

"समय न लगाएँ इसमें कि क्या करना है,
वरना समय ये तय करेगा कि आपका क्या कराना है।"



Beyond the Basics

DSA Launch Pad with JAVA



Welcome & Vision

Why you started this course.

- Communication

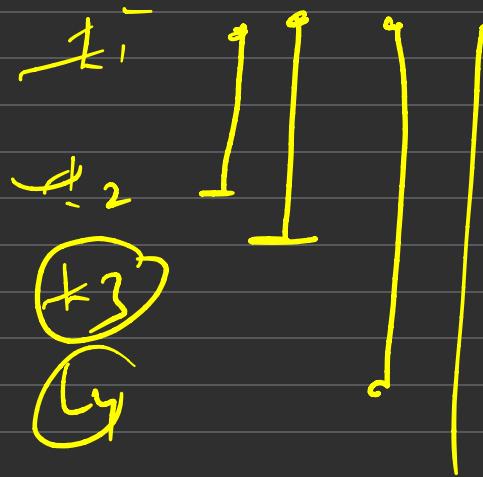
English

Comm.

"I don't just want you to learn coding.....

Don't Mug up - You are Engineer

Change Studying methods



Course Roadmap

Roadmap - Java - OOPs - Collection Frame - DSA

HomeWork Problems

Class Code

Notes

DSA

OOPs

DB

Project
Aph

✓✓✓

first sum = DSA / Aph Mp
project React JS

- -

Let's Start the JAVA 😊

What is Programming ?

Programming means giving instructions to a computer to do something for us.

What is JAVA Programming ?

- OOP → mainly used to build
Enterprise Application
- 1995 — James Gosling

Banking

90's

PROGRAMMING LANGUAGES AND THEIR USES

PYTHON

- 1) Data Science
- 2) Machine Learning
- 3) Web Development
- 4) Automation
- 5) Game Development
- 6) Data analysis
- 7) Data visualization
- 8) Artificial intelligence

JAVA

- 1) Android Apps
- 2) Server-Side Apps
- 3) Enterprise Apps
- 4) Web Based Apps
- 5) Big data
- 6) Game Development
- 7) Internet of things
- 8) Cloud computing

C++

- 1) Games Development
- 2) GUI Apps
- 3) OS
- 4) Database Systems
- 5) Embedded
- 6) Networking
- 7) Virtual Reality
- 8) Computer Vision

JAVASCRIPT

- 1) Server-side Dev
- 2) Web Dev and Apps
- 3) Mobile Apps
- 4) Machine Learning
- 5) IoT
- 6) Automation
- 7) Embedded system
- 8) Chatbot Development

SWIFT

- 1) IOS App Dev
- 2) Deep Learning
- 3) IOT
- 4) Server-side Dev
- 5) Open-source Dev
- 6) MacOS App Dev
- 7) Machine Learning
- 8) Automation

C#

- 1) Games Development
- 2) Web Dev and Apps
- 3) IOT
- 4) Backend Services
- 5) Windows App Dev
- 6) Robotics
- 7) Cloud computing
- 8) Database program

