



School of Computer Science and Engineering (SOCSE) Department of Computer Science and Application



LAB MANUAL

Class: Bachelor in Computer Application

Year/Sem: SYBCA/4th Semester

Course Code/Course: XCA411 – Lab Course based on Java Programming.

Academic Year: 2023 - 2024



Trimbak Road, A/p - Mahiravani, Tal. & Dist. – Nashik, Pin – 422 213
School of Computer Sciences and Engineering
Department of Computer Science and Application
website: <http://www.sandipuniversity.edu.in>

CERTIFICATE

This is to certify that Mr/ Ms.

_____, PRN

No. _____ Student of _____ Semester of the program

_____ has satisfactorily / unsatisfactorily completed the Practical/

Assignments of _____ in the

Department of Computer Science & Application, School of Computer Sciences and Engineering, Sandip University, Nashik in the Academic Year 2023-24.

Date:

Sign. of Faculty

Dean

Head of the Department

Table of contents

Sr. No.	Title of Experiment	Page No	Date	Remark
1	Program for calculating result of a student which includes total marks, percentage,			
2	Program to implement various control statement in java i.e. if, if-else, for, while, and switch			
3	Program to implement method overriding and method overloading in java			
4	Program to implement Single and Multilevel inheritance in java			
5	Program to implement Single and Multilevel inheritance in java			
6	Program to implement Array and its types in java			
7	Program to implement types of constructors in java			
8	Program to perform abstract classes and methods in java			
9	Program to perform interfaces in java			
10	Program to implement exception handling in java			
11	Program to demonstrate Applet in Java			

Lab Assignment No. 1

Aim: Program for calculating result of a student which includes total marks, percentage, average and grades obtained

Theory:

If statement

When you want to test only the single condition and print the result only when the condition is true. If the condition is false, the given statement in the if condition will not get executed.

If else statement

The if else statement checks a condition, if it is true. If the statement is not true, it will execute the statement under the else statement.

Formulas

Considering 5 subjects of 100 marks,

Total marks: Sum of all the marks obtained i.e. $(A+B+C+D+E)$

Percentage: $(\text{Sum of the marks obtained}) / (500) * 100$ i.e. $[(\text{Total marks obtained} / 500) * 100]$

Average: $(\text{Total marks obtained}) / \text{Number of subjects or exams}$ i.e. $(\text{Total marks obtained}) / 5$

Program:

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

        System.out.println("Enter Your Marks :- ");

        float Maths = in.nextFloat();
        float phy = in.nextFloat();
        float Chem = in.nextFloat();
        float his = in.nextFloat();
        float geo = in.nextFloat();

        float total;
        float average;
        float percentage;
        char grade;

        total = Maths + phy + Chem + his + geo;
```

```

        average =(float)(total /5.0);
        percentage =(float)((total/500)*100);

        if(percentage>=90)
            grade ='A';
        elseif(percentage<90 && percentage >=80)
            grade ='B';
        elseif(percentage<80 && percentage >=70)
            grade ='C';
        elseif(percentage<70 && percentage <=50)
            grade ='D';
        else
            grade ='E';

        System.out.println("\n The total marks is :- "+total);
        System.out.println("\n The Average is :- "+average);
        System.out.println("\n The Percentage is :- "+percentage);
        System.out.println("\n The grade is :- "+grade);
    }
}

```

Output:

```

Enter Your Marks :-
78
98
56
76
80

The total marks is :- 388.0

The Average is :- 77.6

The Percentage is :- 77.600006

The grade is :- B

```

Aim: Program to implement various control statement in java i.e. if, if-else, for, while, and switch

Theory:

If statement

When you want to test only the single condition and print the result only when the condition is true. If the condition is false, the given statement in the if condition will not get executed.

If else statement

The if else statement checks a condition, if it is true. If the statement is not true, it will execute the statement under the else statement.

for loop statement

The Java *for loop* is used to iterate a part of the program several times. If the number of iteration is **fixed**, it is recommended to use for loop.

while loop statement

The Java while loop is used to iterate a part of the program repeatedly until the specified Boolean condition is true. As soon as the Boolean condition becomes false, the loop automatically stops.

switch statement

The Java switch statement executes one statement from multiple conditions. It is like if-else-if ladder statement. The switch statement works with byte, short, int, long, enum types, String and some wrapper types like Byte, Short, Int, and Long.

Program:

If statement

