



# **Java Handwritten Notes**

# ADVANCED PROGRAMMING

## Tutorial-1

1. Diff b/w languages  
Website of Indian Railway

## Lecture-1.

### Introduction

#### # JAVA

- \* programming lang.
- \* 1995 (Oak)
- \* 1995 (JAVA) James Gosling
- \* platform Independent (Byte codes)
- \* OOPS Concept Used
- \* Don't have pointers in JAVA becoz complexity less,  
Security
- \* Secured lang.
- \* It is simple
- \* Concepts based on real life problems
- \* Types of Java application : →  
We can design basically 4 applications in  
JAVA
  - \* Stand Alone Applications (desktop) e.g. - Media player
  - \* Web Applications e.g. - Indian Railway
  - \* Enterprise Application e.g. - Mgmt.  
↓  
Java beans
  - \* Mobile Applications e.g. - Android



\* Standalone  
There are also known as desktop applications or window based application i.e. - an application we need to install on every machine such as antivirus, media players etc. Awt and Swings are used in java for creating standalone applications.

\* Web  
An application that runs on the server site & creates dynamic web pages is called as web app. Servlets, jsp, struts technology are used in java.

\* Enterprise  
An application i.e. distributed in nature such as banking app etc. In java EJB (Enterprise Java Bean) is used for creating enterprise application.

\* Mobile  
An application i.e. created for mobile devices currently android & JAVA & F are used to creating mobile app.

Imp What is JAVA?

JAVA is a general object oriented programming language & a <sup>bcuz of JRE</sup> computing platform developed by "James Gosling" of Sun micro system in 1995.

Why Java?

Java is Secure

It is platform independent

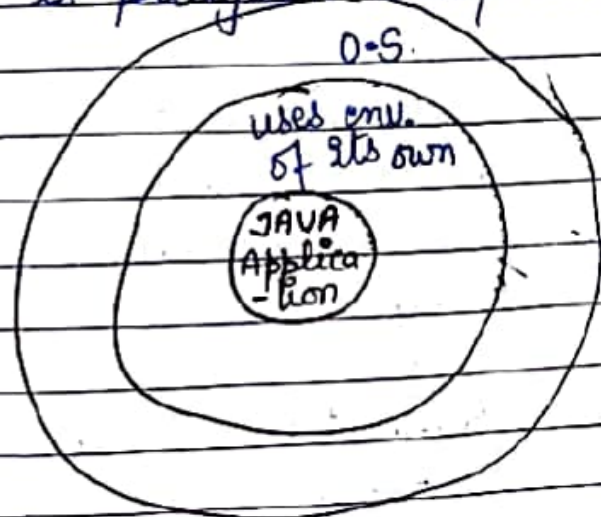
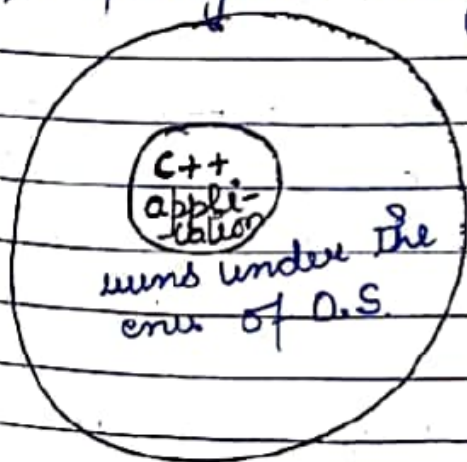
Java is portable



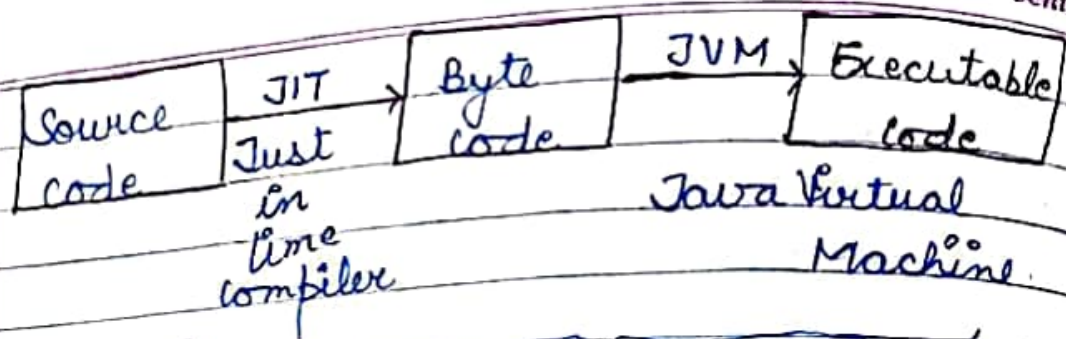
JVM	library files	JVM + library files + development Kit
-----	---------------	---------------------------------------

