# SYLLABUS

## PR-3 OBJECT ORIENTED PROGRAMMING USING JAVA

| | | | |
|---|---|---|---|
| **Total Period:** | **60** | **Examination:** | **3 hr** |
| **Lab. periods:** | **4P/Week** | **Term Work:** | **25** |
| **Maximum marks:** | **50** | **End Semester Examination** | **50** |

List of Practicals:-

1. Write a Java program to print 'Hello' on screen and then print your name on a separate line.
2. Write a Java program to print the sum of two numbers.

3. Write a Java program that takes a number as input and prints its multiplication table upto 10.

4. Write a Java program to print the area and perimeter of a rectangle.

**5.** Write a Java program to swap two variables.

**6.** Write a Java program to convert a decimal number to binary number.

**7.** Write a Java program to compare two numbers.

**8.** Write a Java program and compute the sum of the digits of an integer.

**9.** Write a Java program to count the letters, spaces, numbers and other characters of an input string.

**10.** Write a Java program to reverse a string.

**11.** Write a Java program to accept a number and check the number is even or not. Prints 1 if the number is even or 0 if the number is odd.

12. Write a Java program that accepts two integer values from the user and return the larger values. However if the two values are the same, return 0 and return the smaller value if the two values have the same remainder when divided by 6.

**13.** Write a Java program to get the larger value between first and last element of an array (length 3) of integers .

**14.** Design a class to represent a bank account. Include the following members :

Data members:

- Name of the depositor
- Account Number
- Type of account
- Balance amount in the account

Methods:

- To assign initial values
- To deposit an amount
- To withdraw an amount
- To display the name and balance

15. Given are two one-dimensional arrays, A and B which are sorted in ascending order. Write a program to merge them into a single sorted array C that contains every item from arrays A and B, in ascending order.

16. Write a java program implementing multiple inheritance.

17. Write a java program implementing package.

18. Write a java program to read a file line by line and print the line on the output screen.

19. Write a java program to read content from one file and write it into another file.

20. Define an exception called "No Match Exception" that is thrown when a string is not equal to "India". Write a program that uses this exception.

21. Develop a java project for percentage calculator/temperature conversion tool.

**Books Recommended:-**

| Sl.No | Name of Authors | Title of the Book | Name of Publisher: |
|---|---|---|---|
| 01 | E. Balagurusami | Programming With Java: A Primer | The McGraw-Hill Companies |
| 02 | Patric Naughton Herbert Schildt | Java™ 2: The Complete Reference | Tata McGraw-Hill Publishing Company Limited |
| 03 | Rashmi Kanta Das | Core Java For Beginners | Vikas Publishing |
| 04 | Herbert Schildt | Java: A Beginner's Guide | McGraw-Hill Education |
| 05 | Cay S. Horstmann | Core Java Volume I - Fundamentals | Prentice Hall |

## What is JAVA ?

**Java** is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers *write once, run anywhere* (WORA), meaning that compiled Java code can run on all platforms that support Java without the need to recompile. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. Java is a comprehensive documentation system, created by Sun Microsystems. It provides developers with an organized system for documenting their code. Javadoc comments have an extra asterisk at the beginning, i.e. the delimiters are /** and */, whereas the normal multi-line comments in Java are delimited by /* and */, and single-line comments start with //.

## History of JAVA ?

James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991. Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called *Oak* after an oak tree that stood outside Gosling's office. Later the project went by the name *Green* and was finally renamed *Java*, from Java coffee, a type of coffee from Indonesia. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

## Principles of developing JAVA language

There were five primary goals in creating the Java language:

1. It must be simple, object-oriented, and familiar.
2. It must be robust and secure.
3. It must be architecture-neutral and portable.
4. It must execute with high performance.
5. It must be interpreted, threaded, and dynamic.

