About Me

Incoming SDE Intern + SDE (FTE) @ **Delhivery**

SDE Intern + SDE (FTE) offer @ **Chaayos**

Technical Content Writer/Engineer @ **Scaler**

Technical Content Writer/Engineer @ **PrepBytes**

Ex – Technical Content Writer @ **InterviewBit**

Ex – Technical Content Engineer @ **Pepcoding**

Offers from other Ed-tech institutions like Unstop (Dare2Compete), TuteDude, etc.

Syllabus

Time & Space Complexity

① Getting Started

② Arrays & Strings
- → Two Pointer + Sliding Window
- → Greedy Algorithms
- → ☆ Searching & Sorting

☆ ③ Recursion & Backtracking

④ Object Oriented Programming (Java)

Data Structures & Algorithms

⑤ Linked List

⑥ Stack & Queue

☆ ⑦ Binary Tree & Binary Search Tree

⑧ Hashmap & Heap

☆ ⑨ Graphs

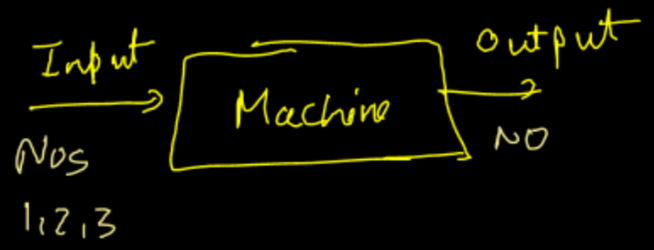
☆ ⑩ Dynamic Programming

⑪ Number Theory & Bit Manipulation

⑫ Trie

Computer → Human convinience

↳ Machine



Communication

Computer understands Binary lang (0 and 1s)

Humans understand Human Lang → instructions + emotions

Prog Lang

① Compiler → Software that converts Prog Lang to Binary Lang

Input/output

```java
// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

    }
}
```

comments

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        //this is a comment
    }
    //this is also a comment
}
```

Print in Java

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        System.out.print("DSA classes");
    }
}
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        System.out.print("Guneet Malhotra");
        System.out.print("DSA Classes");
    }
}
```

```
Finished in 69 ms
Guneet MalhotraDSA Classes
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        System.out.print("Guneet Malhotra ");
        System.out.print("DSA Classes");
    }
}
```

```
Finished in 82 ms
Guneet Malhotra DSA Classes
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        System.out.print("Guneet Malhotra");
        System.out.print(" DSA Classes");
    }
}
```

```
Finished in 84 ms
Guneet Malhotra DSA Classes
```

System.out.print("Rohan");

RI

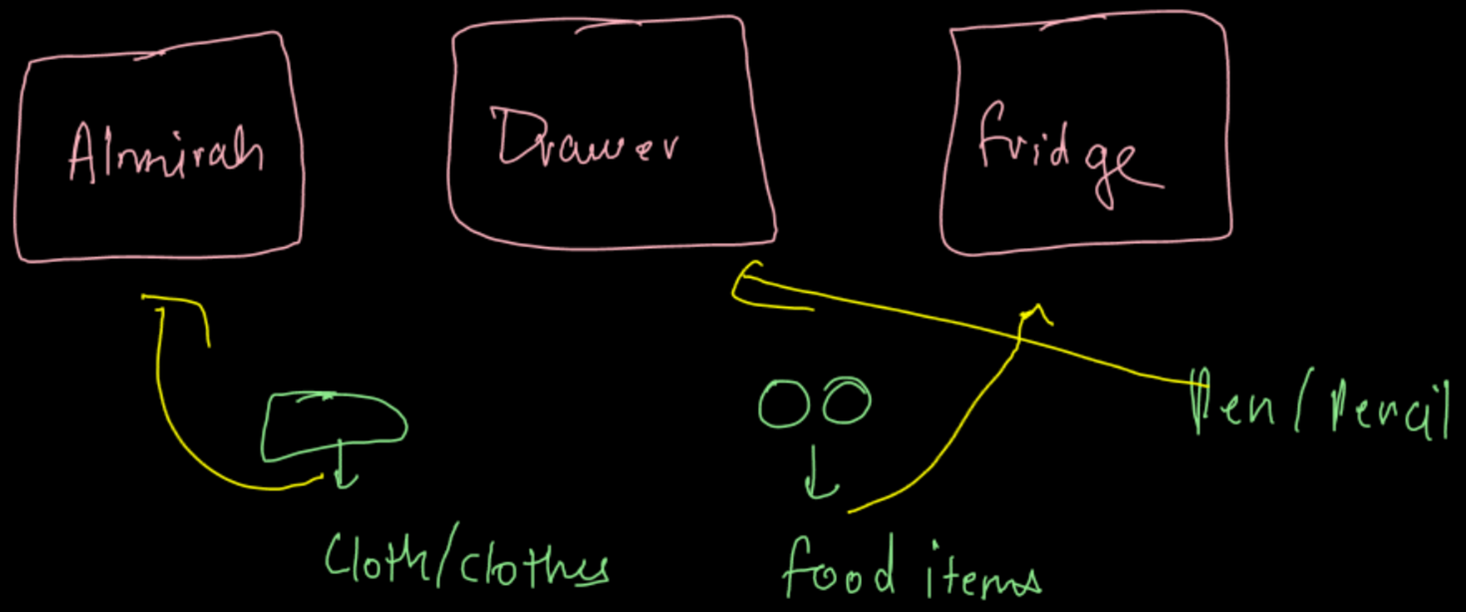
Rohan
I

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        System.out.println("Guneet Malhotra");
        System.out.print("DSA Classes");
    }
}
```

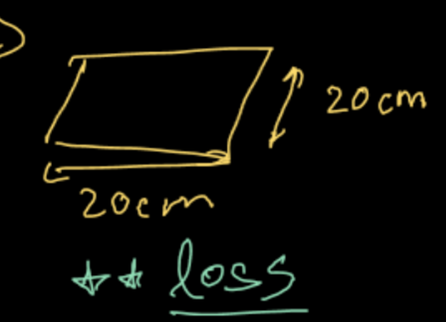
```
Finished in 122 ms
Guneet Malhotra
DSA Classes
```

Data types in Java

Type of containers.



If we try to put item in small size container, there is a loss



** loss

Size in bytes

Integral (w/o decimal)	byte	1	30, 50, 700 etc.
	Short	2	
	int	4	
	long	8	
floating pt type (with decimal)	float	4	3.141, 8.632 etc
	double	8	
Characters →	char	2 → A-Z, a-z, 0-9, #, *, etc.	
True/false →	boolean	Not fixed	

11/

Range of Datatype

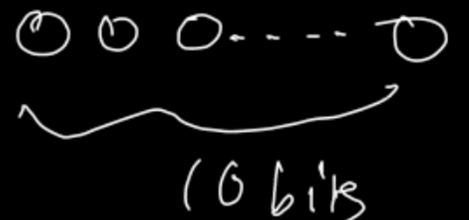
1 byte = 8 bits

$[-2^{N-1}$ to $2^{N-1}-1]$ → Here N is the no of bits.

Range of Short

Size = 2bytes = 2x8 = 16bits

$$\left[-2^{16-1} \text{ to } 2^{16-1} - 1 \right]$$



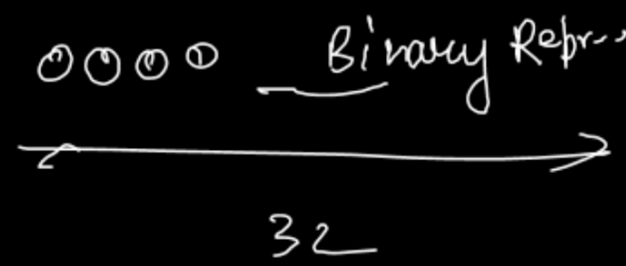
$$\left[-32768 \text{ to } 32767 \right]$$

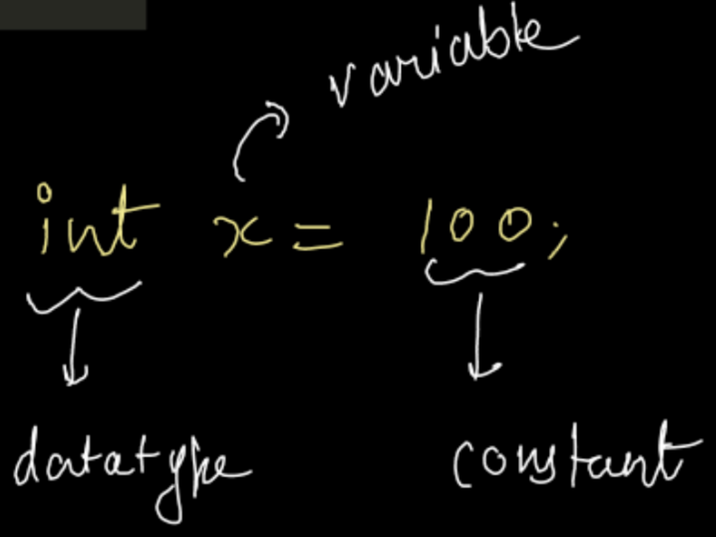
32768 → 17bit

Data loss

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 100;
        System.out.println(x);
    }
}
```

