DAY-1 JAVA

NASA

3 | Jun | 24

Defination :

Programming : is the process of creating a set of instruction that tell a computer, How to perform a task

Java : · is a classed based Object Oriented Programming language

→ Easy to Write, Compile and Debug code

→ Developed by James Gosling in 1995

features :

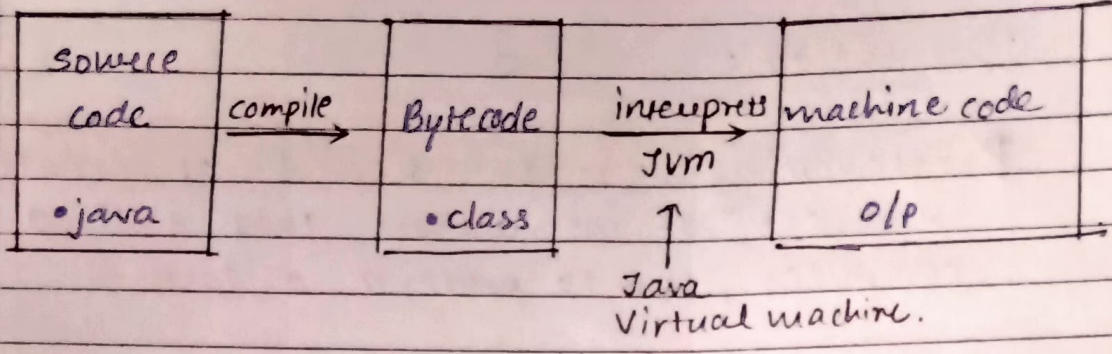
1. Object Oriented language
2. Platform Independent
3. Simple and Secure
4. Large Standard Library

Meaning Of Platform Independent :

→ That mean Java code can run on any Platform that has JVM

→ JVM is a program that interprets and executes Machine Code.

JAVA ARCHITECTURE :

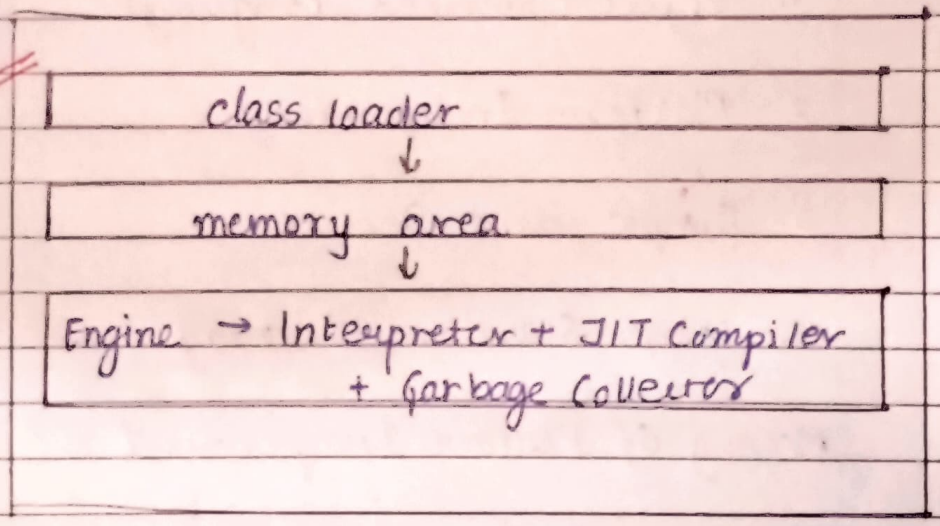


Compile ONCE → Run Every where

Components :

- JVM (Java Virtual Machine)
- JRE (Java Runtime Environment)
- JDK (Java Development kit)

JVM:



Class Loader : it loads bytecode into main memory

NASA

Memory Area :
There is an area where the class data is stored

Execution engine : It main task is to execute bytecode and execute the Java class.

Interepreter : It converts one language to another language
→ works as translator

JIT Compiler :
→ optimize performance of JVM
→ remove drawbacks of Interpreter

Garbage Collector
→ marks an area where garbage collector identifies used and unused chunks of memory.

JRE :

JRE = JVM + Libraries

→ JRE takes our Java code, integrates it with the required libraries and start JVM to execute it

JDK :

→ JDK holds JRE, Compiler, Intepreter, loader and development tools.

