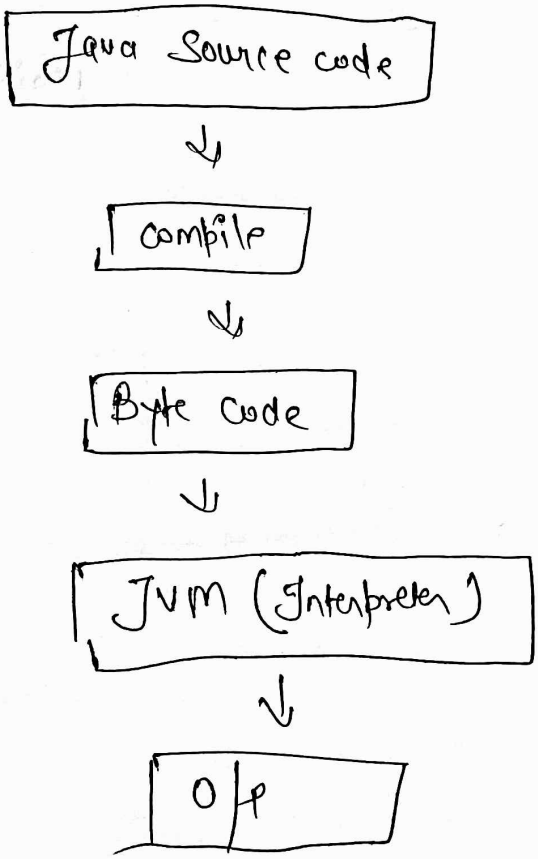
Java (James gosslay)

Sun microsystem lab

:: 1991

(1) Java is platform independent.

Platform independent means a single code can Run in any O.S.

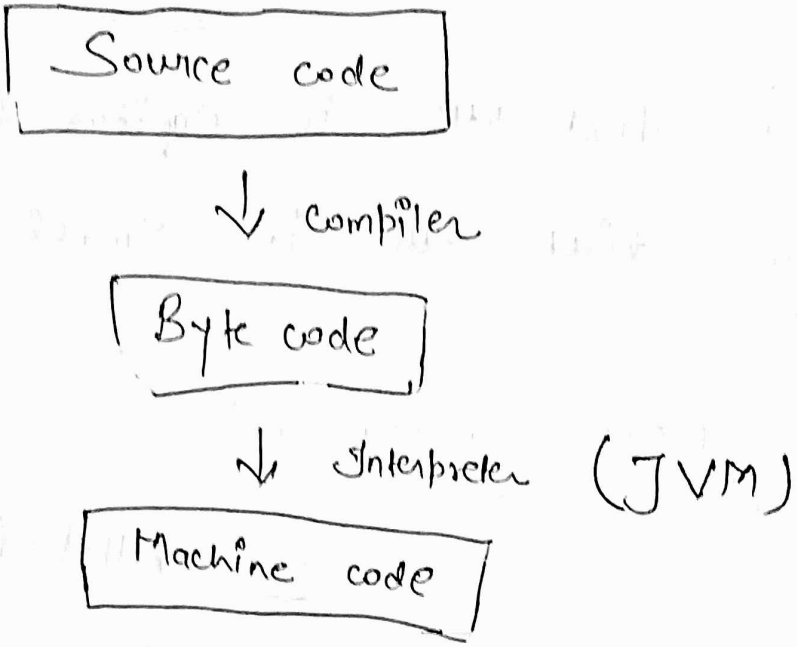


JVM is different for every O.S.

Java → Purely objective oriented lang. (OOPs)

How Java works ?

Java is both Compile and interpreter



* JVM :- Java virtual Machine.

JAVA Installation ?

Download JDK (Java development kit)

In 32 bit Computer architecture support before Java 8 and in 64 bit all latest version support.

∴ JRE :- JAVA Run time enviroment for
This is only for Run a program
In this you can not write, Modify.