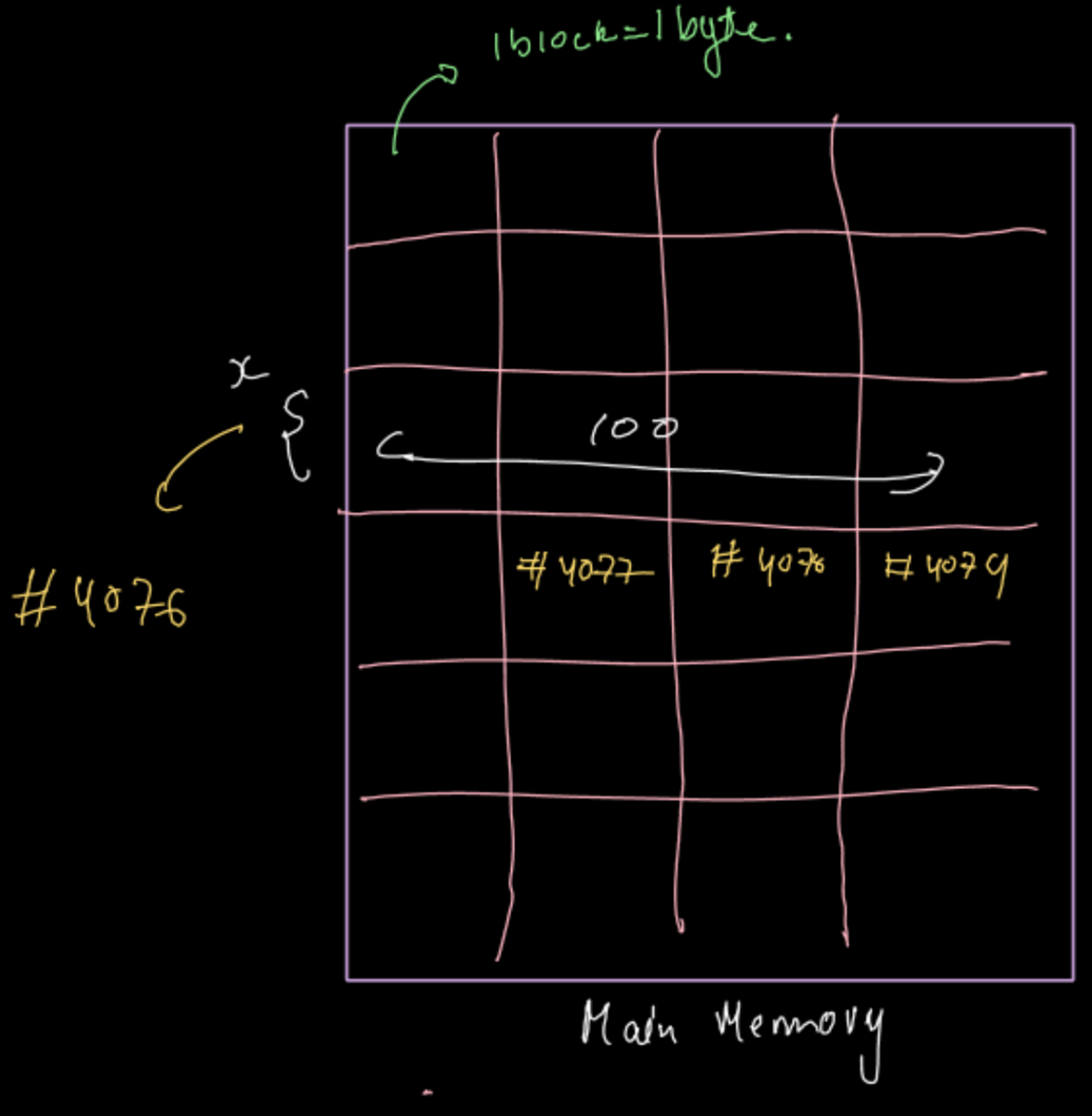
```
Finished in 62 ms
100
```





int x = 100;

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 100; (1)
        System.out.println(x);
    }
}
```



```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 100;
        System.out.println(x);

        char ch = 'a';
        System.out.println(ch);
    }
}
```

```
Finished in 64 ms
100
a
```

Operators in Java

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 100;
        System.out.println(x);

        char ch = 'a';
        System.out.println(ch);
    }
}
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 100;
        int y = x;
        System.out.println(x);
        System.out.println(y);
    }
}
```

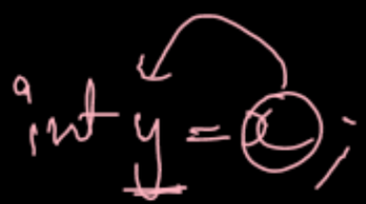
int x = 100;