Difference :

JVM	JRE	JDK
Loader + Memory Area + Execution Engine	JVM + Runtime Libraries	JRE + Development tools.

Installation of Java in VScode :

1. Install JDK
2. Install Vscode if not installed
   → Install Extension related to Java

3. To code Java project
   → press Ctrl + Shift + P
   → type Java
   → select create java project
   → select No tool
   → And Name the project : my_project

4. To run Java project
   → type javac my_project.java (to compile)
   → type java my_project (to execute)

Note: Scanner problem : we use n.close(); at the end.

Java Syntax :

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

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

Java Outputs :

Syntax :

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

always capital

```
System.out.print ( );
```

println() is a method to get outputs.
ln is used to get output in newline.

```
System.out.println ( );
```

- System → class
- out → object of print stream
- println → method in print stream.

meaning of :

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

- public → access modifier (visible by everyone)
- static → allows to call this method
- void → return type
- main → entry point
- (String [] args) → command line argument, [] → list

Java Variables:

→ Variables are container ~~for~~ storing data

syntax:

```
Type variable_name = value;
```

ex:
```
int MoneyCount = 1000;
String name = manvi;
```

Java Naming Convention

Lowercase : moneycount

Uppercase : MONEYCOUNT

Camelcase : moneyCount

So; function, variables → camelcase
class → Pascal case
package → lowercase
constant → upper snake case
→ ex. MIN_CAPACITY.

Rules : Naming of Variable in Java.

1. Variable names should not begin with number
2. Whitespace is not permitted in variable names
3. A java keyword cannot be used as a variable name
4. Give meaningful names
5. if 1 word variable name should be in lowercase.

JAVA Identifiers.

→ All Java variable must be identifier with unique names.

Rules:

1. case sensitive (myVar or myvar are diff)
2. Start with letter or underscore.
3. Whitespace not allowed
4. No reversed words can used as names.

NASA

Java Datatypes:

Primitive Datatypes
→ built in java
→ hold single value
→ cannot be divided

primitive datatypes in detail.

boolean : Stores true or false values
size : 1 bit

byte : store whole number from -2^7 to 2^7-1
size: 1 byte

short : store number from -2^{15} to $2^{15}-1$
size : 2 byte

int : Store number from -2^{31} to $2^{31}-1$
size : 4 byte

long : store number from -2^{64} to $2^{64}-1$ (8 byte)

float : store decimal number from 6 to 7 decimal digits
size : 4 byte

double : store decimal number upto 15 digits
size : 8 byte

char : store character / letter
size : 2 byte

Non primitive:

→ user defined
→ holds memory address of variable

→ string, Array, Classes, Interfaces.

Java Inputs:

Java Scanner Class:
→ Scanner class is used to get user inputs and it is found in java.util.scanner package

Input Types:

nextBoolean() → read boolean value from user

nextByte() → read byte value from user

nextDouble() → read double value from user

nextFloat() → reads a float value from the user

nextInt() → reads a int value from user

nextLine() → string value from user

nextLong() → reads long from user

nextShort() → reads short from user

next().charAt() → char read.

NASA

Java Operators: → used to perform operations and variable and values.

- Arithmetic Operator
- Relational operator
- Logical operator
- Assignment operater
- Unary operator
- Bitwise Operater

Arithmetic :

+ , - , * , / , % , ++ , --

% : modulo → give remainder
++ : increment → post and pre
-- : decrement → post and pre

Relational :

< > >= <= == !=

ex x = 8 y = 10

x < y	x > y	x >= y	x <= y	x == y	x != y
✓	x	x	✓	x	✓

Logical :

&& || !

