ADVANCED PROGRAMMING

Tutorial-1

1 D/f b/w languages
Website of Indian Railway

Lecture-1.

Introduction

JAVA

* programming lang.
* 1991 (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) eg:- Media player
* Web Applications eg:- Indian Railway
* Enterprise Application eg:- 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 end 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 ME are used to creating mobile app.

Imp What is JAVA?

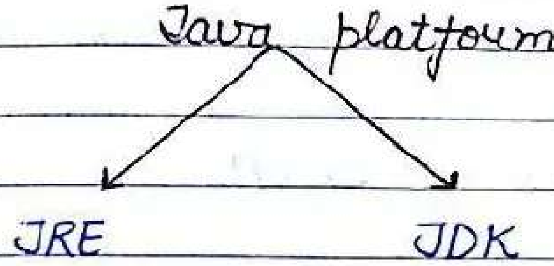
JAVA is a general object oriented programming language & a bcoz of JRE computing platform developed by "James Gosling" of Sun micro system in 1995.

Why Java?

Java is Secure

It is platform independent

Java is portable



Components

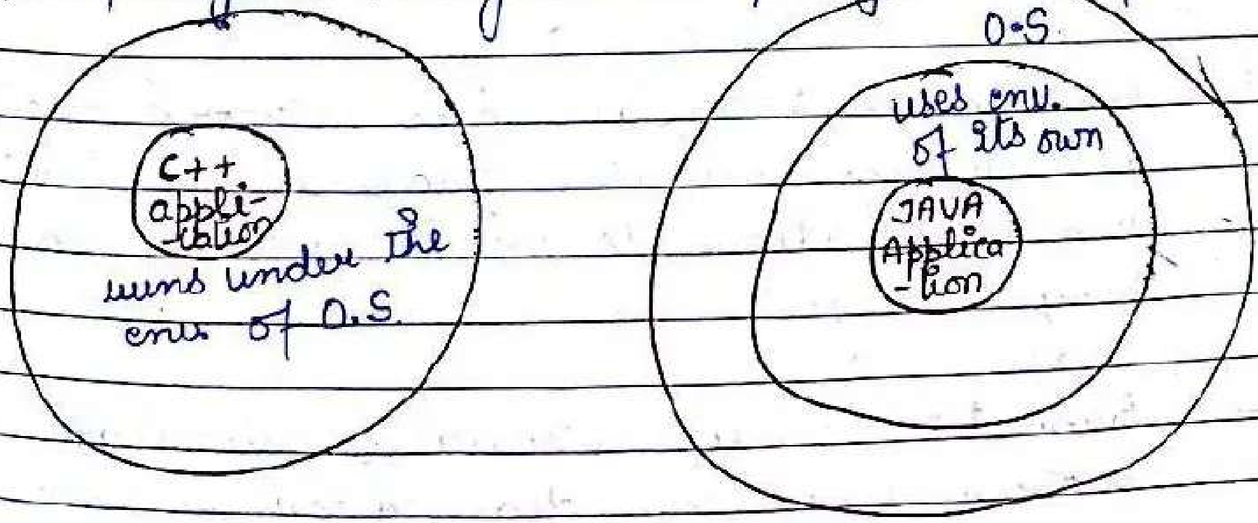
Java Run time environment

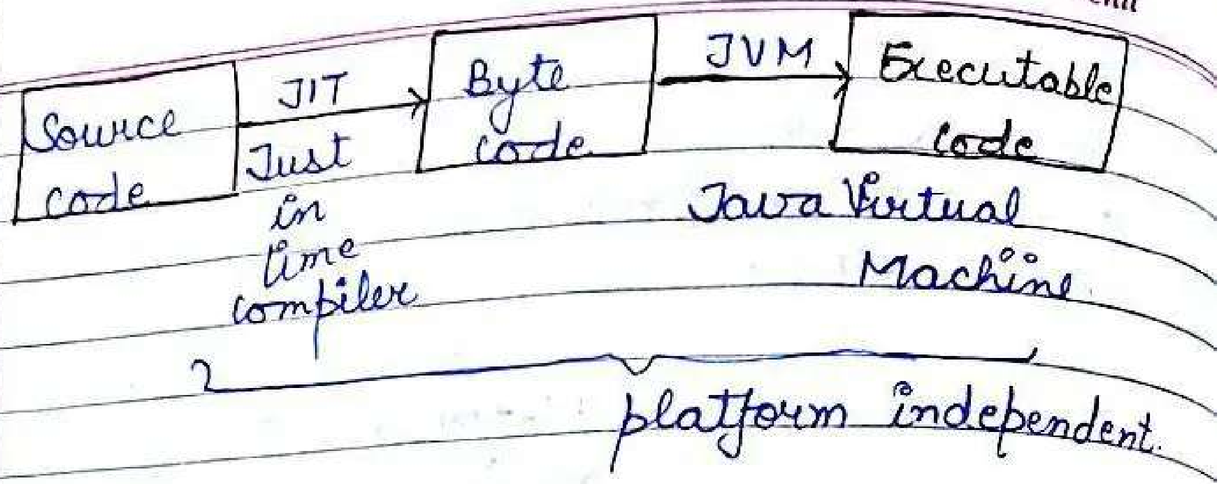
Java Development kit (JRE + Development tools)