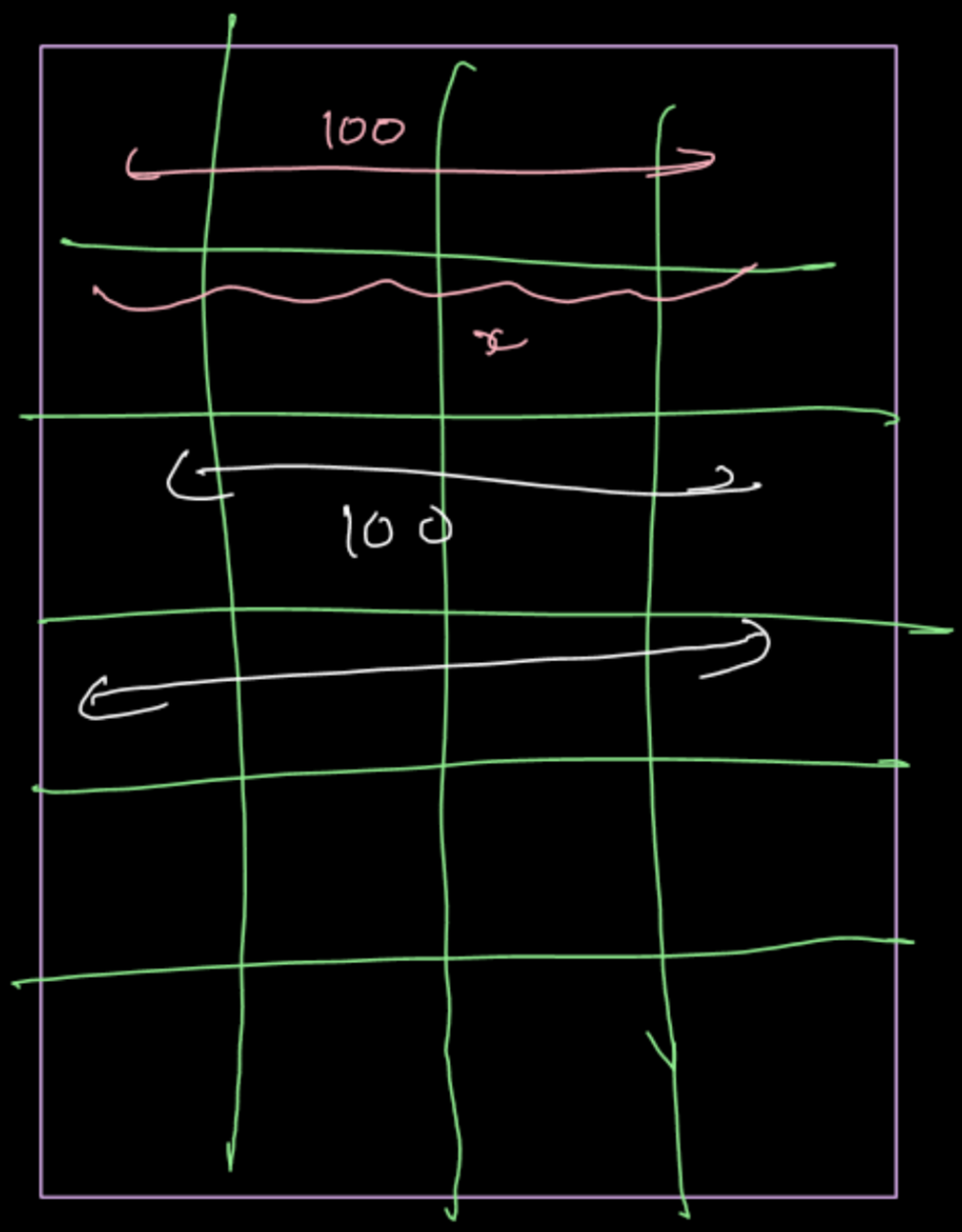
LHS ↓ RHS

assignment operator

RHS gets assigned to LHS

int y = x;


int y = 0;


100
x
100
y

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x;
        x = 100;
        System.out.println(x);
    }
}
```

$x = \cancel{GV}\ 100$

```
Finished in 50 ms
100
```

* Initialization of values is a must in Java.

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x;
        System.out.println(x);
    }
}
```

```
Finished in N/A
Line 5: error: variable x might not have been initialized [in Mai
n.java]
        System.out.println(x);
                           ^
```

Arithmetic Operators

\+ plus

\- minus

* multiply

/ divide

% modulus

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 10;
        int y = 30;
        int res = x / y;
        System.out.println(res);
    }
}
```

10/30 = 1/3 = 0.33 --

X

= 0

int x = 1.99

= 1

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 10;
        int y = 30;
        float res = (float)x / y; //typecasting
        System.out.println(res);
    }
}
```

→ process of converting 1 DT to another.

integer operator → decimal truncates

x is converted to float 10 → 10.00

float / int = float

10/30 →

```
      0
30 ) 10
     -0
     ---
     10
```

→ Quotient is the ans of integer /.

→ remainder is the ans of modulus.

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        int x = 10;
        int y = 30;
        int res = x % y;
        System.out.println(res);
    }
}
```

```
Finished in 70 ms
10
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int num = scn.nextInt();
        System.out.println(num);
    }
}
```

```
Finished in 108 ms
256
```

stdin

```
256
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int num = scn.nextInt();
        System.out.println(num);

        float f = scn.nextFloat();
        System.out.println(f);
    }
}
```

```
Finished in 164 ms
256
10.987
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int num = scn.nextInt();
        System.out.println(num);

    }
}
```

```
Finished in N/A
java.util.InputMismatchException
  at line 939, java.base/java.util.Scanner.throwFor
  at line 1594, java.base/java.util.Scanner.next
  at line 2258, java.base/java.util.Scanner.nextInt
  at line 2212, java.base/java.util.Scanner.nextInt
  at line 5, Main.main
```

stdin

```
10.987
```

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        char ch = scn.nextChar();
        System.out.println(ch);
    }
}
```

```
Finished in N/A
Line 5: error: cannot find symbol [in Main.java]
        char ch = scn.nextChar();
                      ^
  symbol:   method nextChar()
  location: variable scn of type Scanner
```

Next Char is not a method.

Getting Started (Lecture : 2)

→ Conditional statements
- if - else
- if - else if - else ladder
- switch case

} comparison operators + logical operators

→ functions in Java (with Memory concepts)

→ Loops
- for
- while
- do - while

} continue + break

Questions
- Even or odd 1
- Even or odd 2
- factorial

Conditional Statements

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        boolean b = scn.nextBoolean();
        System.out.println(b);
    }
}
```

When we want the code (part of it) to execute in a certain condition.

If - Else

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int guessNo = scn.nextInt();

        System.out.println("Start");

        if(guessNo == 10) {
            System.out.println("You have won the game");
        } else {
            System.out.println("You have lost the game");
        }

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

→ true

```
Finished in 153 ms
Start
You have won the game
end
stdin
10
```

guessNo == 10

LHS — comparison operator — RHS

if LHS = RHS

→ value of this statement is true

else it is false

```
if (condition) {
        ≡          → if condition is true then this executes
} else {
        ≡    } → if condition is false, then this executes
}
```

(condition) → true

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int guessNo = scn.nextInt();

        System.out.println("Start");

        if(guessNo >= 10 && guessNo <= 20) {
            System.out.println("You have won the game");
        } else {
            System.out.println("You have lost the game");
        }

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

Greater than or equal to

Less than or equal to

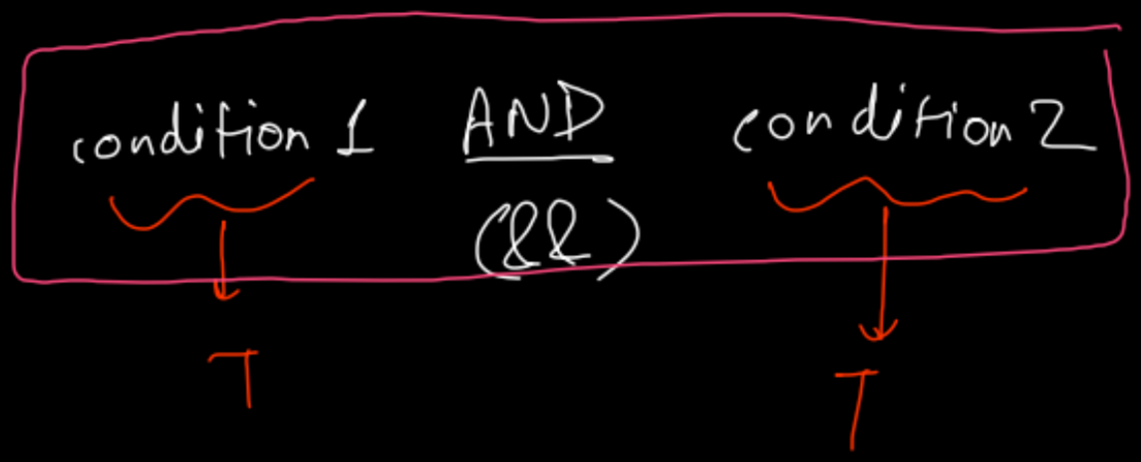
logical AND

HW

> → Greater than

< → Less than

AND



then AND value will be true

else value is false

Logical OR