```
public class controlst1 {  
    public static void main (String  
        args[]) { int num = 20;  
        if (num < 50)  
            System.out.println("The number is lesser than 50");  
    }  
}
```

```
The number is lesser than 50
```

If else statement

```
public class controlst2 {  
    public static void main (String args[]) {
```

```

int num = 80;
if(num < 50)
    System.out.println("The number is lesser than 50");
else
    System.out.println("The number is greater than 50");
}

```

```
The number is greater than 50
```

for loop statement

```

public class controlst3 {
    public static void main (String
        args[]) { for (int i = 20; i > 1; i--)
        System.out.println("The number is: " + i);
    }
}

```

```

The number is: 20
The number is: 19
The number is: 18
The number is: 17
The number is: 16
The number is: 15
The number is: 14
The number is: 13
The number is: 12
The number is: 11
The number is: 10
The number is: 9
The number is: 8
The number is: 7
The number is: 6
The number is: 5
The number is: 4
The number is: 3
The number is: 2

```


while loop statement

```
public class controlst4{  
    public static void main (String args[]){
```

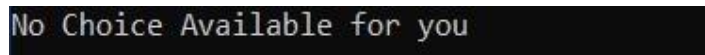
```
int num = 1;
while(num < 10){
    System.out.println(num);
    num++;
}
```

```
1
2
3
4
5
6
7
8
9
```

```
}
```

switch statement

```
public class control5 {  
    public static void main (String  
        args[]) { int num = 6;  
        switch (num + 1)  
        {  
            case 1:  
                System.out.println("Choice 1");  
                break;  
            case 2:  
                System.out.println("Choice 2");  
                break;  
            case 3:  
                System.out.println("Choice 3");  
                break;  
            default:  
                System.out.println("No Choice Available for you");  
                break;  
        }  
    }  
}
```



No Choice Available for you

switch statement

```

public class control5 {
    public static void main (String
args[]) { int num = 2;
    switch (num + 1)
    {
        case 1:
            System.out.println("Choice 1");
            break;
        case 2:
            System.out.println("Choice 2");
            break;
        case 3:
            System.out.println("Choice 3");
            break;
        default:
            System.out.println("No Choice Available for you");
            break;
    }
}

```

```

}
}
}

```

Choice 3

Lab Assignment No. 3

Aim: Program to implement method overriding and method overloading in java

Theory:

Method Overloading

Method overloading is used to increase the readability of the program.

Method overloading is performed within class.

Method overloading is the example of compile time polymorphism.

Method Overriding

Method overriding is used to provide the specific implementation of the method that is already provided by its super class


Method overriding occurs in two classes that have IS-A (inheritance) relationship.

Method overriding is the example of run time polymorphism.

Program:

Method Overloading

```
class adder{
    static int add(int a,int b){return a+b;}
    static int add(int a,int b,int c){return a+b+c;}
}
class Overloading1 {
    public static void main(String[]
args){ System.out.println(adder.add(11,1
1));
System.out.println(adder.add(11,11,11));
}
}
```



Method Overriding

```
class Animal{
    public void move(){
        System.out.println("These are animals");
    }
}

class Dog extends
    Animal{public void
    move(){
        System.out.println("This is Dog");
    }
}

public class overrid{
    public static void main(String
        Args[]){Animal a =new
        Animal(); Animal d =new
        Dog(); a.move();
        d.move();