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 & S/W 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 error are managed by exception

handling. Therefore, java is robust for program failures i.e. memory mgmt. mistakes & mishandled exceptional conditions.

5. Imp Platform Independent & Portable :→ Most significant contribution of java over other lang. is its portability. JAVA program can be easily moved from one computer to another anywhere anytime.

This is the reason why Java has become a very popular lang. for programming on internet which interconnects d/f kinds of system worldwide.

6. Secure:→ Since Java is used on internet. Security is an imp issue. Absence of pointers ensures that programs cannot gain access to memory locations.

7. Compile & Interpreted :→ Generally comp. lang. are either compiled or interpreted but JAVA combines both compiler & Interpreter.

8. Multithreading :→ JAVA was design to meet the real world environments of creating interactive, network programs to accomplish this. JAVA supports multithreaded programming which allows u to write programs o that do so many things simultaneously.

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

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

8. The inheritence allows sub class to inherit all the variables & methods of their parent class.

Inheritence may take d/f forms

1) Single inheritence (only one Super class)
2) Multilevel inheritence (derived from derived class)
Not in Java 3) Multiple inheritence (several Super classes)
4) Hierarchial (one super class & many sub classes)

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

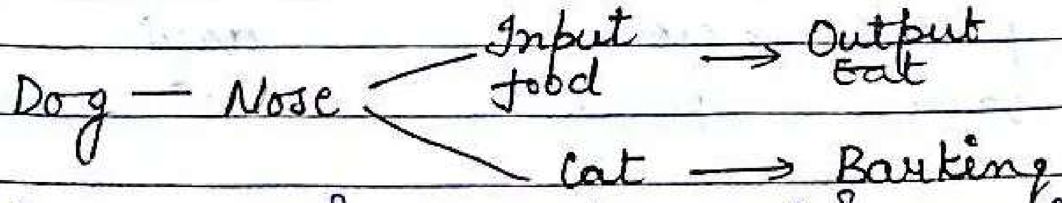
5/8

9. Polymorphism:→

many forms/behaviour

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

Polymorphism Ex-



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

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

5. 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 would for any interferance.
Ex- Mobile phone having d/f features combine In one., class having Students combine in one become CSE

Imp
6. D/f 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 complecated steps u have perform to do a particular operation	2 It is making inaccessable certain details i.e. just hiding the data so that it is not exposed.
It's a philosphical 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 the 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.