```
1  import java.util.*;
2  public class Main {
3
4      public static void main(String[] args) {
5          Scanner scn = new Scanner(System.in);
6          int guessNo = scn.nextInt();
7
8          System.out.println("Start");
9
10         if(guessNo == 10 || guessNo == 20) {
11             System.out.println("You have won the game");
12         } else {
13             System.out.println("You have lost the game");
14         }
15
16         System.out.println("end");
17     }
18 }
```

logical OR

cond 1 OR cond 2

if min one is true

answer is true

Logical Not

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int guessNo = scn.nextInt();

        System.out.println("Start");

        if(!(guessNo == 10 || guessNo == 20)) {
            System.out.println("You have won the game");
        } else {
            System.out.println("You have lost the game");
        }

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

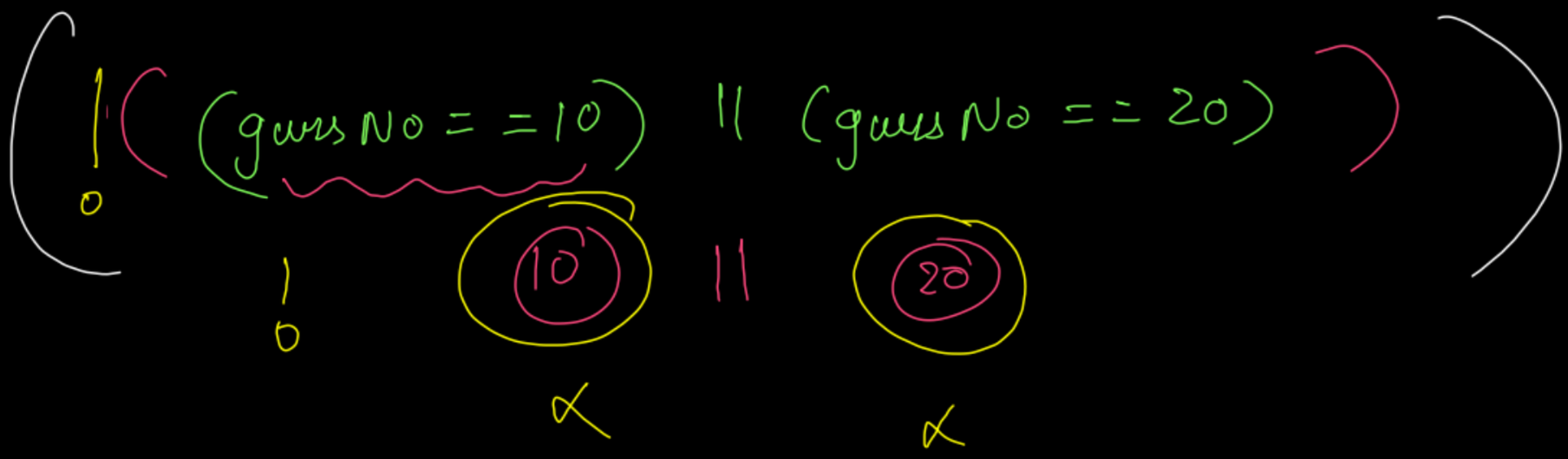
Logical not

! (condition)

↳ it negates the condition

True → false

false → True



Not Equal to

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int guessNo = scn.nextInt();

        System.out.println("Start");

        if(guessNo != 10) {
            System.out.println("You have won the game");
        } else {
            System.out.println("You have lost the game");
        }

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

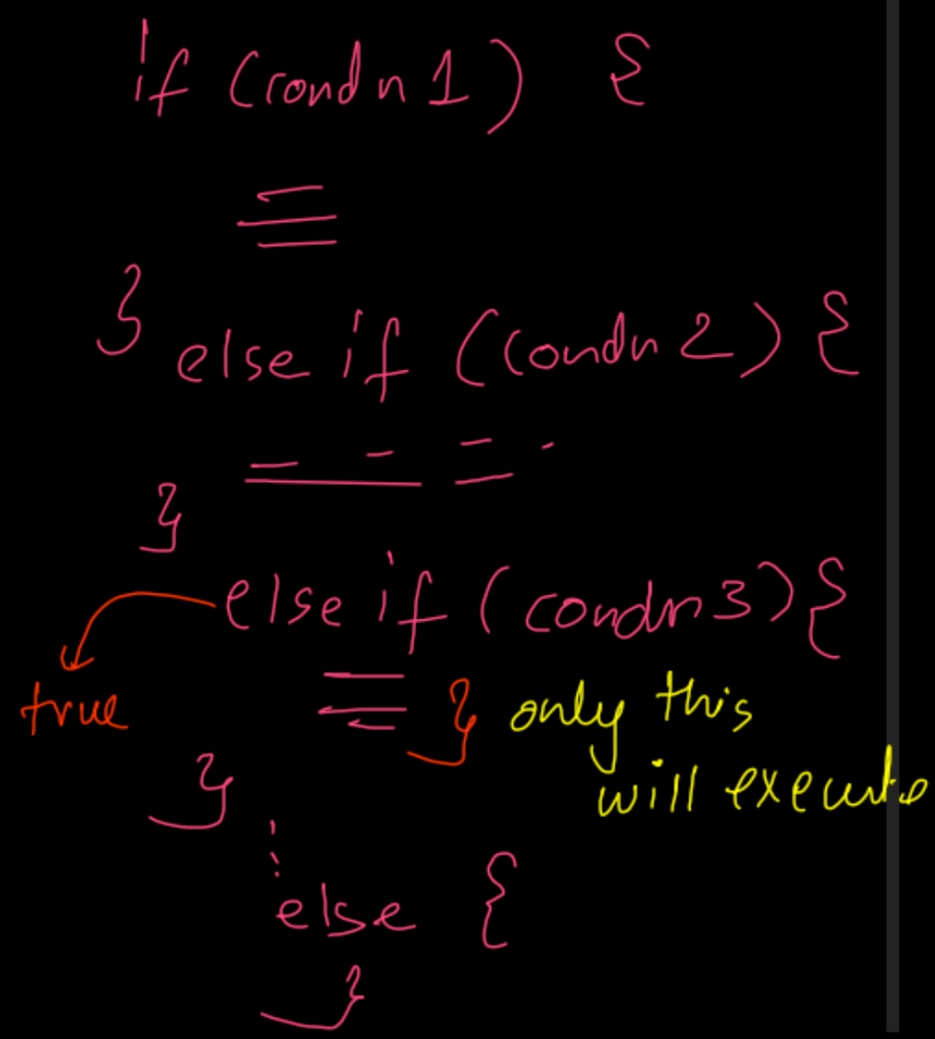
Not equal to

LHS is NOT equal to RHS

Then statement value is true else false.

If – else if – else ladder

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int guessNo = scn.nextInt();

        System.out.println("Start");

        if(guessNo == 10) {
            System.out.println("You have won the game");
        } else if(guessNo == 20) {
            System.out.println("You have won the game");
        } else {
            System.out.println("You have lost the game");
        }

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



Multible Else if

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int guessNo = scn.nextInt();

        System.out.println("Start");

        if(guessNo == 10) {
            System.out.println("You have won the game");
        } else if(guessNo == 20) {
            System.out.println("You have won the game");
        } else if(guessNo == 30) {
            System.out.println("You have won the game");
        }
        else {
            System.out.println("You have lost the game");
        }

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

```
if (condition) {
}
```

```
if ( )
{
}
```

```
if (condn1) {
}
else if (condn2) {
}
if (condition3) {
}
```

If there are multiple (if) statements one after another then they are checked.

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = scn.nextInt();
        int y = scn.nextInt();

        System.out.println("Start");

        if(x == 10) {
            System.out.println("You have won the game");
        } else if(x == 20) {
            System.out.println("You have lost the game");
        }
        if(x == 100) {
            System.out.println("You have won the game");
        }

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

```
Finished in 163 ms
Start
You have lost the game
end
stdin
20
100
```

for Loop

Print your name 10 times

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        for(int i=1;i<=10;i++) {
            System.out.println("DSA Classes");
        }
    }
}
```

```
Finished in 110 ms
DSA Classes
DSA Classes
DSA Classes
DSA Classes
DSA Classes
DSA Classes
DSA Classes
DSA Classes
DSA Classes
DSA Classes
```