JVM: → It is an abstract machine, it is specification that provide run time environment in which JAVA byte code can be executed.

JVM's are available almost for many hardware & SW platform i.e. java is platform independent.







## \* Main Features of JAVA : →

1. **Simple** : → Java is simple bcoz most of the concepts has been taken from C++, it is very easy to learn bcoz.
  - \* it does not use any header file.
  - \* it eliminated the use of pointers
  - \* operator overloading & virtual base classes eliminated.
2. **Object Oriented** : → Java is pure Object Oriented programming lang. Everything in java is an object, all programs & data resides in objects & classes.
3. **Distributed** : → Java has network facilities it enables multiple programmers at remote locations to work together on a single project.
4. **Robust** : → Java Virtually eliminates the problem of memory deallocation by using garbage collection for unused object. Moreover run time errors are managed by exception.







Reusability: → is an aspect of OOP paradigm  
 JAVA supports this concept i.e. JAVA classes can be reused in several ways.

It is always nice if we could use something that already exist rather than creating the same thing all over again.

4. The inheritance allows sub class to inherit all the variables & methods of their parent class.

Inheritance may take d/f forms

- 1) Single inheritance (only one super class)
- 2) Multilevel inheritance (derived from derived class)
- 3) Multiple inheritance (several super classes)
- 4) Hierarchical (one super class & many sub classes)

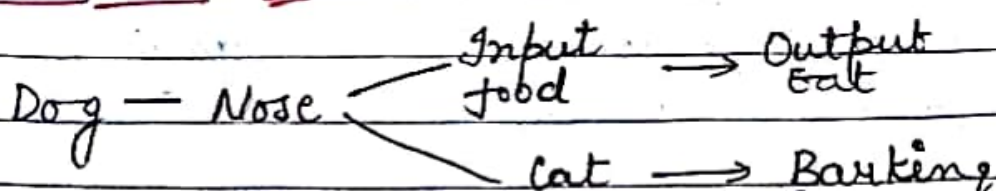
There is no multiple inheritance in the JAVA but we can implement multiple inheritance through interfaces.

5. Polymorphism: →

↓ forms / behaviour  
 many

It is a greek word poly & morphism i.e. same interface acting differently w/d d/f inputs.

Polymorphism Ex -



It is a mechanism by which some interface is used for general class of action but depending upon d d/f inputs d/f outputs.

are retrieve  
(Same interface acting differently w/d  
d/f inputs)

### 3. Encapsulation: →

It is a mechanism by which data members i.e. member function & variables are enclosed into a single entity called class to protect from outside world for any interference.

Ex - Mobile phone having d/f features combine in one, class having Students combine in one become CSE

Imp

### 6. Diff b/w Data Abstraction & Data Hiding

1. In Data Abstraction  
it is all about  
hiding complexity

1. In Data Hiding  
it is all about  
providing security  
to data.

2. It means no need  
to show how comple-  
-xated steps u have  
perform to do a  
particular operation

2. It is making inaccess-  
-ible certain details  
i.e. just hiding the  
data so that it is  
not exposed.

It's a philosophical  
concept i.e. almost  
everything a good



developer writes in  
abstraction

Ex- Just to hide the  
complexity as such  
& in

D.A

Ex- Working of an engine

Data hiding U are  
hiding just to  
keep ur data  
safe as it may  
affect the other  
data.

Ex:- D.H.

Passwords,  
college data  
i.e. It is  
available to  
authorised  
members not to  
everyone.

