTextListener(this);
ta1.addTextListener(this);
   }
   public void actionPerformed(ActionEvent e) {        t1.setText("");// If button is clicked
removes all the values given in components
       t2.setText("");
ta1.setText("");
c1.setState(false);
c2.setState(false);
ch1.select(0);
name ="";        age
="";        address ="";
gender="";
qualification="";
ta2.setText("");
   }
   public void itemStateChanged(ItemEvent e) {
if(e.getItemSelectable() == c1)

 // Checking which checkbox or choice is selected if c1 male, if c2 female if
nothing then it is  choice box for qualification            gender = "Male";        else
if(e.getItemSelectable() == c2)            gender = "Female";        else            qualification =
ch1.getSelectedItem();        ta2.setText("");        if (!name.isEmpty())        ta2.append("Name
: "+name);        if (!age.isEmpty())        ta2.append("\nAge : "+age);        if
(!address.isEmpty())        ta2.append("\nAddress : "+address);        if (!gender.isEmpty())
ta2.append("\nGender : "+gender);        if (!qualification.isEmpty())
      ta2.append("\nQualification : "+qualification);
   }
   public void textValueChanged(TextEvent e) {

       // To update name , age and address values if changes happened


      if(e.getSource() == t1)
```

```
            name = t1.getText();
        else if(e.getSource() == t2)
            age = t2.getText();
        else
            address = ta1.getText();
        ta2.setText("");
        if (!name.isEmpty())
            ta2.append("Name : "+name);
        if (!age.isEmpty())
            ta2.append("\nAge : "+age);
        if (!address.isEmpty())
            ta2.append("\nAddress : "+address);
        if (!gender.isEmpty())
            ta2.append("\nGender : "+gender);
        if (!qualification.isEmpty())
            ta2.append("\nQualification : "+qualification);
    }
    public static void main(String[] args) {

//Calling Constructor where frame and all controls are initialized
            new AWTEvents();
    }
}
```

**OUTPUT:**




## EXERCISE 8 :JAVA BEANS

## AIM:

To write a program on Java Beans

## ALGORITHM:

Step 1: Start
Step 2: Create a javabeans class and declare attributes
Step 3: Define setter and getter methods for the attributes
Step 4: Create main class and main method
Step 5: Create an object for javabeans class
Step 6: Use the object and store values for the attributes using setter method
Step 7: Use the object and retrive values using getter method
Step 8: End

## PROGRAM:

demo.java

```
import java.io.Serializable; class
Student implements Serializable {
int id;
   String name, dept;
   //  Defining setter methods
public void setName(String name) {
this.name = name;
   }
   public void setId(int id) {
      this.id = id;
   }
   public void setDept(String dept) {
      this.dept = dept;
   }
   //  Defining getter methods
   public int getId() {
return id;
   }
   public String getName() {
return name;
   }
   public String getDept() {
return dept;
   }
}
```

Main.java

```
Main.java
import java.io.*;

public class Main {
    public static void main(String[] args) throws IOException {
        //Creating object for java bean class
        Student student = new Student();

        //Getting input from user and giving it in setter methods
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter Student id");
        student.setId(Integer.parseInt(br.readLine()));
        System.out.println("Enter Student Name");
        student.setName(br.readLine());
        System.out.println("Enter Student dept");
        student.setDept(br.readLine());
        System.out.println("\t\tPrinting the Output : ");

        //Getting value from getter methods and printing it inn output
        System.out.println("Student id : " + student.getId());
        System.out.println("Student name : " + student.getName());
        System.out.println("Student dept : " + student.getDept());
    }
}
```

## OUTPUT:


# EXERCISE 9 : SWINGS

## AIM:

To write a program on Swings.

## ALGORITHM:

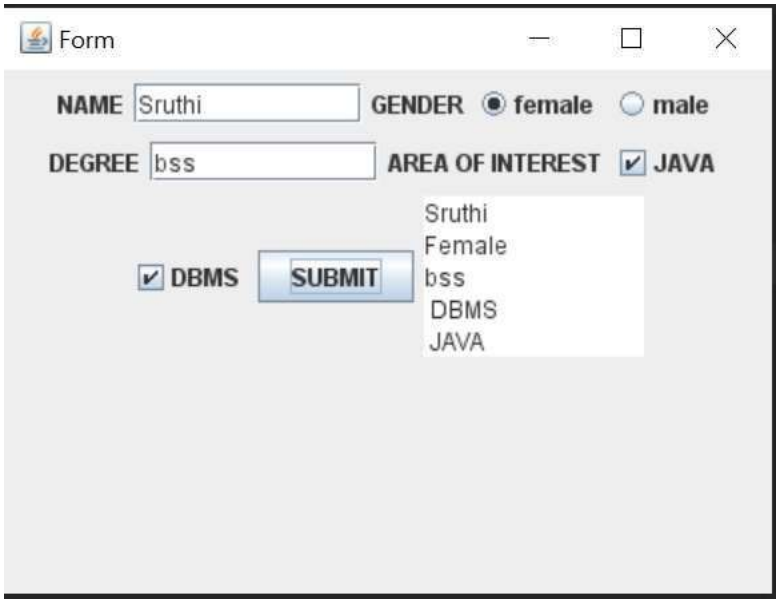
Step 1: Start
Step 2: Import swing packages
Step 3: Create class and declare the swing controls in class
Step 4: Define a constructor and inside constructor initialize Jframe, setsize of Jframe and make it visible
Step 5: After initializing Jframe initialize the controls
Step 6: Declare void method and call the constructor in main
Step 7: End

## PROGRAM:

```
package Oopassignment; import java.awt.*; import
java.awt.event.*; import javax.swing.*; class forms
extends JFrame implements ActionListener
{   //Declaring JSwing Components
    JRadioButton rb1,rb2;
    JTextArea ta;
    String msg;
    ButtonGroup br;
    JLabel lb1,lb2,lb3,lb4;
    JTextField ta1,ta2;
    JButton b1;
    JCheckBox cb1,cb2;
forms()
    {
        Container c=getContentPane();
        c.setLayout(new FlowLayout());
rb1=new JRadioButton("female",true);
rb2=new JRadioButton("male");
br=new ButtonGroup();
        //Initializing Swing Components
lb1=new JLabel("NAME");         lb2=new
JLabel("DEGREE");         lb3=new
JLabel("AREA OF INTEREST");
lb4=new JLabel("GENDER");
```

```
        ta1=new JTextField(10);
ta2=new JTextField(10);         b1=new
JButton("SUBMIT");         cb1=new
JCheckBox("DBMS");         cb2=new
JCheckBox("JAVA");         ta=new
JTextArea(5,10);
        //Adding Swing Components to
JFrame         br.add(rb1);         br.add(rb2);
c.add(lb1);
        c.add(ta1);
        c.add(lb4);
        c.add(rb1);
        c.add(rb2);
        c.add(lb2);
        c.add(ta2);
        c.add(lb3);
        c.add(cb2);
        c.add(cb1);
        c.add(b1);
        c.add(ta);         setSize(400,300);
setVisible(true);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setTitle("Form");
        b1.addActionListener(this);
    }
    public static void main(String args[])
    {   // Calling Constructor where frame and all controls are initialized
forms f = new forms();
    }
    //Set ActionPerformed Listener
    public void actionPerformed(ActionEvent ae)
    {
        msg=ta1.getText();
if(rb1.getModel().isSelected())msg+="\nFemale\n";
else msg+="\nMale\n";         msg+=ta2.getText();
if(cb1.getModel().isSelected())msg+="\n DBMS";
if(cb2.getModel().isSelected())msg+="\n JAVA";
ta.setText(msg);
    }
}
```

**OUTPUT:**


Form
NAME
Sruthi
GENDER
female
male
DEGREE
bss
AREA OF INTEREST
JAVA
DBMS
SUBMIT
Sruthi
Female
bss
DBMS
JAVA

# EXERCISE 10 : SERVLETS

## AIM:

To write a program on Servlets.

## ALGORITHM:

Step 1: Start
Step 2: Design an HTML form and give action as servlet class name and method as post
Step 3: Create a servlet class extending HttpServlet class Step 4: Override doPost method with parameters for HttpRequest and HttpResponse and throw ServletException
Step 5: Using request.getParameter() get the values of the HTML form
Step 6: Load the Database driver using Class.forname
Step 7: Establish database connection with using DriverManager.getConnection() and store the connection in Connection class
Step 8: Create an object for Statement class using connection class intialize it.
Step 9: Do SQL Operations with statement object
Step 10: Close connection
Step 11: End

## PROGRAM:

### index.html

```
<!DOCTYPE html>
<html>
<head>
<title>OOP Assigment</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
```

```
<form action="myservlet"  method="POST">
<h2>Enter values</h2>
<label><b>ROLL NO:</b></label>
<label><b>NAME</b></label>
<input type="text" placeholder="Enter Name" name="name">
<label><b>PLACE</b></label>
<input type="text" placeholder="Enter place" name="place">
<label><b>MARKS</b></label>
<input type="text" placeholder="Enter marks" name="marks">
<hr> <input type="submit" value="submit">
</form> </body>
</html>
```

myservelet.java

```
import java.io.IOException; import
java.io.PrintWriter;
import javax.servlet.ServletException; import
javax.servlet.annotation.WebServlet; import
javax.servlet.http.HttpServlet; import
javax.servlet.http.HttpServletRequest; import
javax.servlet.http.HttpServletResponse; import
java.sql.*;
@WebServlet(urlPatterns = {"/myservlet"}) public
class myservlet extends HttpServlet {
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out=response.getWriter();// Storing response writer in
printwritter to print in webpage
        // Getting values from html page using request
try{
            int rollno = Integer.parseInt(request.getParameter("rollno"));
            String name=request.getParameter("name");
            String place=request.getParameter("place");
            int marks = Integer.parseInt(request.getParameter("marks"));
Class.forName("oracle.jdbc.OracleDriver");// Loading mysql driver
            // creating connection
            Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE",
"sruthi","jackey");
            // Initializing statement to run sql statements
            Statement stmt=con.createStatement();
// Inserting values to database
stmt.executeUpdate("insert into students
values('"+rollno+"','"+name+"','"+place+"','"+marks+"')");
out.println("Values inserted successfully");
con.close();
        }catch(Exception e){
            e.printStackTrace();
out.println(e.toString());
        }finally {
out.close();
```

```
}}}
```

# OUTPUT:

"NetBeans Connector" started debugging this browser Cancel

Values inserted successfully

```
SQL> select * from students;

   ROLL_NO NAME                 PLACE                    MARKS
---------- -------------------- -------------------- ----------
        13 Sang                 ap                           44

SQL>
```