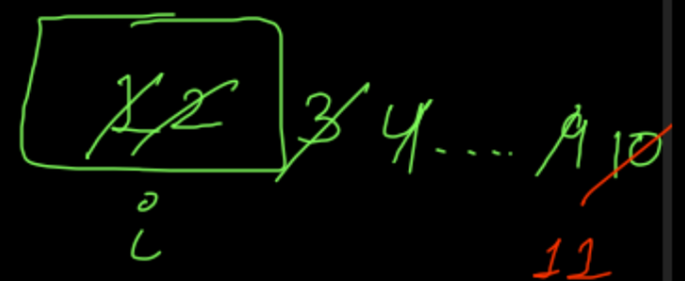
Generally used when exact no of iterations is known.

initial condition

condition

```
for (int i = 1; i <= 10; i++) {
    Syso("DSA");
}
```

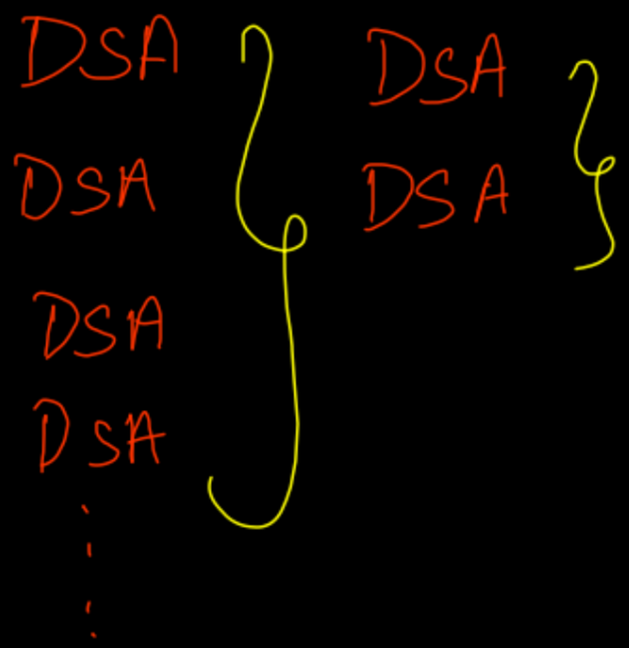
① ② ③



i++ → i = i + 1

(1) + 1

(2)



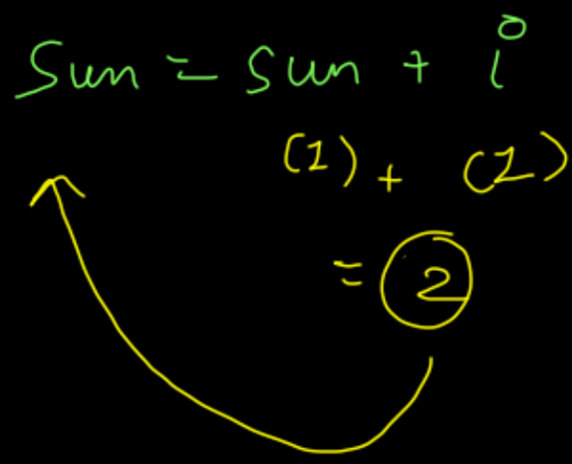
Counting 1 to N

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        int num = scn.nextInt();

        for(int i=1;i<=num;i++) {
            System.out.println(i);
        }
    }
}
```

Sum of 1 to N

num = 10

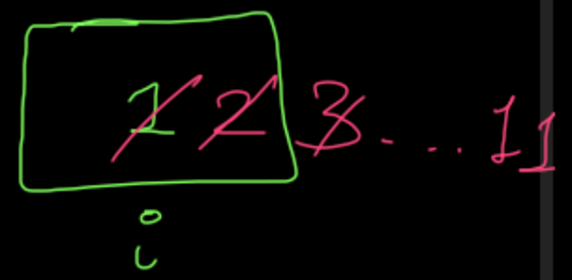
```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        int num = scn.nextInt();
        int sum = 0;
        for(int i=1;i<=num;i++) {
            sum = sum + i;
        }

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



```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        int num = scn.nextInt();
        int sum = 0;
        for(int i=1;i<=num;i++) {
            sum += i; //sum = sum + i
        }

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

While Loop (when we don't know exact no of iterations. But, we know the condition)

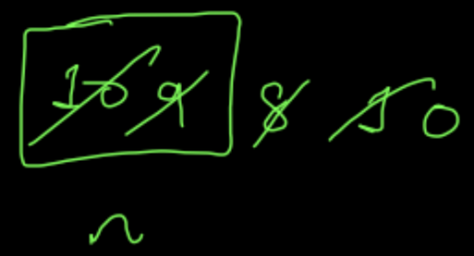
```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        int i = 1; → initialization
        while(i <= 10) { → condition
            System.out.println("DSA Classes"); } ②
            i++; ③
        }
    } →
}
```



DSA
DSA
DSA
.
.
.

Count N to 1

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        while(n >= 1) { (1)
            System.out.println(n);(2)
            n--;  (3)
        }
    }
}
```

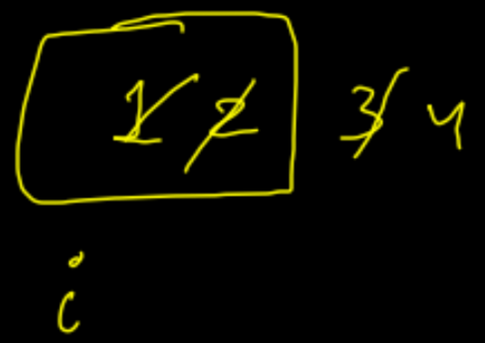


10
9
⋮
1

n-- → n = n - 1

Do while Loop

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        int i = 1; → initialization
        do{
            System.out.println("DSA"); → ①
            i++; → ②
        } while(i<=10); → ③
    }
}
```



```java
int i = 11;

while(i <= 10) {
    System.out.println("DSA");
    i++;
}
```

0

$11 \leq 10 \rightarrow$ false

0x

Loop not executed

```java
int i = 11;
do{
    System.out.println("DSA");
    i++;
} while(i<=10);
}
```

1

DSA

i = 12

$12 \leq 10$ ✗

Continue in Loops

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        for(int i=1;i<=10;i++) {
            if(i % 3 == 0) {
                continue;
            }
            System.out.println(i);
        }
    }
}
```

① ③ ② ↓ Skips the current iteration

```
Finished in 119 ms
1
2
4
5
7
8
10
```

1 2 ~~3~~ 4

1
2
4

i % 3 == 0

Break in loops

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        for(int i=1;i<=10;i++) {
            if(i % 3 == 0) {
                break;
            }
            System.out.println(i);
        }
    }
}
```

break from the loop
i.e exit the loop.

Switch Case

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int num = scn.nextInt();

        switch(num) {
            case 10:
                System.out.println("Option 10");
                break;
            case 20:
                System.out.println("Option 20");
                break;
            case 30:
                System.out.println("Option 30");
                break;
            default:
                System.out.println("Please select out of 10, 20 or 30");
        }

```

or character or String

Getting Started - Lecture 3

① Unary Operators

② functions and their Memory Mappings

DSA Sheet Questions

① Even or Odd - 1

② Even or odd - 2

③ factorial

④ Sum or Product

⑤ Sum of Even Nos Till N

Unary Operators

Operand 1 operator Operand 2

+, -, *, /, %, &&, || etc

operator Operand

++
--

++ → pre, post

-- → pre, post

Operand operator

+ (Unary Operator)

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = +3 + 5;
        System.out.println(x);

    }
}
```

Unary

Binary

- (Unary Operator)

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = -3 + 5;
        System.out.println(x);

    }
}
```

Unary

Binary

++ (Increment Operator)