Class :- first letter is capital (Pascal Naming)

function :- first letter is Small (Camel Naming)

Pascal Naming Conversation :-

Ex. AllAreGood, AddTwoNum

→ first letter is Capital

Camel Naming Conversation :-

Ex. addTwoNumber, addtwonum

→ first letter is small

1st program :-

~~Package com.Co~~

Package com.company; → optional

Public class Main { → class Name always in capital letter

public static void main (String[] args)

↓ Return Nothing ↘ function (always start with Small letter)

{

// write your code. → Comments

System.out.println ("Hello world");

↓ ↳ Change line (ln)

Anything Print Statement

}

}

★ Package → Same type of classes contain in Package. There are two type of Package :-

(i) built in package

(ii) User defined package.

Class :- Same type of function contain.

★ Package
↓
Class Contain
↓
function Contain
↓
Same type of operation contain

Data type :-

Data type are of two type :-

(i) primitive data type
(ii) Non primitive data type.

(i) Primitive data type :- like float, int, byte, long, Char, bool, Short.

int → integer (1, 2, 3)
Char → Characters (A, B, C)
float → 0.33 decimal No.
byte → No. Contain (-127 to 128)
bool → True, false
Short → Integer store.
double → large float No.
long → large No.

☆ Syntax :- Set of Rules is called Syntax.

Variable :-

A Variable is Contain that Store a value. This value Can be changed during execution of program.

```
int number = 8 ;
```

→ Number Stored.

↙ Data type ↓ Variable Name

Rules for declaring a Variable :-

(i) Must not begin with digit

Ex. 1int ✗ , 23Number ✗.

(ii) Case Sensetive like ayush, Ayush treated as different in Variable bcz Java is Case Sensetive.

(iii) Space is not used in Variable Name

Ex. int Number with =2 ✗

↳ Space ✗

(iv) Number is in mid like int, Num1 = 8 ✓

, Nam2Num = 8 ✓

System.out.print("Hello"); ✓

But in this ~~Next line~~ all statement in single line

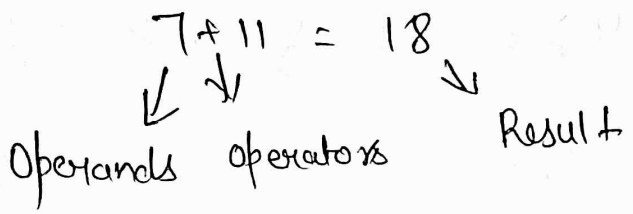
System.out.println("Hello");

in this Next line jump cursore.

*** ∴ In is used to jump the cursor in New line

Operators & Expressions :-

Operators are used to perform operations on Variables and Values



Type of operators :-

(i) Arithematic operator → +, -, *, /, %, ++, --, ---- (% → Modulous / Reminder.)

(ii) Assignment operator → =, +=

(iii) Comparision operator → ==, >=, <= {Return true, false}

(iv) logical operator → &&, ||, !

(v) Bitwise operator → &, |

logical operator :-

(And) && uses Both answer is true than answer is true

Statement (64 > 8) && (64 > 7)

True

(64 > 8) && (64 < 7)

false

|| (or) Any one is true, Answer will be true

Statement (64 > 8) || (64 < 9)

True

Statement (64 > 8) || (68 > 60)

True

Statement (64 < 8) || (64 < 9)

False

! (Not)

Statement :- 8 ! 9

True

Statement 8 ! 8

False

Precedence and Associativity :-

* No BodMarks Rule Apply in Java

only precedence and Associativity apply

Sign	Operators	Associativity	Precedence
() [] . →	function call Array Subscript Dot (Member of Structure) Arrow (Memember of Structure)	Left to Right	Highest 14
! ~ - ++ -- & (type) Size of	Logical Not one's complement Unarinary minus Increament decreament Address of Casting Size of	Right to Left	13
* / %.	Multiply Divide Modulus (Reminder)	Left to Right	12
+ -	Add Subtract	Left to Right	11
<< >>	Left Shift Right Shift	Left to Right	10

< <= > >=	Less than Less than equal to greater than greater than equal to	Left to Right	9
== !=	equal to (Comparision) Not equal to	Left to Right	8
&	Bitwise AND	Left to Right	7
^	Bitwise XOR	Left to Right	6
|	Bitwise OR	Left to Right	5
&&	Logical AND	Left to Right	4
||	logical OR	Left to Right	3
? :	Conditional	Right to left	2
=, +=	Assigment operators	Right to left	1
,	Comma	Left to Right	0 lowest

Left to Right Means ⟶

Right to left Means ⟵