## There are mainly 4 types of applications that can be created using Java programming:

### 1) Standalone Application

Standalone applications are also known as desktop applications or window -

based applications. These are traditional software that we need to install on every

machine. Examples of standalone application are Media player, antivirus, etc. AWT and

Swing are used in Java for creating standalone applications.

**2) Web Application**

An application that runs on the server side and creates a dynamic page is

called a web application. Currently, Servlet, JSP, Struts, Spring, Hibernate, JSF, etc.

technologies are used for creating web applications in Java.

**3) Enterprise Application**

An application that is distributed in nature, such as banking applications, etc.

is called enterprise application. It has advantages of the high -level security, load

balancing, and clustering. In Java, EJB is used for creating enterprise applications.

**4) Mobile Application**

An application which is created for mobile devices is called a mobile

application. Currently, Android and Java ME are used for creating mobile applications.

## Advantages of JAVA language

- **Platform independent:** Java code can run on any platform that has a **Java Virtual Machine (JVM)** installed, which means that applications can be written once and run on any device.
- **Object-Oriented:** Java is an object-oriented programming language, which means that it follows the principles of encapsulation, inheritance, and polymorphism.
    - **Security:** Java has built-in security features that make it a secure platform for developing applications, such as automatic memory management and type checking.
    - **Large community:** Java has a large and active community of developers, which means that there is a lot of support available for learning and using the language.
    - **Enterprise-level applications:** Java is widely used for developing enterprise-level applications, such as web applications, e-commerce systems, and database systems

## Disadvantages of JAVA language

1. **Performance:** Java can be slower compared to other programming languages, such as C++, due to its use of a virtual machine and automatic memory management.
2. **Memory management:** Java's automatic memory management can lead to slower performance and increased memory usage, which can be a drawback for some applications.

# Execution Model of Java

Ø Five phases in Java Programs

Ø In the figure below, 5 phases of the Java Program are described clearly with edit, compile, load, verify and execute.

**Ø Phase 1: Edit**

Ø We create the program on editor, after that it stored in the disk with the name's ending .java

**Ø Phase 2: Compile javac (java compiler**)

Ø Compiler translate from high -level language program to byte codes and store it in disk with the ending name .class.

**Ø Phase 3: Load**

Ø Class loader compile read and put those byte codes from disk to Primary Memory.

**Ø Phase 4: Verify**

Ø Verify byte codes to confirm that all byte codes are valid and do not risk for the Java's security restrictions.

**Ø Phase 5: Execute**

Ø Java Virtual Machine (JVM) read and translates those byte codes to language that computer can understand (Machine Language). Then execute the program, store it values in primary memory.

## Conclusion

**Java** is a powerful and versatile **programming language** that's great for beginners and experienced developers alike. By learning the basics, like what **classes**, **objects**, and **methods** are, you can start creating your own programs and see how **Java** can be used in real-world applications. Whether you're interested in building games, mobile apps, or websites, **Java** provides a solid foundation to bring your ideas to life. Remember, the more you practice, the better you'll get. Keep exploring, and soon you'll be writing your own **Java programs** with confidence!

**1.2 What is object-oriented programming (OOP)?**

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic. An object can be defined as a data field that has unique attributes and behaviour. Or Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects.

What is Object?

Object means a real-world entity such as a pen, chair, table, computer, watch, etc.

Why OOP?

focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. This approach to programming is well-suited for programs that are large, complex and actively updated or maintained.

**1.3 OOP Concepts and Terminology**

o Object

o Class

o Inheritance

o Polymorphism

o Abstraction

o Encapsulation

o Message Passing

o Dynamic Binding

**1. Object**

Any entity that has state and behaviour is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical.

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

Example: A dog is an object because it has states like colour, name, breed, etc. as well as behaviour like wagging the tail, barking, eating, etc.

**2. Class**

Collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object.

Class doesn't consume any space.

**3. Inheritance**

When one object acquires all the properties and behaviours of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

**4. Polymorphism**

If one task is performed in different ways, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.