Post

x++; ⟶ x = x + 1

ov + 1

Pre

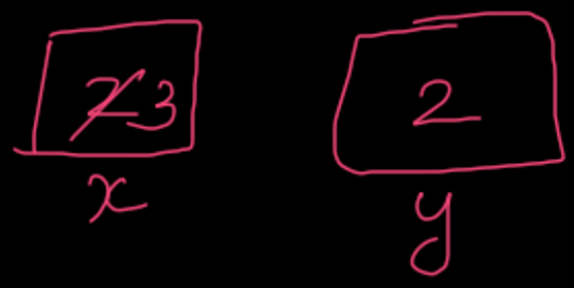
++x; ⟶ x = x + 1

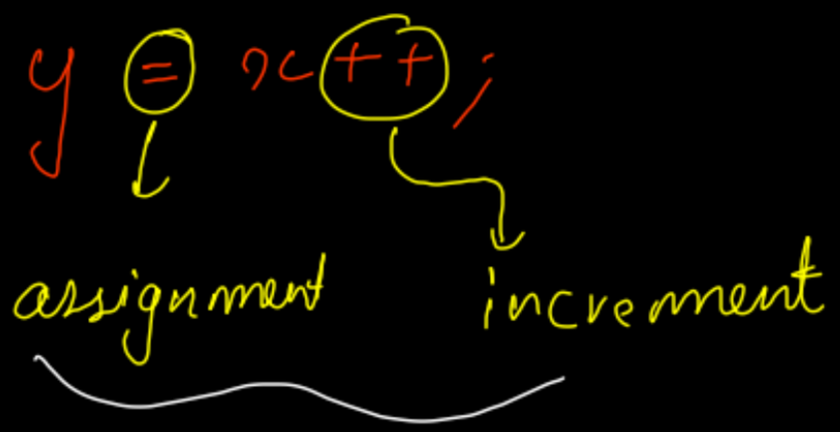
y = x++ ⟶ These ⟵ y = ++x

2 statements are different.

Post Increment

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = 2;
        int y = x++;
        System.out.println(x);
        System.out.println(y);
    }
}
```



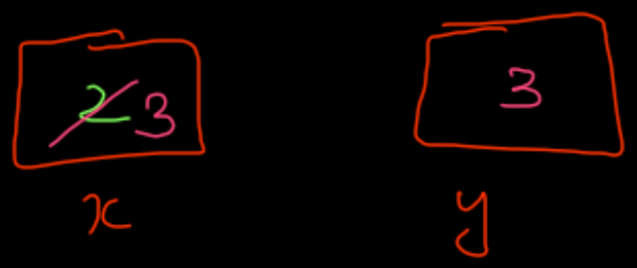


① $y = x$

② $x = x + 1$

Pre Increment

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = 2;
        int y = ++x;
        System.out.println(x);
        System.out.println(y);
    }
}
```



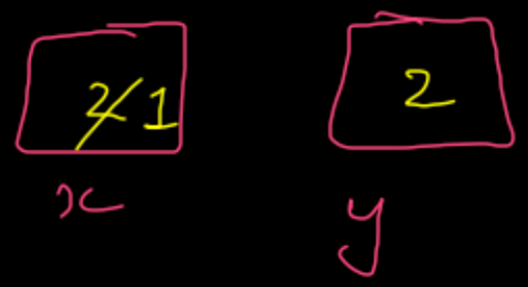
$y = ++x;$

① $x = x + 1$

② $y = x$

Post Decrement

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = 2;
        int y = x--;
        System.out.println(x);
        System.out.println(y);
    }
}
```



y = x - - ;

① y = x

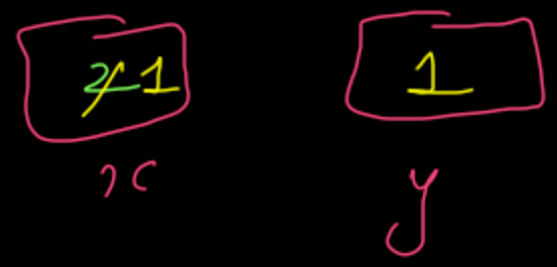
② x = x - 1

x - - ;

↳ x = x - 1 ;

Pre decrement

```java
import java.util.*;
public class Main {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int x = 2;
        int y = --x;
        System.out.println(x);
        System.out.println(y);
    }
}
```



y = --x;

① x = x - 1

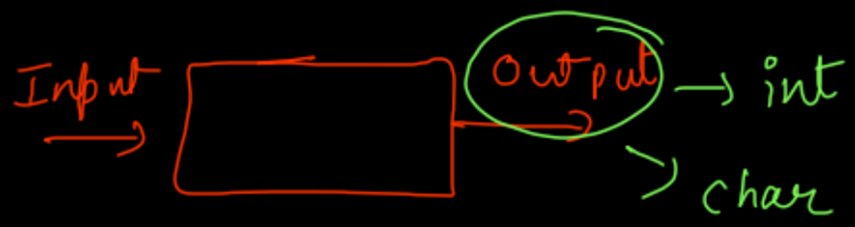
② y = x

functions

```java
import java.util.*;
public class Main {

    public static void counting() {
        //code
    }

    public static void main(String[] args) {
        //some code
    }
}
```

name

Return type



datatype

void means function does not return anything.

```java
import java.util.*;
public class Main {

    public static void counting(int n) {
        //code
    }

    public static void main(String[] args) {
        //some code
    }
}
```

parameter

```java
import java.util.*;
public class Main {

    public static void counting(int n) {
        //code
        for(int i=1;i<=n;i++) {
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        //function call
        counting(10);

    }
}
```

function body

function call

```java
import java.util.*;
public class Main {

    public static void counting(int n) {
        //code
        for(int i=1;i<=n;i++) {
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        //function call
        counting(10);
        System.out.println();
        counting(20);
    }
}
```

Execution of Java program starts from the main method (function)