Apple → Swift

Windows → C#

Android → Java

Machine Learning → Python

Game Development → C++

Web Development → JavaScript

Server-Side Apps → Java

Enterprise Apps → Java

Cloud Computing → Java

Big Data → Java

Machine Learning → Python

Robotics → C#

Cloud Computing → C#

Database Systems → C#

Networking → C#

Virtual Reality → C#

Computer Vision → C#

Embedded → C#

OS → C++

GUI Apps → C++

Games Development → C++

Networking → C++

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

IOS App Dev → Swift

Machine Learning → Python

Automation → C#

Cloud Computing → C#

Database Programs → C#

Robotics → C#

Cloud Computing → C#

Networking → C#

Machine Learning → Python

Deep Learning → Python

Backend Services → C#

Robotics → C#

Cloud Computing → C#

Database Programs → C#

Automation → C#

Embedded Systems → C#

Mobile Apps → JavaScript

Server-Side Dev → Java

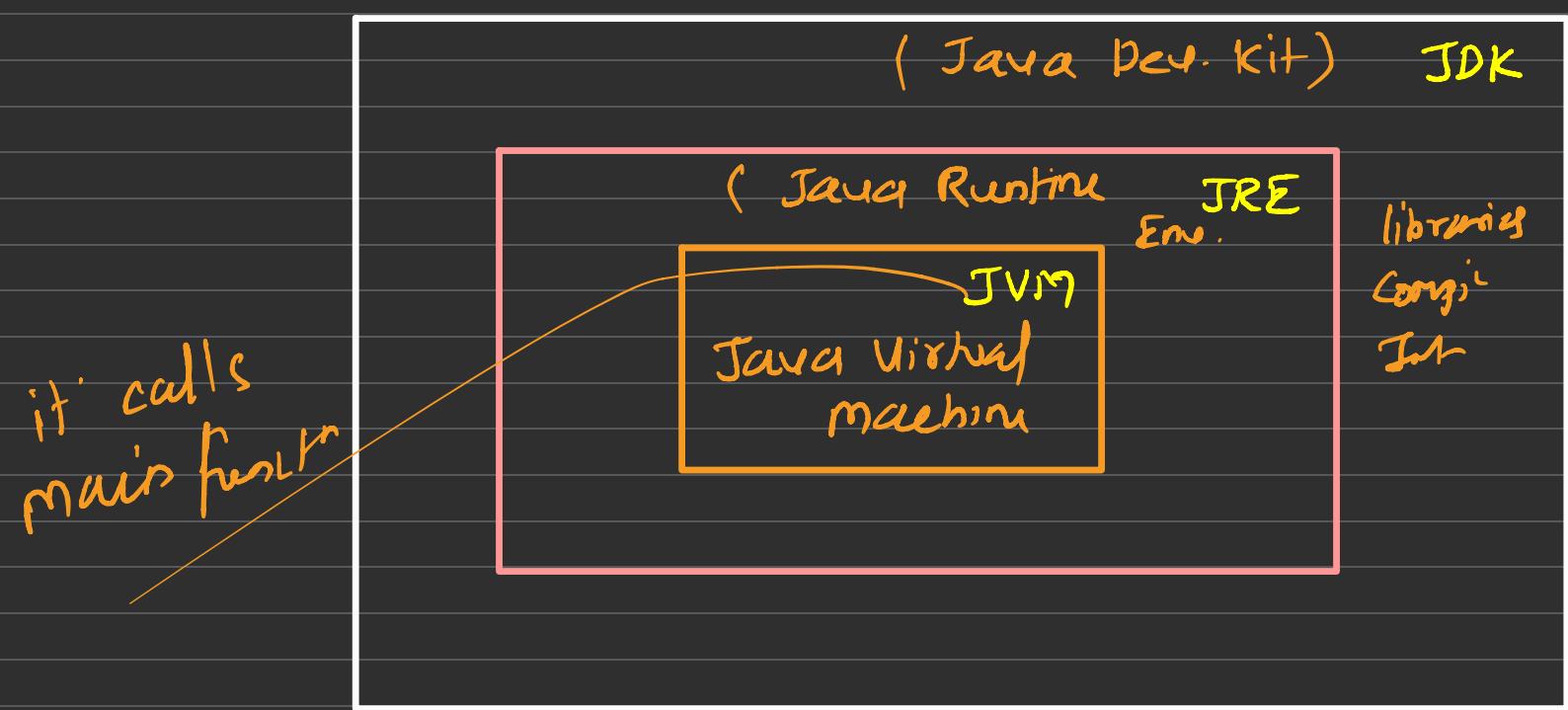
Deep Learning → Python

Open-Source Dev → C#

MacOS App Dev → Swift

JDK JRE JVM

Kitchen + Chef + Recipe

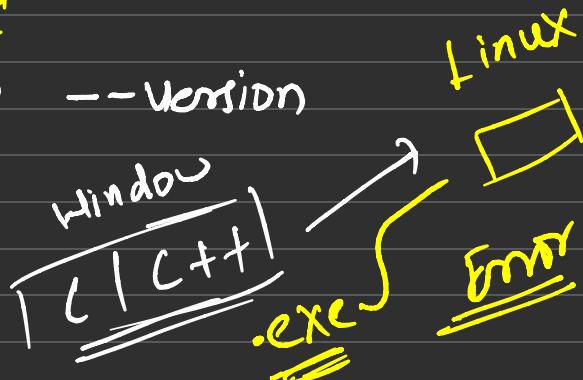


Setting Up Java Environment

Window = Wind + { R = cmd

java --version

UVIM



Features of Java Programming

* OOP * Platform Independent * Secure

* Portable — Window — Linux — Mac

AB

int

4 B (32 bit)
2 B (16 bit)

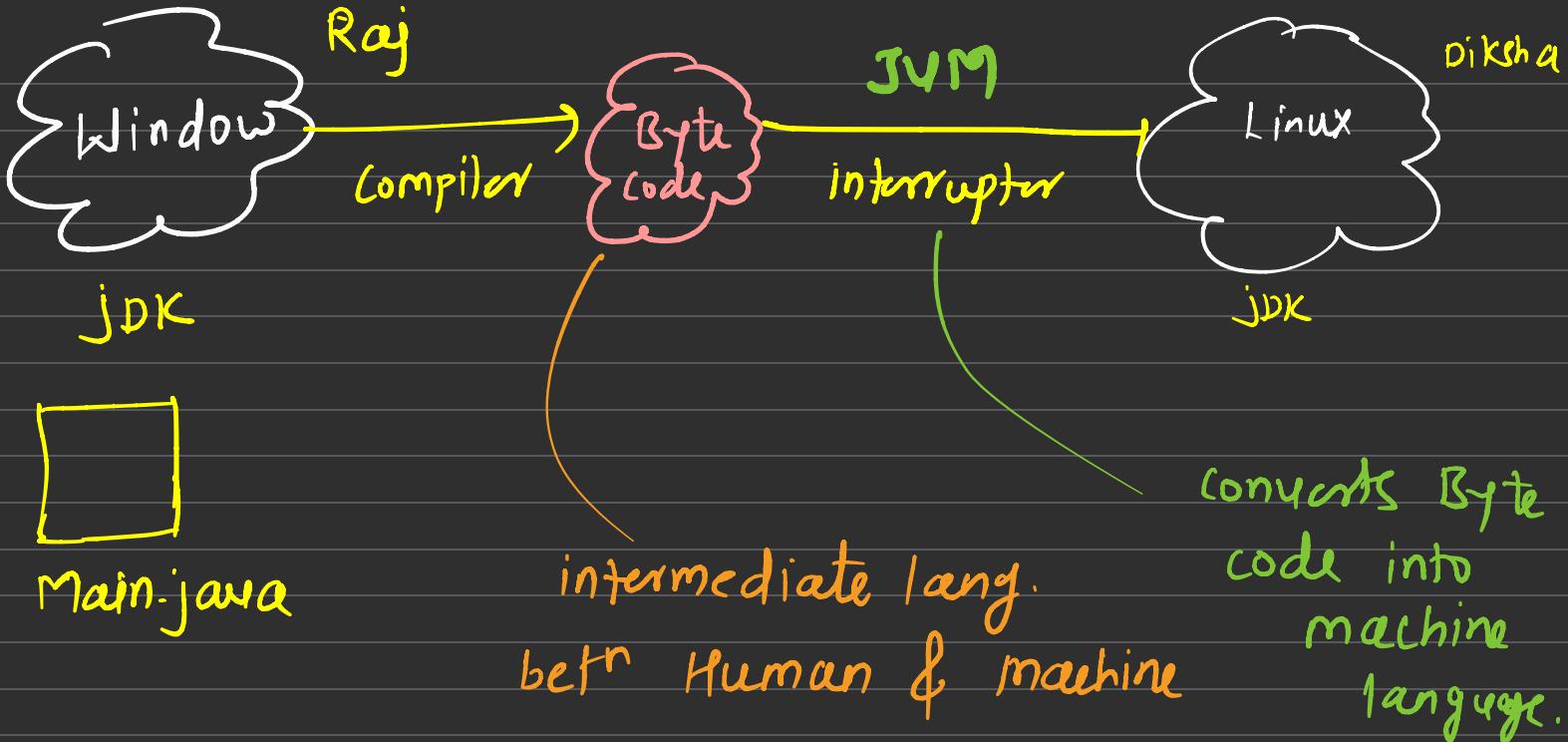
Structure of JAVA Programming

OOP (class & object)

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

Class Name and File Name should be same

class which contains main function → must
be public & class Name = fileName.java



public class Main
}

PSVM()
}
}
}

javac Main.java (compiles)

Main.class

Hello.class

class Hello

}

{
}
}

java Main (run)

add()
}

$x + y$
l

main()

add()

y

jym

Writing First “Hello World” Program

How to Print ?