    }
}
```

```
These are animals
This is Dog
```

```
class Dog extends Animal{
```

Lab Assignment No. 4

Aim: Program to implement Single and Multilevel inheritance in java

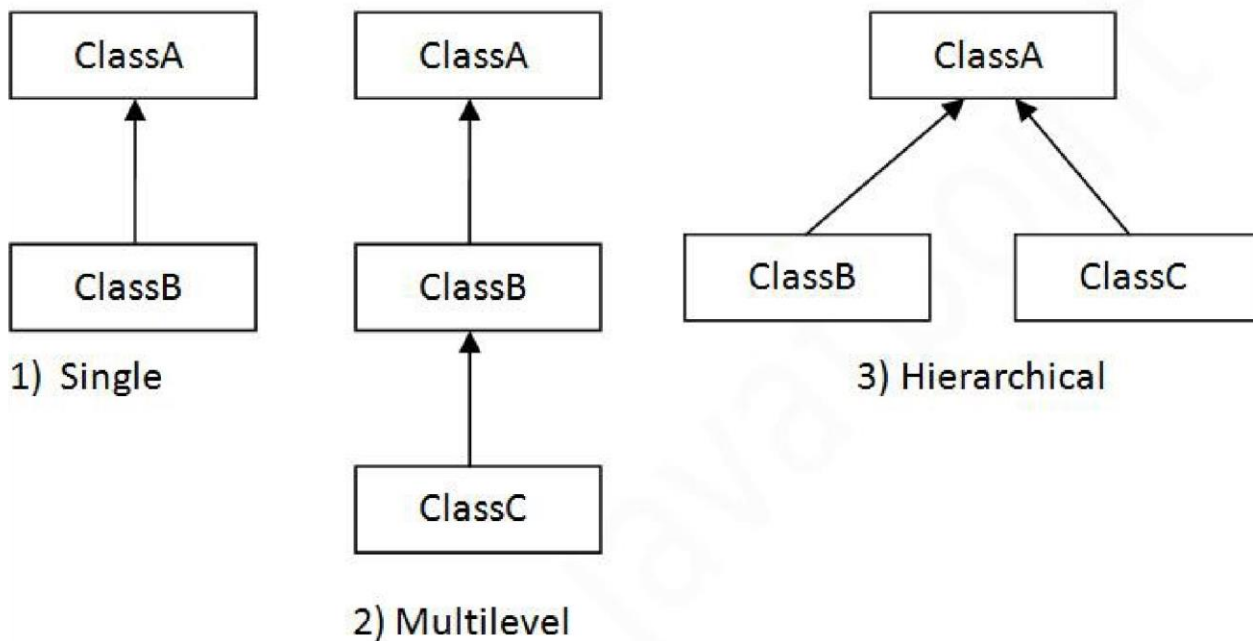
Theory:

Inheritance

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviours of a parent object. It is an important part of OOPs (Object Oriented programming system).

On the basis of class, there can be three types of inheritance in java: single, multilevel and hierarchical.

In java programming, multiple and hybrid inheritance is supported through interface only.



Program:

Single Inheritance

```
class Animal{  
    public void cat(){  
        System.out.println("These are animals");  
    }  
}
```



```
        public void type(){
            System.out.println("This is Dog");
        }
    }

    public class singin{
        public static void main(String
            Args[]){Dog d =new Dog();

            d.cat();
            d.type();

        }
    }
```

```
These are animals
This is Dog
```

Lab Assignment No. 5

Multilevel Inheritance

```
class Animal{
    publicvoid cat(){
        System.out.println("These are animals");
    }
}

class Dog extends
    Animal{publicvoid
    type(){
        System.out.println("This is Dog");
    }
}

class Lab extends Dog{
    publicvoid breed(){
        System.out.println("This breed is Labrador");
    }
}

publicclass multin{
    publicstaticvoid main(String
        Args[]){Lab l =new Lab();
        l.cat();
        l.type();
        l.breed();
    }
}
```

```
These are animals
This is Dog
This breed is Labrador
```

Aim: Program to implement Single and Multilevel inheritance in java

Hierarchical Inheritance

```
class Animal{
    publicvoid cat(){
        System.out.println("These are animals");
    }
}

class Dog extends Animal{
    publicvoid type(){
        System.out.println("This is Dog");
    }
}

class Cat extends Animal{
    publicvoid type(){
        System.out.println("This is Cat");
    }
}

publicclass hierin{
    publicstaticvoid main(String
        Args[]){Dog d =new Dog();
        Cat c =new Cat();
        d.type();
        c.type();
    }
}
```