```java
import java.util.*;
public class Main {
    public static void counting(int n) {
        //code
        for(int i=1;i<=n;i++) {
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        //function call
        int x = 10;
        counting(x);
        System.out.println();
    }
}
```

function definition

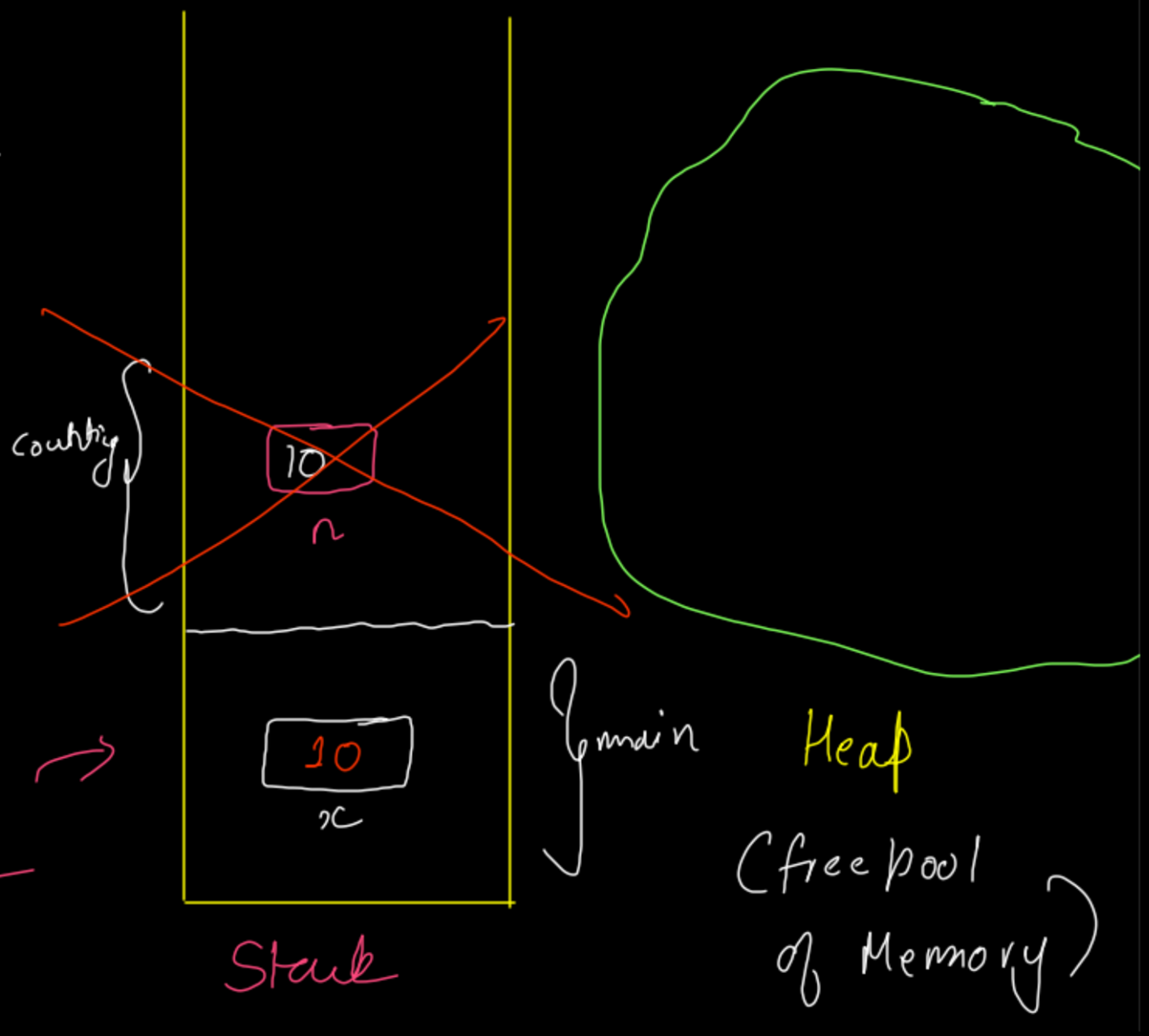
parameter

Pause

argument

1 2 3 4 5 6 7 8 9 10

function call Stack

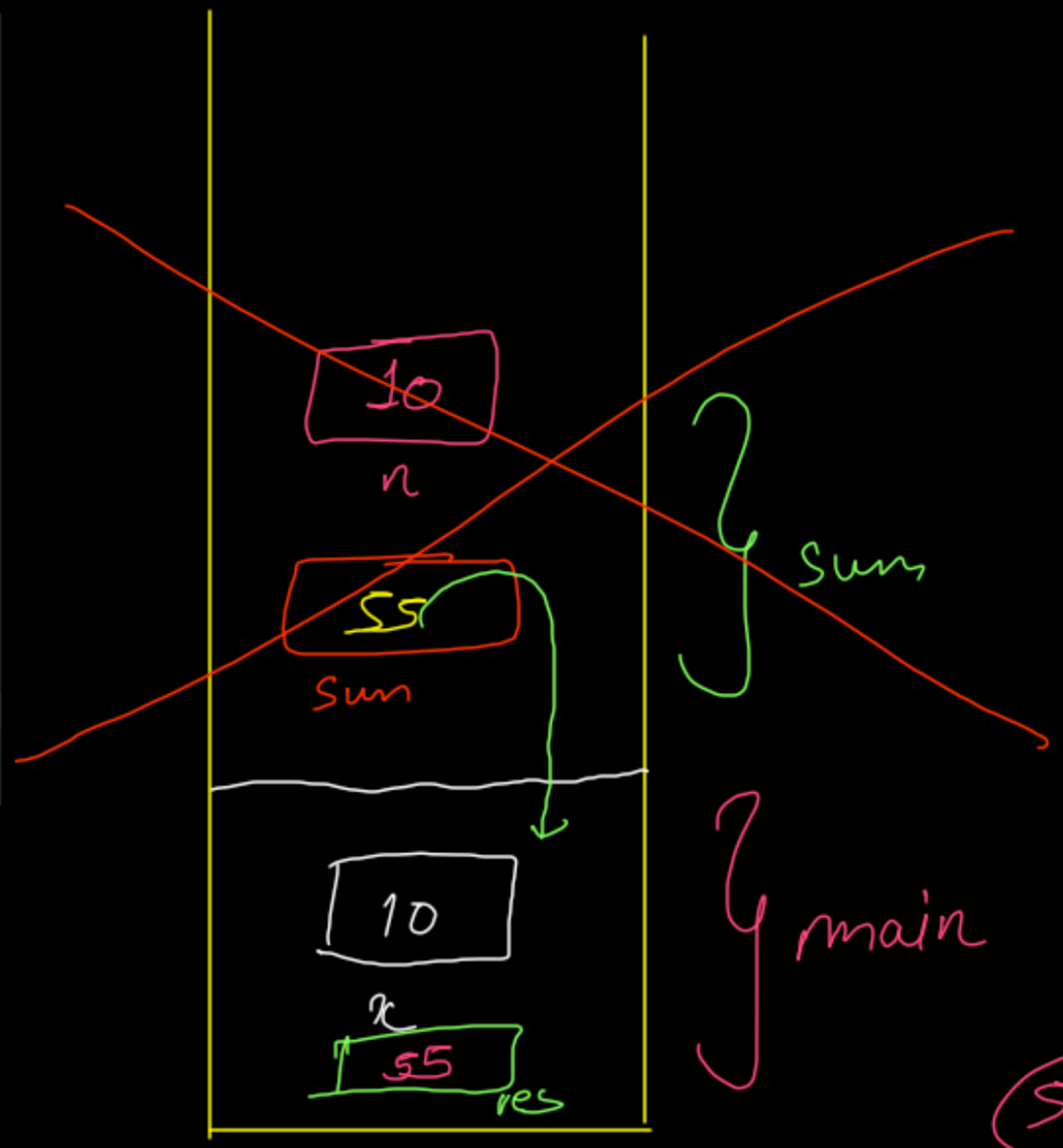


① All primitive datatypes (8) are allocated memory inside the Stack.

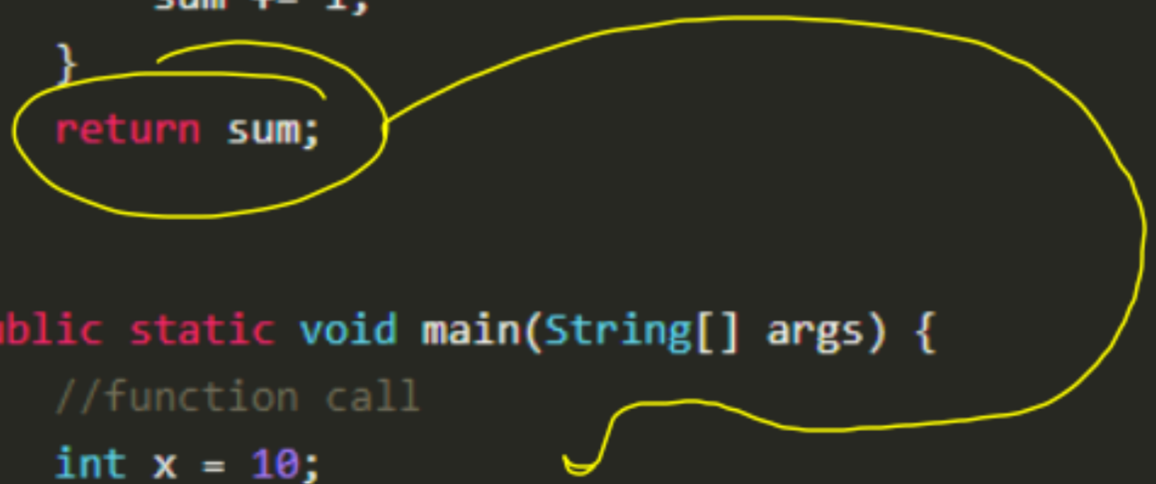
② As soon as all the lines of code of a function are executed, it gets wiped out of the stack.

③ In Java, functions are called Methods because in OOPs any function inside a class is called a Method & in Java everything is inside a class.