In Java, we use method overloading and method overriding to achieve polymorphism.

Another example can be to speak something; for example, a cat speaks meow, dog barks woof, etc.

**5. Abstraction**

Hiding internal details and showing functionality is known as abstraction. For example, phone call, we don't know the internal processing.

In Java, we use abstract class and interface to achieve abstraction.

**6. Encapsulation**

Binding (or wrapping) code and data together into a single unit are known as encapsulation.

For example, a capsule, it is wrapped with different medicines.

A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

**7. Message Passing**:

Objects communicate with one another by sending and receiving information to each other. A message for an object is a request for execution of a procedure and therefore will invoke a

function in the receiving object that generates the desired results. Message passing involves specifying the name of the object, the name of the function and the information to be sent.

**8. Dynamic Binding**

Dynamic binding also called dynamic dispatch is the process of linking procedure call to a specific sequence of code (method) at run-time. It means that the code to be executed for a specific procedure call is not known until run-time. Dynamic binding is also known as late binding or run-time binding.

## EXPERIMENT- 1

**AIM OF THE EXPERIMENT -**

Wap in Java to print “Hello” on screen and then print your name on a separate line.

```
Class Abc
{
public static void main (String args[])
   {
System.out.println("Hello");
System.out.println("Ayushmita Jena");
   }
}
```

OUTPUT –
Hello

Ayushmita Jena

## EXPERIMENT-2

**AIM OF THE EXPERIMENT**

wap in java to print the sum of two number

```
class sum
{
int a,b;
A=10;b=30; int
sum =a+b
System.out.println(“sum of ”+a+”&”+b+”is”+sum);
}
}
```

OUTPUT- sum of
10&30 is 40

## EXPERIMENT- 3

**AIM OF THE EXPERIMENT -**

Write a program that take a number as input and print its multiplication table uoto 10.

```
Import java.util.*; class Table
{
public static void main (String args[])
{
Scanner sc=new Scanner (System.in)
Intn,I;
System.out,println(“Enter a no:”);
n=sc.nrxtInt(); for (i=1; i=10; i++)
System.out,println(n+ ”*” + I + ”=” + n*I );
}
}
```

OUTPUT:

Enter a no:10

10*1=10

10*2=20

10*3=30

10*4=40

10*5=50

10*6=60

10*7=70

10*8=80

10*9=90

10*10=100

## EXPERIMENT- 4

**AIM OF THE EXPERIMENT -**

Write a program to print the area & perimeter of a circle.

```
import  java.util.Scanner;
class circle
   {
 Public static void main(String args[ ])
    {
  Scanner sc=new Scanner ();
  System.out.println("Enter the  value for radius");
  int r = sc.nextInt();
   doublearea,per;
   area=3.14*r*r;
   per=2* 3.14*r;
   System.out.println("Perimeter ="+per+"Area="+area);
  }
}
```

OUTPUT:-

Enter the value for radius

3

Perimeter= 18.84 A

## Experiment – 5

**AIM OF THE EXPERIMENT** – Write a program to swap 2 variables

```
public class SwapVariables {


 public static void main(String[] args) {
int a = 5;
 int b = 10;


 System.out.println("Before swap: a = " + a + ", b = " + b);


// Swapping without a temporary variable
 a = a + b;
 b = a - b;
  a = a - b;


  System.out.println("After swap: a = " + a + ", b = " + b);
 }
}
```

## EXPERIMENT- 6

**AIM OF THE EXPERIMENT -**

Write a program to convert decimal number to binary number.

```
Import java.util.* ;
Class Dectobinary
{
Public static void main(String args[])
{
```

```
int n,ans=0,power=1,rem,temp; Scanner
sc=new Scanner(System.in);
System.out.println(“Enter decimal no:”):
N=sc.nextInt():temp=n;
While(n>0)
{
Rem=n%2;
Ans+=rem*power;
Power*=10; n=n/2;
}
Systerm.out.println(“The binary of”+temp+”is”+ans);
}
}
```