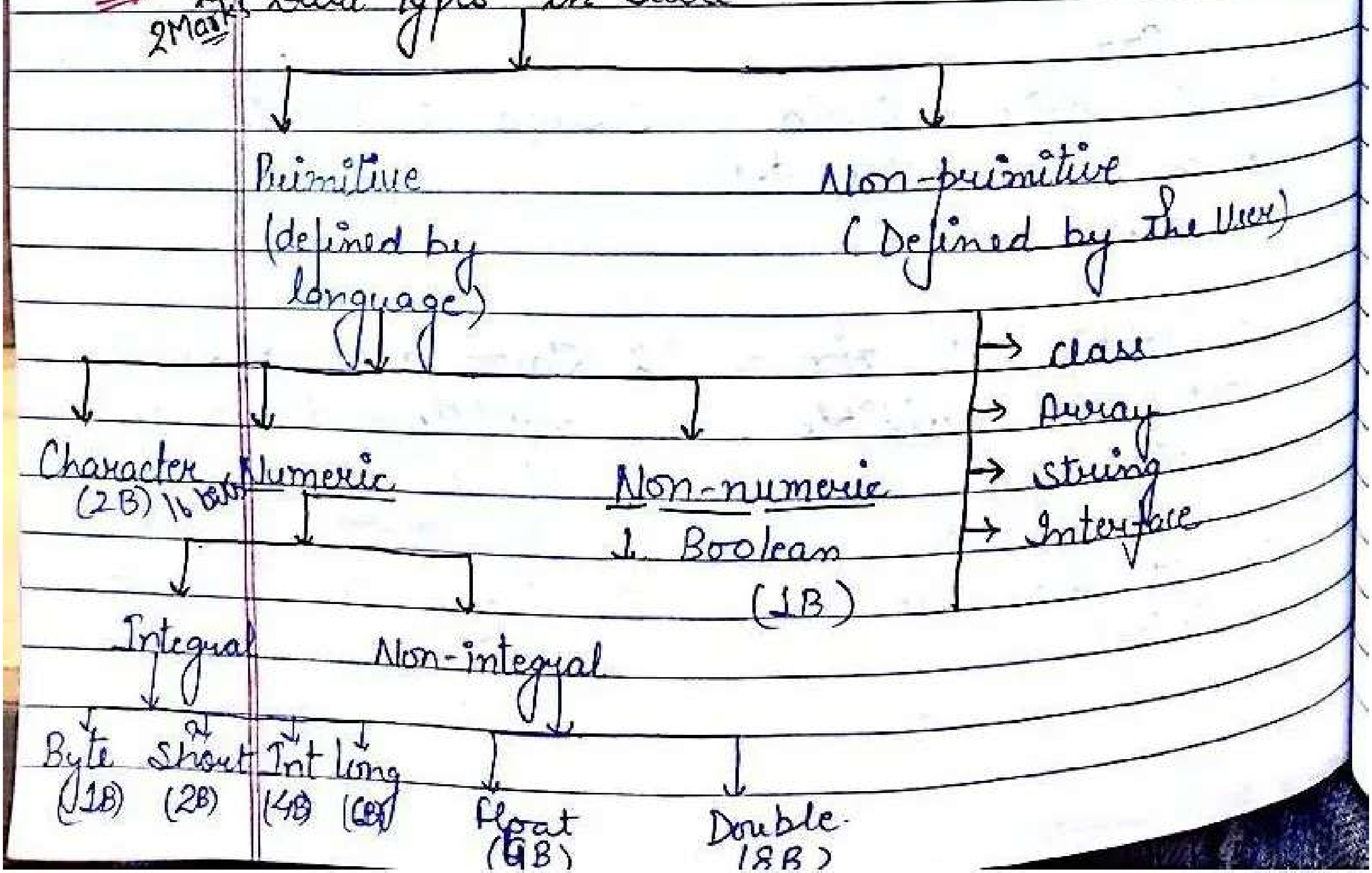
Imp

* D/f b/w C++ & JAVA

C++	JAVA
1. C++ is basically C wid extended Object Oriented extension.	1. Java is purely OOP lang.
2. It implements the concepts of multiple inheritence	2. Java does not support multiple inheritence of classes.
3. In C++ we use pointers.	3. There is no use of pointers.

C++	Java
4. In C++ we have destructor	4. Java replaced destructor fnc. wid finalize() method.
5. In C++ we use header files.	5. There is no use of header files in Java
6. There is operator Overloading in C++.	6. There is no Operator Overloading in Java
7. In C++ we use Global variable.	7. In Java there is no use of global variable
8. In C++ ~~we have~~ there is a concept of template classes.	8. It does not have template classes as in C++.

27/8 * Data Types in Java. 2Marks



(Short - Big)

* Type Conversion

* In some case it might want 2 assign value of one data type to variable of another type

* If both d source & destination types r compatible then JAVA performs d conversion.

* JAVA automatic Conversion

JAVA automatically converts one type to another only wen d following 2 conditions r satisfied.

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

Wen d ~~other~~ above two conditions r satisfied then Java performs "implicit conversion." It is also known as "Widening Conversion".

* Type Casting "Narrowing" (Big - Short)

If we want to convert two types which r incompatible size of destination type is less then the size of source type, then d 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 d 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
```

?: equivalent to if else statement

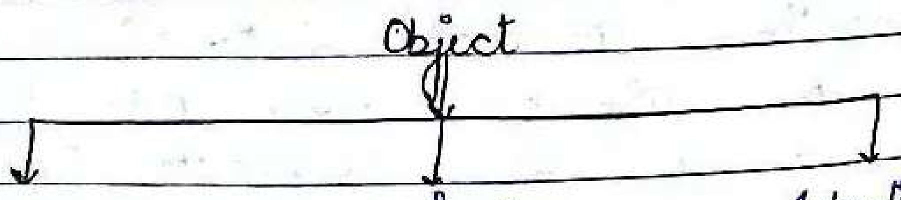
```
  15 10 if
 (a>b?: a : b
```

$$(a>b?\underbrace{(a>c?a}:c):(b>c?b:c)$$

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



*	State (Value)	Behaviour (funtionality)	Identity (Reference)
*	what object has	what object can perform	to identify the object
*	It is defined by the value that variable contains	It is defined by the fnc. of class	We 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 u 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 fnc. that will be known as local variables.

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

* Variables declare outside any fnc. wid 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[])   Command line argument
{
    int num1, num2; Double num3
    num1 = Integer. parseInt (arg[0]);
           wrapper class

        parsing of arg[0]
    num2 = Integer. parseInt(arg[1]);
    num3 = Double. parse ~~Int~~ Double (arg[2]);
    int sum = num1 + num2;
```

```
Compile    javaC Cse.java
Run        java Cse 5 10 10:56
           Sum is 15
```