&& → return true if both are true
|| → return true if one of the statement are true
! → reverse the result

Assignment:

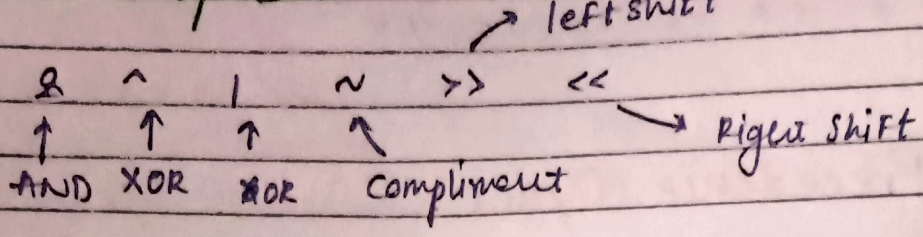
= += -= *= /= %=

Precedence Order:

1	postfix	++ --
2	unary	++ -- + - / ~
:	multiplication	* / %
	Addition	+ -
	shift	<< >> >>>
	relation	<<= > >=
	equality	== !=
	Bitwise AND	&
	XOR	^
	OR	|
	logical AND	&&
	condition	?:
	Assigment	= += -= . . .

increasing Order.

Bitwise Operator:



AND (&):

x	y	&
0	0	0
0	1	0
1	0	1
1	1	1

OR (|)

x	y	|
0	0	0
0	1	1
1	0	1
1	1	1

XOR (^)

x	y	^
0	0	0
0	1	1
1	0	1
1	1	0

compliment (~) 0 → 1 1 → 0

left shift (<<) → ex : ←0 0 1 0←

↓

0 1 0 0

$x << n$

$x * 2^n$

right shift (>>) → 0 0 1 0

↓

0 0 0 1

$x / 2^n$

DAY-2

4 | Jun | 24

Conditional Statement:

We use conditions to perform diff actions for diff decisions.

if statement:

→ block of code to be executed if a condition is true.

syntax:

```
if (condition){
    //statement;
}
```

if-else:

→ if block code to be executed if a condition is true

→ else block code to be executed if a condition is false

Syntax:

```
if (condition) {
    //statement
}
else {
    //statement
}
```

else-if:

→ used if there are more then 1 condition

Syntax:

```
if (    ) {
    //statement
} else-if (    ) {
    //statement
}
```

ternary operator

it consist of three operands.

Syntax:

```
variable = (condition) ? True : False;
```

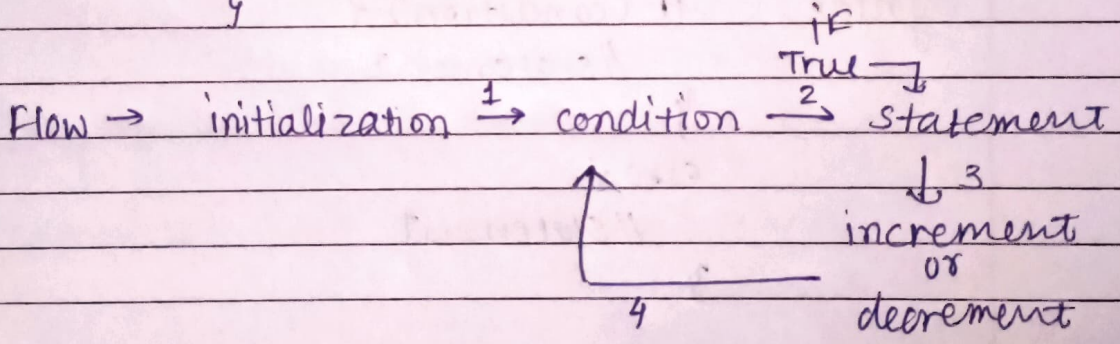
Loops

- Loops is used to iterate a part of program several times.
- for loop
- while loop
- do-while loop

For Loop:

syntax:

```
for (intialization; condition; inc/dec){
    //statement;
}
```



Label :

We use label before the for loop. Using while nested for loop as we can break/continue specific for loop.

Syntax :

```
aa:
  For (int i = 1; i<=3; i++){
    bb:
      for (int j=1; j<=3; j++){
        //statement ; break aa;
      }
  }
```

while :

Syntax :

```
while (condition){
  //statement
  //increment / decrement;
}
```

DO-While Loop :

Syntax :

```
do {

} while (condition);
```

Break :

→ It is used to break loop. It break the current flow of program at specified ~~contion~~ condition

Syntax :

```
break;
```

Switch Case: it executes one statement from used for multiple conditions

Syntax:

```
switch (condition) {
    case value1:
        // statement
        break;
    case value2:
        // statement
        break;
    default:
        // statement
}
```

Continue Statement:

→ used in loop control structure when you need to jump to next iteration of loop.

Syntax:

```
continue;
```