OUTPUT:-

Enter decimal no:12

The binary of 12 is 1100

## EXPERIMENT- 7

**AIM OF THE EXPERIMENT -** Write a program to compare 2 numbers.

```
Import java.util.Scanner;
Class Compare
{
Public static void main(String[ ]  args)
{
Inta,b;
Scanner sc=new Scanner();
System.out.println(“Enter 1st no.”);
```

```
a=sc.nextInt();
System.out.println("Enter 2nd no.");
b=sc.nextInt(); if(a>b)
System.out.println(a+ "is greater than"+b); else
System.out.println(b+"is greater than"+a);
} }
```

O/p:- Enter

1st no.

10

Enter 2nd no.

20

20 is greater than 10

## EXPERIMENT- 8

**AIM OF THE EXPERIMENT -**

Write a java program to compute the sum of the digits of an integer.

```
importjava.util.Scanner; class Sum
{
public static void main(String [] args)
  {
intn,r,sum=0,t;
     Scanner sc=new Scanner(System.in); System.out.println("Enter
the number");
     n=sc.nextInt();
t=n;
while(n!=0)
     {
        r=n%10;
```

```
sum+=r;
        n=n/10;
      }
System.out.println("Sum of digits of "+t+"is"+sum);
   }
}
```

O/P:-

Enter the number

3142

Sum of digits of 3142 is 10

## EXPERIMENT- 9

**AIM OF THE EXPERIMENT -**

Write a program to count the letter, space, number &othe characters of an input string.

```
importjava.util.Scanner; classStrlen
{
public static void main(String [] args)
   {
      Scanner sc=new Scanner(System.in);
System.out.println("Enter string");
String       s=sc.nextLine();       int
n=s.length();
System.out.println(n);
   }
}
```

O/P
Enter string: chocoshake @3435
16

## EXPERIMENT- 10

**AIM OF THE EXPERIMENT -**

Write a program to reverse a string. class

Reverse

{

public static void main(String [] args)

{

StringBuilderstr=new StringBuilder("Abhishek"); str.reverse();

System.out.println(str);

}

}

O/P:- kehsihbA

## EXPERIMENT- 11

**AIM OF THE EXPERIMENT -**

Write a program in java to accept a number & check the number is even or not print 1 if the number is even or 0 if the number is odd.

Import java.util.*; public class Num check

{

Public static void main(String args[])

{

Scanner sc=new Scanner();

System.out.println(“Enter a number”);

Int n=sc.nextInt();

If(n%2==0)

System.out.println(“1”); else

```
                System.out.println("0");
            }
          }
```

O/P:-

Enter a number

14

1

## EXPERIMENT- 12

**AIM OF THE EXPERIMENT -**

**Write a program to accept 2 integer value from the user & return the larger value. However if the 2 value are the same return the smaller value if the 2 values have the same reminder when devided by 6.**

```
Import java.util.*;
Public class Ex12          `
{
Public static void main(String args [])
{
Scanner sc=new Scanner (System.in)
System.out.println("Enter 1st no. :")
Int a=sc.nextInt();
System.out.println("Enter 2nd no. :")
Intb=sc.nextInt();
System.out.println("Result :" +result (a,b));
}
Public staticint result (intx ,int y)
{
If (x==y)
Retun 0;
```

If (x%6==y%6)

Return(x<y)?X:y; Return(x>y)?X:y;

o/p :-

Enter 1st no. : 6

Enter 2nd no. :12

6

## EXPERIMENT- 13

**AIM OF THE EXPERIMENT -**

**Write a program to get larger value between first,last element of an array(length 3) of integers.**

importjava.util.*;

Public static void main(String args[])

{ int

I;

int a[]=new int[3];

System.out.println("Enter elements"); Scanner

se=new Scanner(System.in);

for(i=0 ; i<3;  i++) a[i]=sc.nextInt();

System.out.println("Array is:");

for(i=0; i<3; i++)

System.out.println(a[i]); int

max=a [10]; if (a[2]>=max)

max=a[2];

System.out.println("Larger value between first & last element" + max);

}

}

O/P:- Enter
element:

3 2 9 Arrya
is:
3 2 9
Larger value between first & last element is 9.