\* N/t b/w C++ & JAVA

C++

1. C++ is basically C w/ d  
extended Object Oriented  
extension.

2. It implements the  
concepts of multiple  
inheritence

3. In C++ we use  
pointers.

JAVA

1. Java is purely OOP lang.

2. Java does not support  
multiple inheritance  
of classes.

3. There is no use of  
pointers.

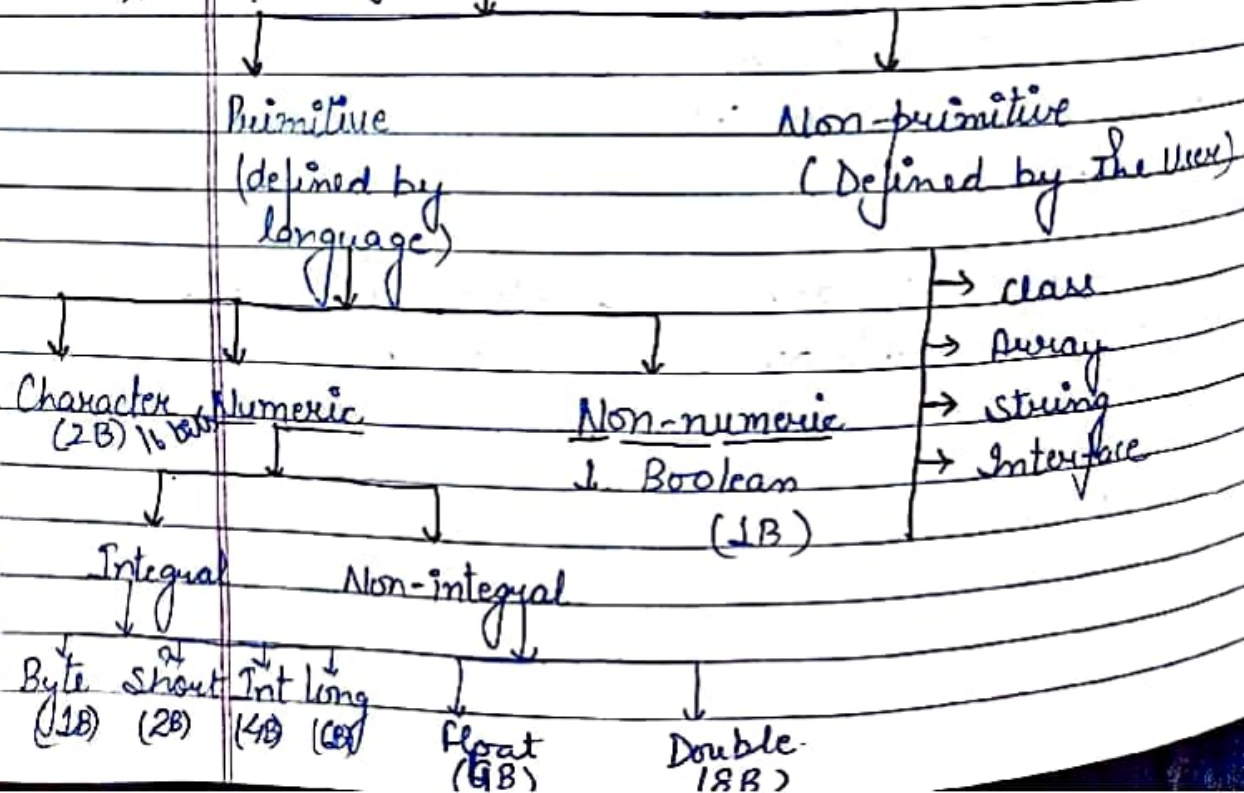
C++

4. In C++ we have destructor
5. In C++ we use header files.
6. There is Operator Overloading in C++.
7. In C++ we use global variable.
8. In C++ <sup>there is a concept</sup> we have of template classes.
4. Java replaced destructor <sup>the</sup> with finalized method.
5. There is no use of header files in Java
6. There is no Operator Overloading in Java.
7. In Java there is no use of global variable.
8. It does not have template classes as in C++.

27/8

\* 2 Marks

Data Types in Java.





(Short - Big)

## \* Type Conversion

- \* In some case it might want 2 assign value of one data type to variable of another type
- \* If both the source & destination types are compatible then JAVA performs the conversion.

## \* JAVA automatic conversion

JAVA automatically converts one type to another only when the following 2 conditions are satisfied.