What is System.out.println

System.out.println

Variables & Data Types

What is Variable

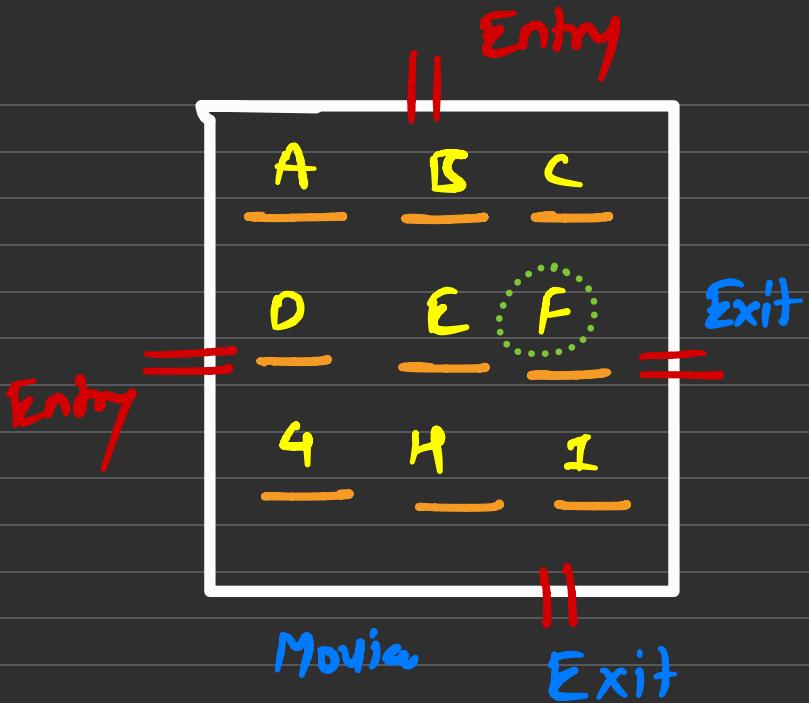
Pramod



Yash
○

Selling ticket

xyz

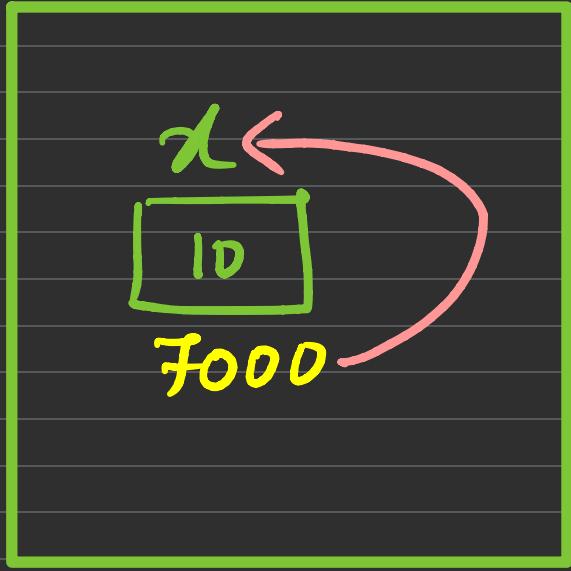


int $x = 10$

print (x)

Name given to memory

location



RAM

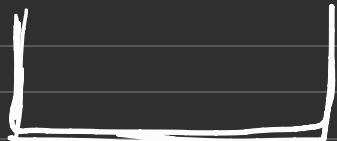
What is Data Types



water



Milk



chocolates

Bowl

thermas

- Mainly tells what type of data we store.
- Also tells us, what type of operations we can perform on them

Types of Data Types



Primitive

- ✓ - int
- ✓ - short
- ✓ - char
- ✓ - Byte
- ✓ - boolean
- ✓ - float
- ✓ - double

Derived

- Array
- String

User Defined

- class

(int) = mainly stores decimal values

size = 4B 1B = 8 bits

int x {
 Pos
 Neg}

4B = 32 bits

$$\frac{2^{32}}{2^1} = 2^{32} \times 2^{-1}$$
$$2^{31}$$

-2147483648

-2^{31}

2^{32}
 $+ 2^{31}$

2147483647

$$\text{Range} = -2^{n-1} - 2^{n-1} - 1$$

Data Type

(int/char) short

Byte long

(n = No. of bits)