## EXPERIMENT- 14

**AIM OF THE EXPERIMENT -**

Design a class to represent a bank account includingmembers:-

- Name of depositor
- Account number
- Type of account
- Balance amount in the account

Methods –

- To assign initial values
- To deposit an amount
- To withdraw an amount
- To display the name and balance

```
Import java.util.*;
Class bank
{
String name,type ;
Int acc.no.;
Double balance;
Void input()
{
Scanner sc=new scanner(system.in);
System.out.println(“enter name “);
Name=sc.nextline();
System.out.println(“enter type:”);
Type=sc.nextline();
System.out.println(“enter account no.”);
Acc.no.=sc.nextint();
System.out.println(“enter balance:”);
Balance= sc.nextdouble();
}
Void deposite()
{
Scanner sc =new scanner(system.in);
System.out.println(“amount to be deposited:”);
Double amount=sc.nextdouble();
```

```
Balance+=amount;
System.out.println(“total balance=” +balance);

}
Void withdrawl()
{
Scanner sc=new scanner ( system.in);
System.out.println(“amount to be credited:”);
Double amount=sc.nextdouble();
Balance-=amount;
System.out.println(“total balance=”+balance);
}
Void display()
}
System.out.println(“name=”+name);
System.out.println(“balance=” +balance);
}
Public static void main (string args[]);
{
Bank a= new bank(); a.input();
a.deposite();
a.withdrawl();
a.display();
}
}
```

O/P-Enter your name: Abhisek
Enter type :major
Enter account no-: 36094570293
Enter balance: 40000
Amount to be deposited:20000
Total balance: 60000
Amount to be credited: 10000
Total balance: 50000
Name :Abhisek
Blance: 50000

## EXPERIMENT- 15

**AIM OF THE EXPERIMENT**

Given are two one-decimal arrays,A & B which are sorted in assending order.Wap to merge them into a single sorted array c that contains every item from arrays A & B in assending order.

```
Class merge Array{
```

```
Public static int []  mergearray (int[] a,int [] b){
Int [] c=new int[a.length+b.length];
Int i=0,j=0,k=0;
While(i<a.length && j<b.length)
{
If(a[i] <b[j])
{
C[k] = a[i];
I++;
K++;
}
Else{
C[k] = b[j];
J++;
K++;
}
}
While(i<a.length){
C[k] = a[i];
I++;
K++;
}
While(j<b.length){
C[k]= b[j];
J++;
K++;
}
```

```
retuen c;
}
Publc static void main(string[]args){
Int[]a=new int[]{-7,12,17,29,41,56,79}; Int[]b=new int[] {-9,-3,0,5,19}
Int[]c=mergearray(a,b);
System.out.println("Array A:"+Arrays.tostring(a));
System.out.println("Array B:"+Array.to string (b));
System.out.println("Merged Array:"+Array.to.string(c));
}
}
```

**o/p:-**

```
Array A:[-7,12,17,29,41,56,79]
Array B [-9,-3,0,5,19]
Merged Array[-9,-7,-3,0,5,12,17,19,29,41,56,79]
```

## EXPERIMENT- 16

**AIM OF THE EXPERIMENT**

Write a program in java implementing multiple inheritance .

```
 Interface Backend
{
Public void connectserver( );
}
Class Frontend
{
Public void responsive(string str)
{
System.out.println(str +"can also be used as frontend")
} }
```

```
Class language extends frontend implement Backend
{
  String language =”java”
Public void connect server( )
{
System.out.println(“language + “ can be used as backend language”);
Public static void main (string args[ ])
{
Language java = new language( );
Java connectserver( );
Java.responsive(java.language);
}
```