1. Both types are compatible with each other.
2. Size of destination type is more than the source type.

When the above two conditions are satisfied then Java performs "implicit conversion". It is also known as "widening conversion".

## \* Type casting "Narrowing" (Big - Small)

If we want to convert two types which are incompatible size of destination type is less than the size of source type, then the conversion is done "explicitly". This process is known as Type casting. Ex - If we want to convert integer value through byte value Java cannot do this automatically as the size of int is.

Double → float → int → long → Byte      int i;  
Byte = (destination type)      i = float(i)

$a = 15$      $b = 5$      $c = 10$

if  $(a > b)$

    A

else

    B

15 10 if  
 $(a > b ? a : b)$

? : equivalent  
 to if else  
 statement

$(a > b ? (a > c ? a : c) : (b > c ? b : c))$

1/9

int num = 5

Integer num = new Integer(5);

Float

Instance of class

Double

## \* Object

It is a thing through which we can interact we can send messages to objects it is a physical entity

Every object has its own state, behaviour & Identity.

## State

## Object

### \* State

(Value)



\* what object has



\* It is defined by the value that variable contains

### Behaviour

(functionality)



what object can perform



It is defined by the func. of class

### Identity

(Reference)



to identify the object



he can identify an object by its name.



## \* class

It is a user defined data type which is a collection of objects.

- It contains member variables & member func.
- Values are assign to objects & to variables. It acts as a template for objects.

## 3/9 Types of Variables in JAVA

3 types of Variables in JAVA

1. local
2. Instance
3. Static

\* Variables that will be declare inside any func. that will be known as local variables.

\* Variables declare outside any func. that will be known as Instance variables.

\* Variables declare outside any func. with a keyword static is known as static variables.

class Cse

{

public static void main (String arg [])

{

int num1 = 5, num2 = 10, sum = 0

sum = num1 + num2

System.out.println ("sum is" + sum);

} }

Ex

class Cse

{

public static void main(String arg[]) <sup>Command line arguments</sup>

{

int num1, num2; Double num3  
num1 = Integer.parseInt(arg[0]);  
<sub>reference class</sub>

parsing of arg[0]

num2 = Integer.parseInt(arg[1]);

num3 = Double.parseDouble(arg[2]);

int sum = num1 + num2;

Compile  
Run

javac Cse.java

java Cse 5 10 10.56

Sum is 15

5/9

Example Create an object of the class

class Rectangle <sup>(file Name - Rectangle.java)</sup>

{

int length, breadth;

Rectangle()

{

length = 10;

breadth = 20;

}

void area()

{ int area = length \* breadth;

class RectangleMain

{

psum(String arg[])



{

Rectangle obj = new Rectangle();  
 obj.area();

}

15/9 How to create a Simple class.

class Area

{

int length, breadth; int area;  
 void area();

{

length = 10;

breadth = 20;

area = length \* breadth;

}

class AreaMain

{

psvm (String arg[])

Area obj = new Area(), // object created

obj.length = 10;

obj.breadth = 20;

int area = obj.length \* obj.breadth;

obj.area;

S.O.pln ("Area is" + obj.area);

How to create Constructor

class Area

{

int length, breadth, int area;  
 Area();

{

length = 10;

```
    breadth = 20;
}
```

```
class AreaMain
{
```

```
    psum()
    Area obj = new Area();
```

How to pass parameters in the fn.

Ex class Area.

```
{
    int length, breadth, int area;
    void area (int 10l, int 20b)
}
```

```
    length = l;
    breadth = b;
}
```

```
class AreaMain
{
```

```
    psum()
    Area obj = new Area();
    obj.area (10, 20);
```

This keyword is used when any ambiguity is exist b/w the local & instance variable.

```
class Area.
```

```
{
    int length, breadth, int area;
```

instance variable

in void area (int length, int breadth)

```
{
    this.length = length;
```



this.breadth = breadth;

}

class Area Main

{

psum ()

Area obj = new Area (); obj created

obj.area (10, 20); call obj

obj.area =

## \* This Keyword

It is a special keyword in JAVA which is used to refer to the current <sup>object or</sup> instance variable of any particular class.