5/9 Example Create an object of the class (File Name - Rectangle.java

```
class Rectangle
{
  int length, breadth;
  Rectangle ()
 {
   length = 10;
   breadth = 20;
 }
 void area ()
 { int area = length * breadth;
 }
}
class RectangleMain
{
psvm (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. length.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 fnc.

Ex

```
 class Area
{
 int length, breadth, int area;
                  10       20
 void area (int l, int b)
{
    length - l;
      breadth = b;
}
Class Area Main
{
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
fn Void area (int length, int breadth)
{
 this. length = length;
```

```
    this.breadth = breadth;
}
class AreaMain
{
psvm ()
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 object or 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 fnc. name but d/f parameters.

```
class Area
{
 int length, breadth, int area;
rectangle//void area (int l, int b)
 {
      length = 10;
      breadth = 20;
area - length * breadth
S.o pln ("Rectangle+" area);

}
                int s
void area (,) // square.
{
                 s          s
  area - length * length ;

}
```

Ex

```
class AreaMain
{
psum (   )
Area obj - new Area();
             10,20
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
{
psvm
 {
Employee obj1 = new Employee
101, "Goyal", "#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.

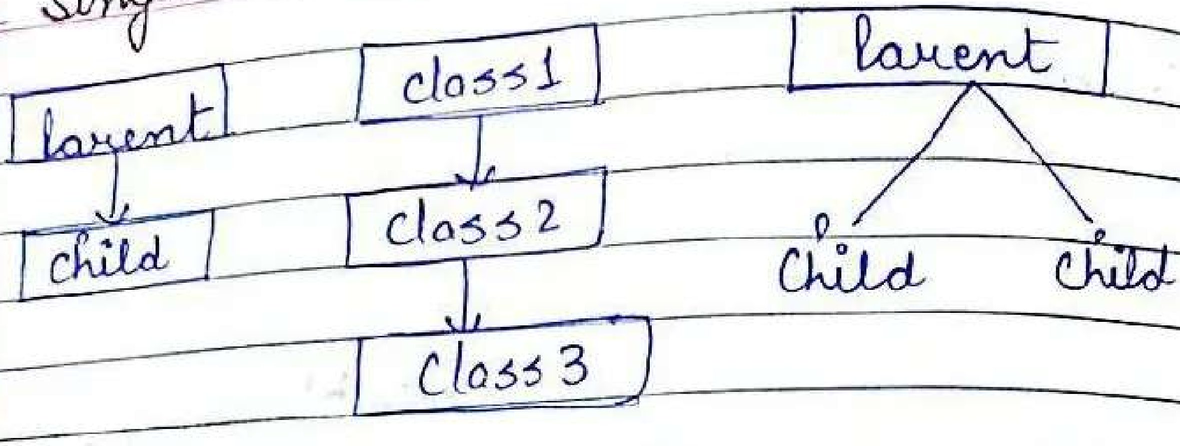
Q. fibonacci series.

* Inheritence

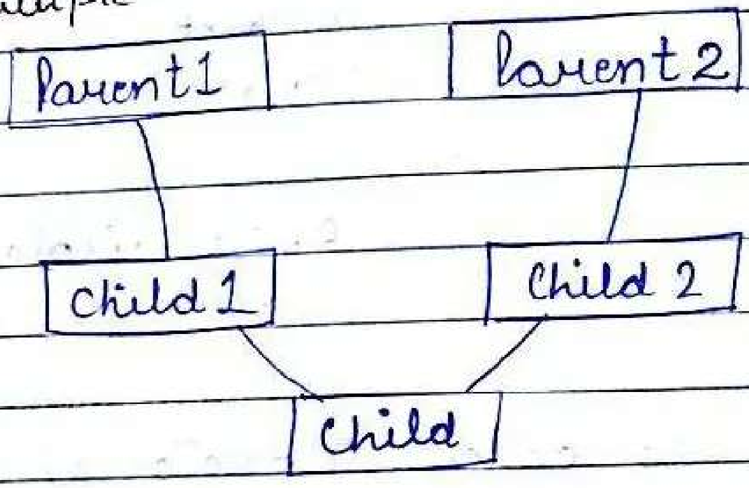
Ex
```
class Parent
 {
  int num1 = 10;

 }
 Class child extends Parent
  {
   int num 2
   num2 = num1 + 10;
 }
```

Single



Multiple


Parent 1
Parent 2
Child 1
Child 2
Child