- Short (int)

$$\text{size} = 2B \quad 1B = 8 \text{ bits} \quad = 16 \text{ bits}$$

$$\text{Range} = -2^{15} - 2^{-1} = -32768 \text{ to } 32767$$

- Byte (int)

$$\text{size} = 1B \quad 1B = \text{bits}$$

$$\text{Range} = -2^7 - 2^7 - 1 = -128 \text{ to } 127$$

long (int)

size = 8B 1B = 8 bits



- boolean

{ True
False

boolean x =

- char
 - char are shown / declared in single quote

Eg: 'x' '+' '/' '\z' '?'

size = 2B

(

unicode

C / C++ = ASCII values

(-128 to 127)

(0 to 255)

Size = 2B

1B = 8 bits

Range = 0 → 65535

(

(0 - 255) = ASCII

(256 - 65535) = Unicode

Java

float | double = floating points
(2.16 | 3.09)

Java

float y = 90.8F

double x = 90.6

float x = 90.90

(Cpp)

double y = 90.8

How to take input from the user

import java.util.Scanner

Scanner = mainly scans the input given by
User through Keyboard

Scanner sc = new Scanner(System.in)

(
Object

Scanner sc = new Scanner(System.in)

* int

int x = sc.nextInt()

* boolean

boolean x = sc.nextBoolean()

* char

char x = sc.next()

* double $x = sc.\text{nextDouble}()$

* $\text{nextLine}()$

(

Reads multiple words

with Space

(String)

* $\text{next}()$

(Single word)

(

String

Method

nextInt()

nextBoolean()

nextDouble()

nextFloat()

nextLine()

next()

work

int

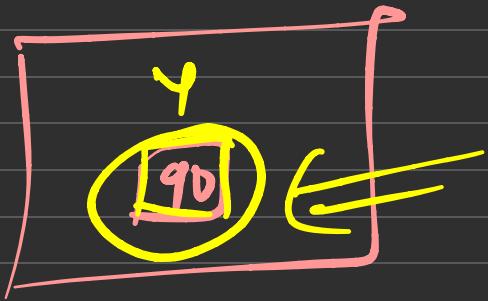
Boolean

Double

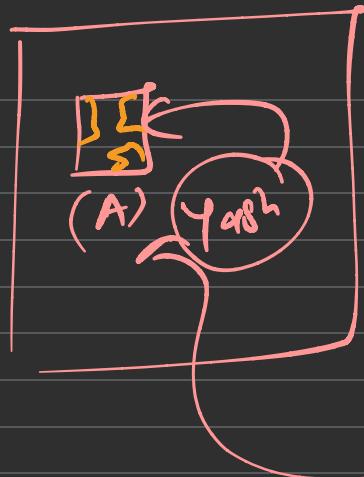
Float

String with multi word

single word



Ram



~~982~~
12

Yash
(1 - 4)

1 1 1

(30 min)

Garbag Collector



MM (Ram)

Operator Types

- Unary operator : Which req. single operand

$x++$, $++x$, $--x$, $x--$

- Binary operator : Which req. two operand

$x+y$, $x < y$, $x || y$, $x \& y$

- Ternary operator : Which req. three operands

$x ? y : z$

Operators

Arithmetic Operator

+ - * / %

(+) Add

(-) Sub

(*) Mul

(/) div

(%) Reminder

Priority

(/ * %) High

(- +) Low

$$3 / 4 + 6 - 3 * 2 + 9 \% 2$$

$$x = 4 \quad y = 7$$

$$ans = x * y$$

$$ans = \underline{28} \text{ (integer)}$$

Relational Operator

Mainly returns boolean {

True
false

< > ≤ ≥ == !=

Priority

(High) = < > ≤ ≥

(Low) = == !=

$$A = \underbrace{30 > 20 > 0}_{!} = 2 < 50 > 40 ! = 50$$

$$1 > 0 ! = 2 < \underbrace{50 > 40}_{!} = 50$$

$$1 > 0 ! = \underbrace{1 > 40}_{!} = 50$$

$$\underbrace{1 > 0}_{!} = 0 ! = 50$$

$$1 ! = 0 ! = 50$$

$$\underbrace{1 !}_{=} = 1 \Rightarrow 0$$

Logical Operator

Mainly returns either true or false

OR (||)

AND (&&)

NOT (!)

Priority

NOT (!)

AND (&&)

OR (||)

NOTE

?