If there is any ambiguity b/w the instance variable & the parameters pass, this keyword is used to resolve the ambiguity.

## \* Method Overloading

Same func. name but  
diff parameters.

{

int length, breadth, int area;

void area (int l, int b)

{

length = 10;  
breadth = 20;

area = length \* breadth  
s.o pln ("Rectangle:" area);

```
}  
void area (int s) // square.
```

```
{  
    area = lengths * lengths;  
}
```

Ex class Area Main

```
{
```

```
    psun ( )
```

```
    Area obj = new Area();
```

```
    obj.area(); // Rectangle
```

```
    obj.area(); // Square.
```

Ex

```
class Employee
```

```
{
```

```
    int id;
```

```
    String name, address;
```

```
    double salary;
```

```
Employee (int i, String n, String a, double s)
```

```
{  
    id = i;
```

```
    name = n;
```

```
    address = a;
```

```
    salary = s;  
}
```



}

void display()

Class Employee Main

{

  psum

}

Employee obj1 = new Employee

101, "Loyal", "#123", 50,000);

Employee obj2 = new Employee

110, "Pia", "#23", 25,000);

obj1 display();

Q. Write a program to calculate factorial of the no. using recursion.

Q. Fibonacci series.

\* Inheritance

Ex class Parent

{

  int num1 = 10;

}

Class Child extends Parent

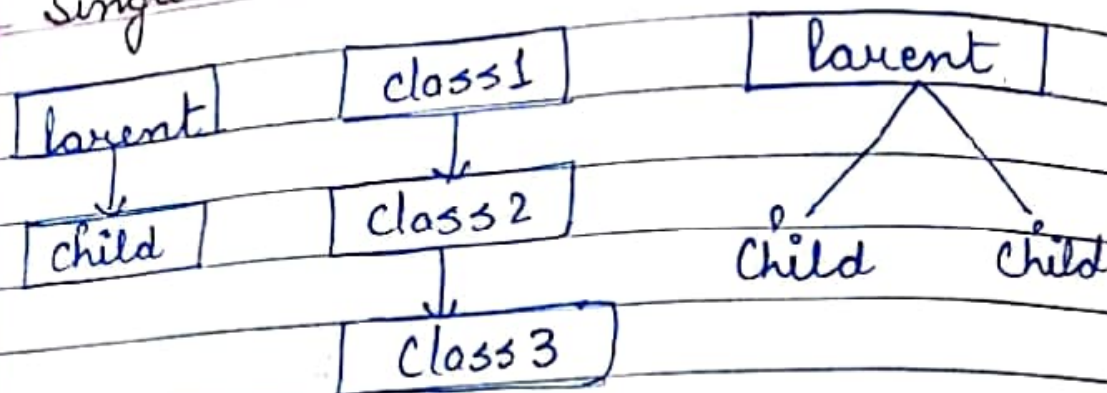
{

  int num2

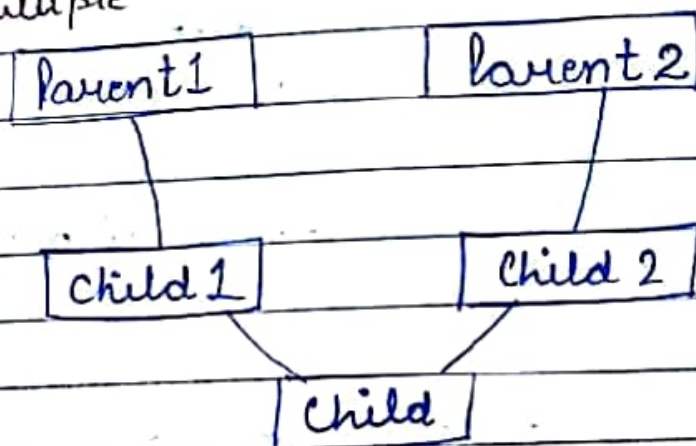
  num2 = num1 + 10;

}

Single



Multiple





A dark, blurry night photograph of a city street. The background is filled with out-of-focus lights in various colors, including white, yellow, and blue, suggesting a busy urban environment at night. The lights are scattered across the upper half of the image, with some appearing as distinct points of light and others as soft, glowing bokeh. The overall tone is dark and moody.

THANK  
YOU