```
This is Dog
This is Cat
```

Lab Assignment No. 6

Aim: Program to implement Array and its types in java

Theory:

Array: An array is a collection of similar type of elements which has contiguous memory location.

Types of Array in java:

There are two types of array.

- Single Dimensional Array
- Multidimensional Array

1.Single Dimensional Array Program

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

int a[]=new int[5];//declaration and instantiation
a[0]=10;//initialization
a[1]=20;
a[2]=70;
a[3]=40;
a[4]=50;

//printing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);

}}
```

Output:

```
10
20
70
40
50
```

2. Multidimensional Array Program

```
class Testarray3 {
public static void main(String args[]){

//declaring and initializing 2D array
int arr[][]={{1,2,3},{2,4,5},{4,4,5}};

//printing 2D array
for(int i=0;i<3;i++){
for(int j=0;j<3;j++){
    System.out.print(arr[i][j]+" ");
}
System.out.println();
}
```

Output:

```
1 2 3
2 4 5
4 4 5
```

Lab Assignment No. 7

Aim: Program to implement types of constructors in java

Theory:

Constructors

In Java, a constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

It is a special type of method which is used to initialize the object.

A constructor is called "Default Constructor" when it doesn't have any parameter.

Syntax of default constructor:

```
<class_name>(){}
```

A constructor which has a specific number of parameters is called a parameterized constructor.

Program:

Default Constructor (No- Arg Constructor)

```
//Java Program to create and call a default constructor
class Constr{
    //creating a default constructor
    Constr(){System.out.println("New Constructor is created");}
    //main method
    publicstaticvoid main(String args[]){
        //calling a default constructor
        Constr b=new Constr();
    }
}
```

```
New Constructor is created
```

Copy Constructor

```
//Let us see another example of default constructor
} //which displays the default values
class copyc{
```

```
int id;
String name;
copyc (int i, String
      n){id = i;
        name = n;
}

copyc(copyc s){
    id = s.id;
    name = s.name;
}
//method to display the value of id and name
void display(){System.out.println(id+" "+name);
}
publicstaticvoid main(String args[]){
//creating objects
copyc s1=new copyc(7458,"Arvind");
copyc s2=new copyc(6050,"Prathamesh");
//displaying values of the object
s1.display();
s2.display();
}
}
```

```
7458 Arvind
6050 Prathamesh
```

Parameterized Constructor

//Java Program to demonstrate the use of the parameterized constructor.

```
class Parcons{
int id;
    String name;
//creating a parameterized constructor
    Parcons(int i,String n){
        id = i;
        name = n;
    }
//method to display the values
void display(){System.out.println(id+" "+name);}

publicstaticvoid main(String args[]){
//creating objects and passing values
    Parcons s1 =new Parcons(6716,"Piyush");
    Parcons s2 =new Parcons(6100,"Sanika");
//calling method to display the values of object
    s1.display();
    s2.display();
}
}
```

```
6716 Piyush
6100 Sanika
```


Lab Assignment No. 8

Aim: Program to perform abstract classes and methods in java

Theory:

Abstract Class

The abstract class in Java cannot be instantiated (we cannot create objects of abstract classes). We use the abstract keyword to declare an abstract class.

Abstract Method

A method that doesn't have its body is known as an abstract method. We use the same abstract keyword to create abstract methods.

Program:

```
abstractclass shoes{
    abstractvoid sporty();
}
class Nike extends
    shoes{void
    sporty(){
        System.out.println("Just Do It");
    }

    publicstaticvoid main (String
        args[]){shoes obj =new
        Nike(); obj.sporty();
    }
}
```