* Any Non-Zero value is True {
 pos
 neg}

NOT

$\text{! } \text{f} \rightarrow \text{! } \text{True} \rightarrow \text{False}$

$\text{! } \text{o} \rightarrow \text{! } \text{False} \rightarrow \text{True}$

AND (&&)

(A && B = Y)

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR (||)

(A || B = Y)

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

AND

① $x \& y$

- if x is true then and then only y
is checked

- if x is false, then y is not checked

OR (||)

- if x is True, then y is not checked
- if x is false, then y is checked

Increment Decrement

* Increment

- pre-Increment = first increase & then use
 - post-Increment = First use & then increase
-

- pre-decrement = ~~= / \ =~~

- post-decrement = ~~= - / \ -~~

* pre

int $x = 10$

$y = ++x$

$z = x + y$

printf(x, y, z)

int $x = 10$

$x = x + 1$ $x = 11$

$y = x$ $y = 11$

$z = x + y$ $z = 22$

printf(x, y, z)

* post

int $x = 20$

$y = x++$

$z = x + y$

pf(x, y, z)

int $x = 20$

$y = x$ $y = 20$

$x = x + 1$ $x = 21$

$z = x + y$ $z = 41$

pf(x, y, z)

int $m = 10$

$n = ++m$

$n_1 = m++$

$n--$

$--n_1$

$n = n - m$

$pf(n)$

12 \ 3)

✓ int $m = 10$

✓ $m = m + 1$

✓ $n = m$

✓ $n_1 = m$

✓ $m = m + 1$

✓ $n = n - 1$

✓ $n_1 = n_1 - 1$

✓ n_1

$n = n - m$

m

$\cancel{H_{12}}$

n

$\cancel{H_{10}}$

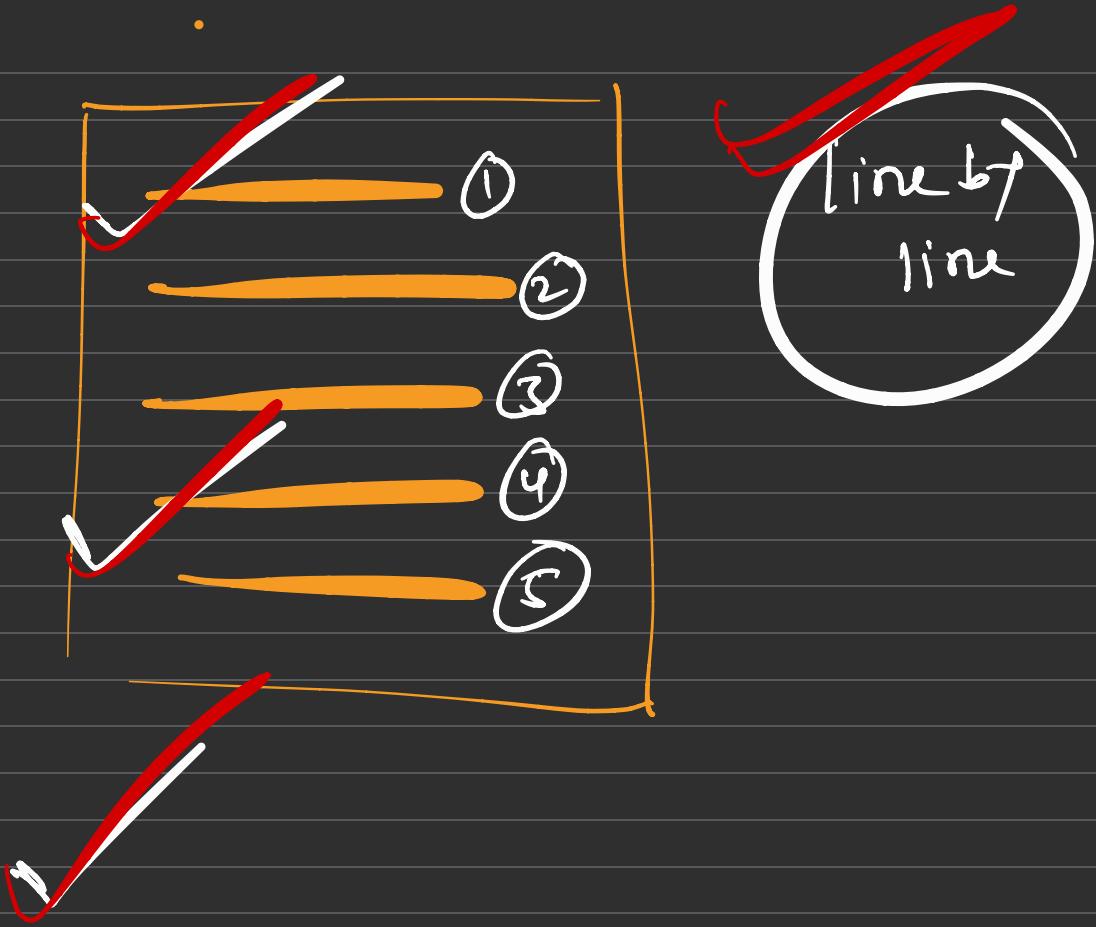
n_1

$\cancel{H_{10}}$

$10 - 12$

-2

Control Flow Operator



(Contro) flow



Selection

- if
- if else
- Switch

For loop

While

do while

Continu

break

if :