## EXPERIMENT- 17

**AIM OF THE EXPERIMENT**

**Write a program in java implementing package**.

```
package pack ; public
class A
{
public void msg ()
{
system.out.println(“Hello”);
}
packagemypack;
import pack.*; class
B
{
public static void main(string args[])
```

```
{
A obj = new A(); obj.msg();
}
}
```

## EXPERIMENT- 18

**AIM OF THE EXPERIMENT**

**Write a program in java to read a file line by line and print the line output on the screen.**

```
Import java.in.*;
Public class Read
{
Public static void main (String args[ ] )
{
Try
{
file file= new File(“Demo.txt”);
File reader fr=new File reader(File);
Bufferedreaderbr=new Bufferedreader(fr);
String Buffer b=new String Buffer()
String line;
While(line=br.readline()):=null)
{
Sb. append(line); Sb.
append(“\n”);
fr. close();
System.out.println(“Content of file”);
System.out.println(“sb. toString());
```

}

## EXPERIMENT- 19

**AIM OF THE EXPERIMENT**

**Write a program in java to read content from one file and write in on another file.**

Import java.io.*; class File1 public static void main (string args[])

{

File inf = new file(“in.data);

File outf = ne file(“out.data”);

File reader ins = null; File

writer outs = null; try{ ins =

new file reader(inf) outs =

new file reader(outf); intch;

while(ch=ins.read())! = -1

{

outs.write(ch);

}}

Catch(10 Exception e){

System.out.println(c);

System.exit(-1);

}

finaly{ try{

ins.close();

outs.close();

}

## EXPERIMENT- 20

**AIM OF THE EXPERIMENT**

**Define an exception called "No match Exception" that is thrown when a string is not equal to "India" WAP that uses this exception.**

```
Class No match exception {

String & ;

No match exception(string &){

This.& = &;

If(s.equals("India)){

System.out.println("Matched");

}

else {

throus new No Match exception ("Not matched");

} } }

Class Nomatchx {

Public static void main(string[ ]args){

No match exception V=new No match exception ("America");

} }
```

## Experiment -21

**AIM OF THE EXPERIMENT**

**Write a program to Develop a java project for percentage calculator/temperature conversion tool.**

```
public class UtilityTool {

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

```
        System.out.println("Percentage Calculator and Temperature Conversion Tool");

        // Example inputs for percentage calculation
        double totalMarks = 500;
        double obtainedMarks = 430;
        calculatePercentage(totalMarks, obtainedMarks);

        // Example input for temperature conversion
        double celsius = 25;
        convertCelsiusToFahrenheit(celsius);

        double fahrenheit = 77;
        convertFahrenheitToCelsius(fahrenheit);
    }

    // Method for percentage calculation
    public static void calculatePercentage(double totalMarks, double obtainedMarks) {
        if (totalMarks > 0) {
            double percentage = (obtainedMarks / totalMarks) * 100;
            System.out.printf("Percentage: %.2f%%\n", percentage);
        } else {
            System.out.println("Total marks must be greater than zero.");
        }
    }

    // Method to convert Celsius to Fahrenheit
    public static void convertCelsiusToFahrenheit(double celsius) {
        double fahrenheit = (celsius * 9 / 5) + 32;
        System.out.printf("Celsius to Fahrenheit: %.2f°C = %.2f°F\n", celsius, fahrenheit);
    }

    // Method to convert Fahrenheit to Celsius
    public static void convertFahrenheitToCelsius(double fahrenheit) {
        double celsius = (fahrenheit - 32) * 5 / 9;
        System.out.printf("Fahrenheit to Celsius: %.2f°F = %.2f°C\n", fahrenheit, celsius);
    }
}
```