Just Do It

Lab Assignment No. 9

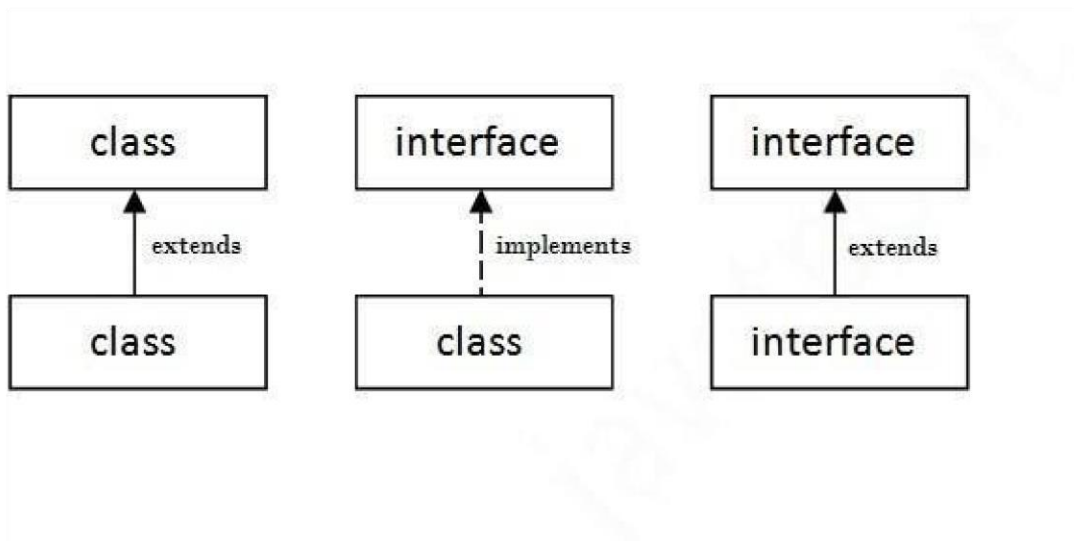
Aim: Program to perform interfaces in java

Theory:

Interfaces

An interface in Java is a blueprint of a class. It has static constants and abstract methods.

The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java.



Program:

Interfaces to class

```
interface shows{
    void show();
}
class Arvind implements
shows{publicvoid
show(){
    System.out.println("Hello, this is a show function");
}
publicstaticvoid main(String
args[]){Arvind obj =new
Arvind(); obj.show();
}
}
```

```
Hello, this is a show function
```

Interfaces to interface (Multiple Inheritance)

```
interface printable{
    void print();
}
interface showable{
    void show();
}
class implement implements printable,
    showable{public void print(){
        System.out.println("Hello, this is a print function");
    }
    public void show(){
        System.out.println("Hello, this is a show function");
    }
    public static void main(String
        args[]){ implement obj =new
        implement();obj.print();
        obj.show();
    }
}
```

```
Hello, this is a print function
Hello, this is a show function
```

Lab Assignment No. 10

Aim: Program to implement exception handling in java

Theory:

Exception Handling

The Exception Handling in Java is one of the powerful mechanism to handle the runtime errors so that the normal flow of the application can be maintained.

Program:

ArithmeticException

```
public class Exception1 {  
    public static void main(String  
        args[]) { try {  
        int a = 100/0;  
        }  
        catch (ArithmeticException  
            a) { System.out.println(a)  
            ;  
        }  
        System.out.println("Ends the code");  
    }  
}
```

```
java.lang.ArithmeticException: / by zero  
Ends the code
```

ArrayIndexOutOfBoundsException

```
public class Exception2 {  
    public static void main(String  
        args[]) { try {  
        int a[] = new int[5];  
        a[10] = 50;  
        }  
        catch (ArrayIndexOutOfBoundsException  
            ar) { System.out.println(ar);  
        }  
    }  
}
```

```
}  
}
```

```
java.lang.ArrayIndexOutOfBoundsException: Index 10 out of bounds for length 5  
Ends Code
```

Lab Assignment – 11

11. Program to demonstrate Applet in Java

Java Applet

Applet is a special type of program that is embedded in the webpage to generate the dynamic content. It runs inside the browser and works at client side.

Simple example of Applet by html file:

To execute the applet by html file, create an applet and compile it. After that create an html file and place the applet code in html file. Now click the html file.

```
1. //First.java
2. import java.applet.Applet;
3. import java.awt.Graphics;
4. public class First extends Applet{
5.
6.     public void paint(Graphics g){
7.         g.drawString("welcome",150,150);
8.     }
9.
10. }
```

myapplet.html

```
1. <html>
2. <body>
3. <applet code="First.class" width="300" height="300">
4. </applet>
5. </body>
6. </html>
```

Write in command prompt:

```
c:\>javac First.java
```