Syntax:

if (Condition)
{
 // statement

if (s)
{
 println(Hello)

if (Expression)
{
 // statements

Any Statement
which has Value

```
if (15 < 10)
```

```
    println ("Hello");
```

```
    println ("Bye");
```

```
    println (" Hi");
```

if no curly braces,
then only first semi-
colon statement is consid
(loop) if / else

Bye

Hi

if else

- if is true, then else

Syntax:

if (expression)
{

will not execute

- if is false, else will
execute

}

else
{

}

if ($a < 10$)
println (Hello);

else
println (Bye);

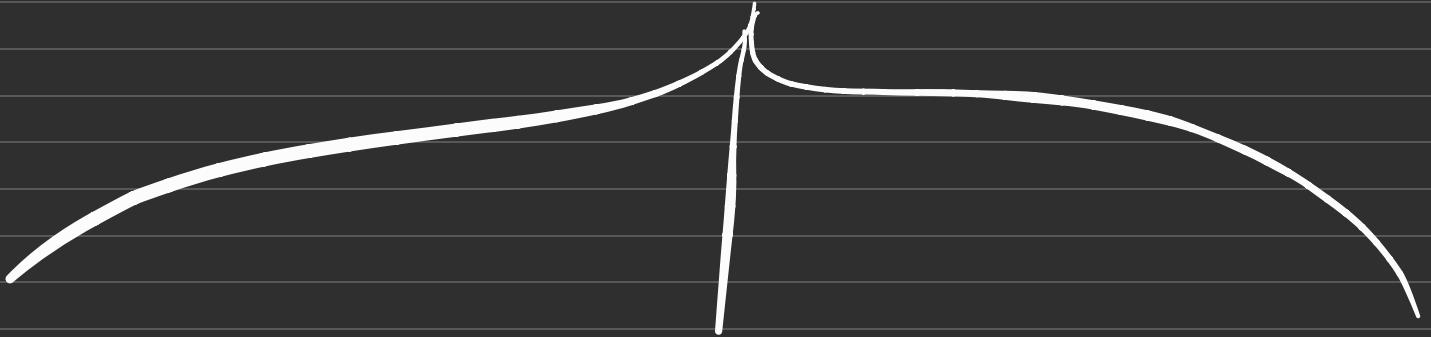
if ($a > 10$)
println (Hi);

else
println (Bye);



$x = 10, y = 20$
if ($x < y$)
println (Hi);
println (Bye);
else
println (Guru);

Loop: mainly used to do the work Repeatedly



FOR

WHILE

DO
WHILE

* for

Syntax:

for (init ; condition ; inc/dec)

Eg: for (1 ; 2 ; 3)
 println ("Hello")



for (Exp1 ; Exp2 ; Exp3)

1 Time

for (Exp1 ; Exp2 ; Exp3)

}

>

A hand-drawn diagram illustrating a single iteration of a for loop. At the top left, the word "Time" is underlined. To its right, a circled number "1" is connected by a curved arrow to a large circle representing the loop body. Below the "1", another curved arrow points from the "1" to the closing brace "}" of the loop. A yellow bracket underneath the "1" and the brace indicates the scope of the iteration. The entire loop structure is enclosed in a large oval.

```
for ( ; ; )  
    println("Hello")
```

```
for ( ; ; c : )  
    println("Hello")
```

```
for ( 2 ; ; 3 )  
    println("Hello")
```

By default Min-Zero
value

While

Compulsion

while (Exp | condition)

}

int $x = 12$

while ($x < 20$)

{ print | Hello)

inc) dec

{

first check condition

}

$x++$

& Then Execute

* Do While

int $x = 10$

do
{

do
{

println (Hello)

.

$a++$

} while (condition);

} while ($a > 100$);

It first execute & then check condition