```java
import java.util.*;
public class Main {

    public static int sum(int n) {
        //function body
        int sum = 0;
        for(int i=1;i<=n;i++) {
            sum += i;
        }
        return sum;
    }

    public static void main(String[] args) {
        //function call
        int x = 10;
        int res = sum(10);
        System.out.println(res);
    }
}
```



```java
import java.util.*;
public class Main {

    public static int sum(int n) {
        //function body
        int sum = 0;
        for(int i=1;i<=n;i++) {
            sum += i;
        }
        return sum;
    }

    public static void main(String[] args) {
        //function call
        int x = 10;
        System.out.println(sum(10));
    }
}
```



Even Odd

Easy Accuracy: **48.64%** Submissions: **26932** Points: **2**

We've learnt about operators and other basics of CPP. Now, it's time to take another leap and learn to use control structures that helps us choose flow of any code.

Given two integers **a** and **b**. Your task is to print the **even number first and odd number next** in individual lines.
Note: Must print end of the line at the end.

```java
//User function Template for Java

class Solution{
    public void evenOdd(int a, int b){
        // Code here
        if(a % 2 == 0) {
            System.out.println(a);
            System.out.println(b);
        } else {
            System.out.println(b);
            System.out.println(a);
        }
    }
}
```

Even No

N % 2 = 0

odd No

N % 2 != 0
(= 1)

Odd or Even

School Accuracy: **60.31%** Submissions: **13337** Points: **0**

Given a positive integer N, determine whether it is odd or even.

Example 1:

Input:
N = 1
Output:
odd
Explanation:
The output is self-explanatory.

```java
//User function Template for Java
class Solution{
    static String oddEven(int N){
        // code here
        if(N % 2 == 0) {
            return "even";
        } else {
            return "odd";
        }
    }
}
```

System.out.println("|");

This is a

String

Factorial

Basic Accuracy: **53.69%** Submissions: **34040** Points: **1**

Given a positive integer, **N**. Find the factorial of **N**.

Example 1:

Input:
N = 5
Output:
120
Explanation:
5*4*3*2*1 = 120

```java
class Solution{
    static long factorial(int n){
        // code here
        if(n == 0) return 1L;
        long product = 1L;
        for(int i=n;i>=1;i--) {
            product = product * i;
        }

        return product;
    }
}
```

factorial

$N! = N*(N-1)*(N-2)\dots 1$

$= 1*2\times3\dots*(N-1)*(N-2)$

5 4

i

1 2 20 ~~60~~ 120

product

$p = p * i$

(3) * (4)

Problem | Submissions | Solution | New Discuss

Sum or Product

1006 Difficulty: EASY

Avg. time to solve
15 min

Success Rate
80%

Contributed By
Ankush Gupta | Level 1

Problem Statement Suggest Edit

You are given a number 'N' and a query 'Q.' If 'Q' is 1, then you have to return the sum of all integers from 1 to 'N,' else if 'Q' is equal to 2 then you have to return the product of all integers from 1 to 'N.' Since the product can be very large, return it modulo 10 ^ 9 + 7.

For example

```
Given 'N' = 4, 'Q' = 1.
Then the answer is 10 because the sum of all integers between 1 and 4 are 1, 2, 3, and
4. Hence 1 + 2 + 3 + 4 is equal to 10.
```

Sum of 1st N natural No

$N(N+1)/2$

$10^9+7 = (1000000007)$

0 to 10^9+6

$7 \overline{)1}$

0

(1)

$7 \overline{)8}$

1

$=7$

1

$1 \% 7 = 1$

$2 \% 7 = 2$

$3 \% 7 = 3$

$4 \% 7 = 4$

$5 \% 7 = 5$

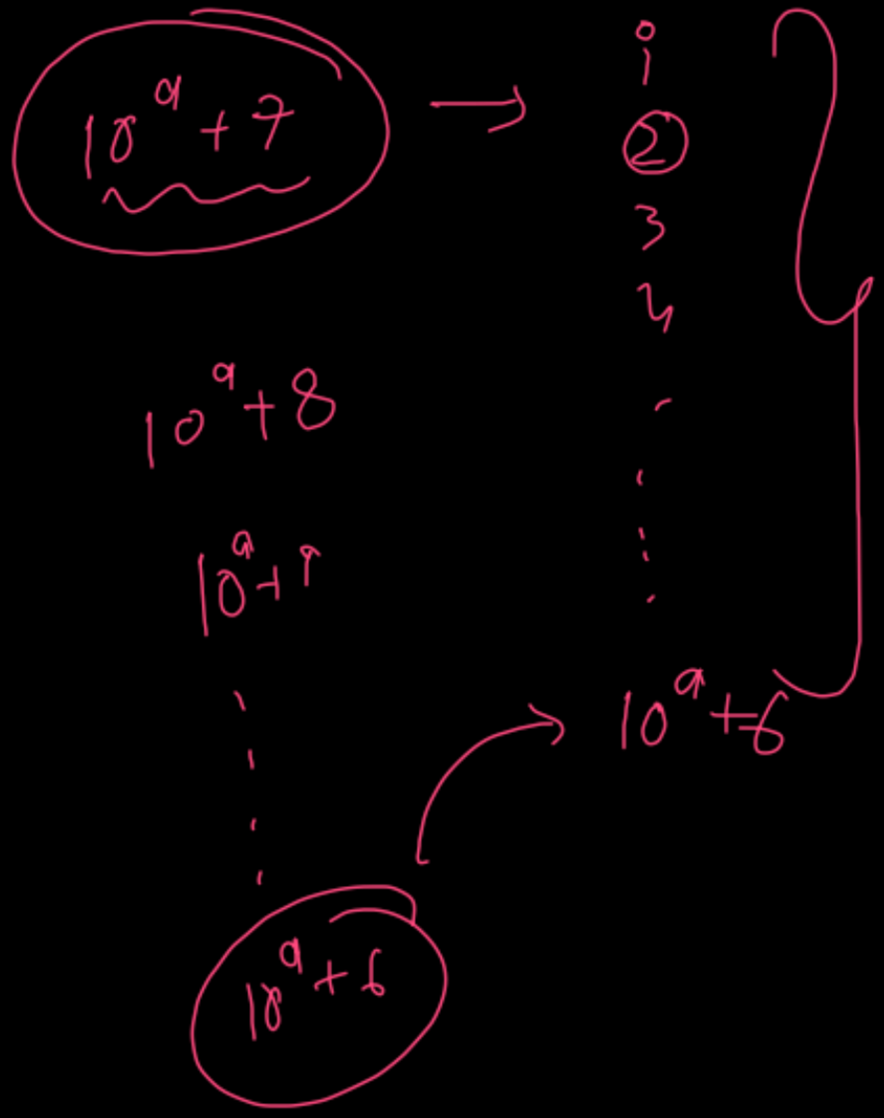
$6 \% 7 = 6$

$7 \% 7 = 0$

$8 \% 7 = 1$

$9 \% 7 = 2$

$\vdots$



INT * INT → Out of Range

(∞) (∞-1)

$N*(N-1)\ldots\ldots 1$

```java
public class Solution {

    public static long sum(int n) {
        return (long)((n*(n+1))/2);
    }

    public static long factorial(int n) {
        if(n == 0) return 1L;
        long product = 1L;
        for(int i=n;i>=1;i--) {
            product = (product * i) % 1000000007;
        }

        return product;
    }
```

```java
    public static long sumOrProduct(int n, int q) {

        if(q == 1) {
            //sum
            return sum(n);
        } else {
            //factorial
            return factorial(n);
        }
    }
```