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

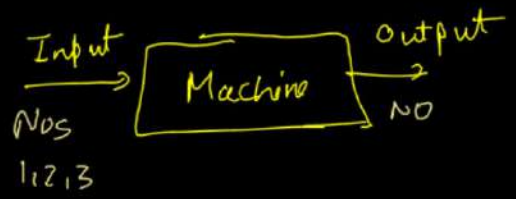
1. Getting Started
2. Arrays & Strings
   - → Two Pointer + Sliding Window
   - → Greedy Algorithms
   - → ☆ Searching & Sorting
3. ☆ Recursion & Backtracking
4. Object Oriented Programming (Java)

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 → instruction + 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");

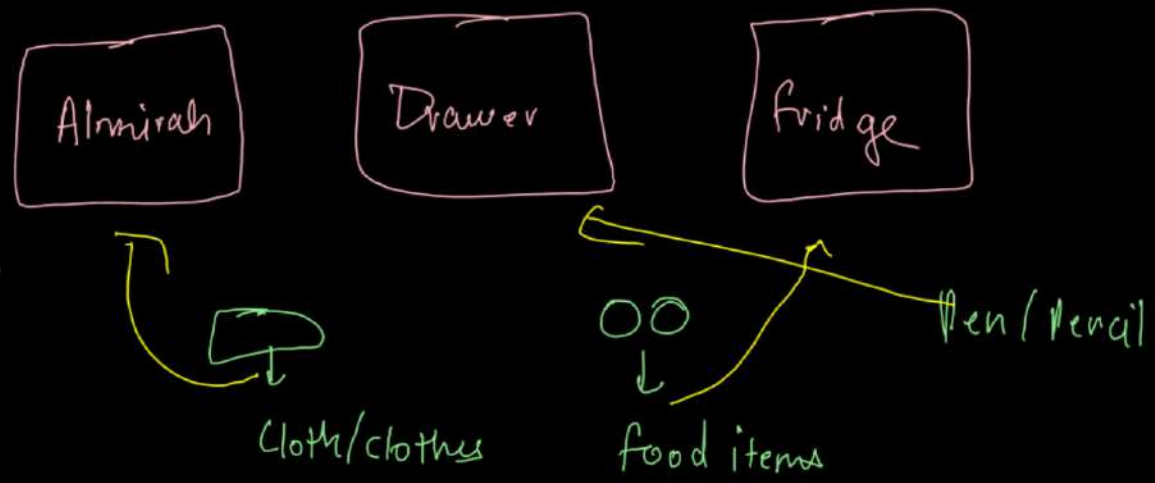
RI Rohan
I

```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 container.



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	

Range of Datatype

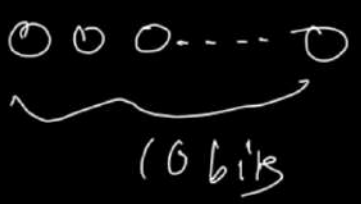
1 byte = 8 bits

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

Range of Short

Size = 2bytes = 2x8 = 16bits

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



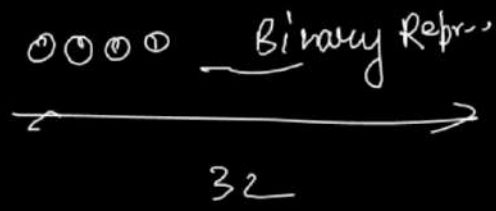
[-32768 to 32767]

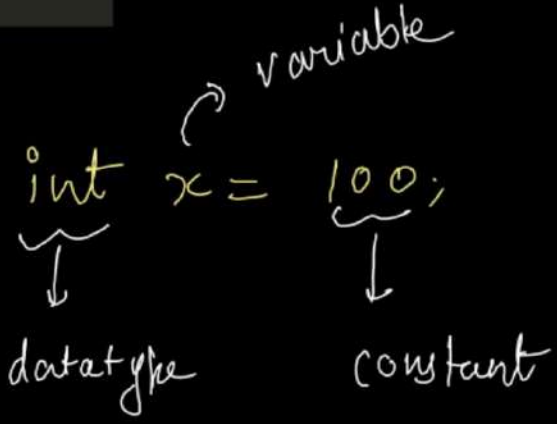
32768 → 17bit

Data loss

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

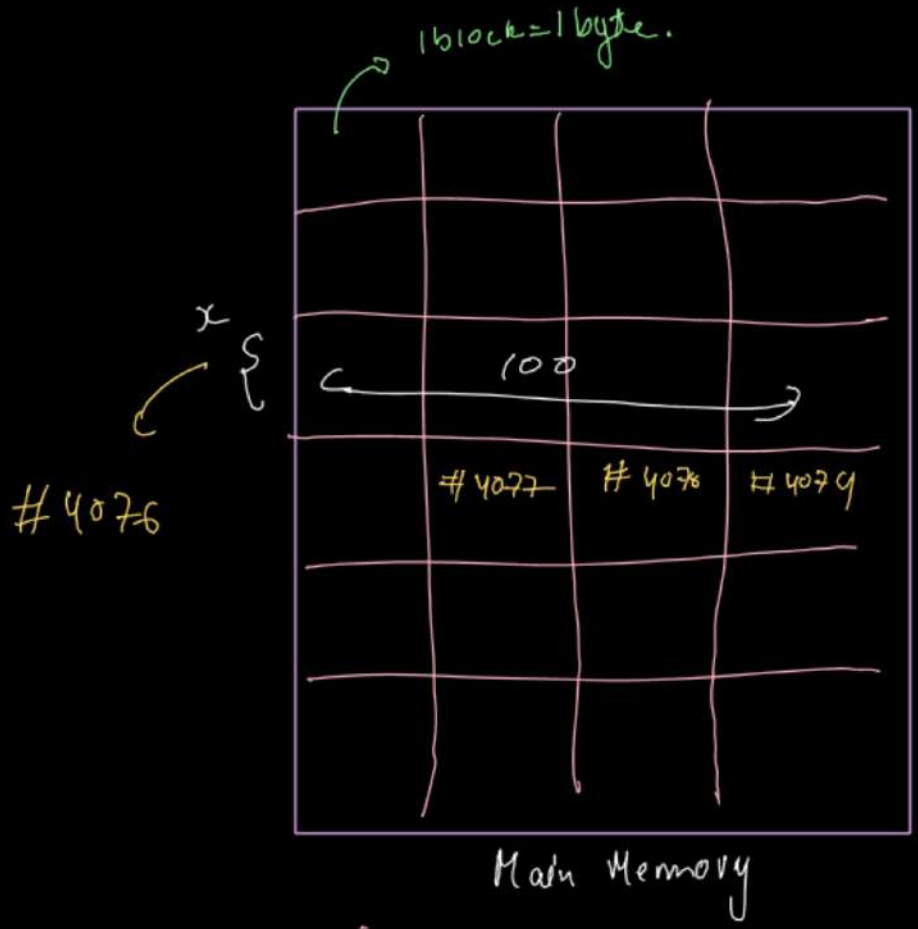
```
Finished in 62 ms
100
```





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);
    }
}
```

int x = 100;

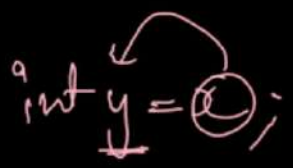
LHS ↓ RHS

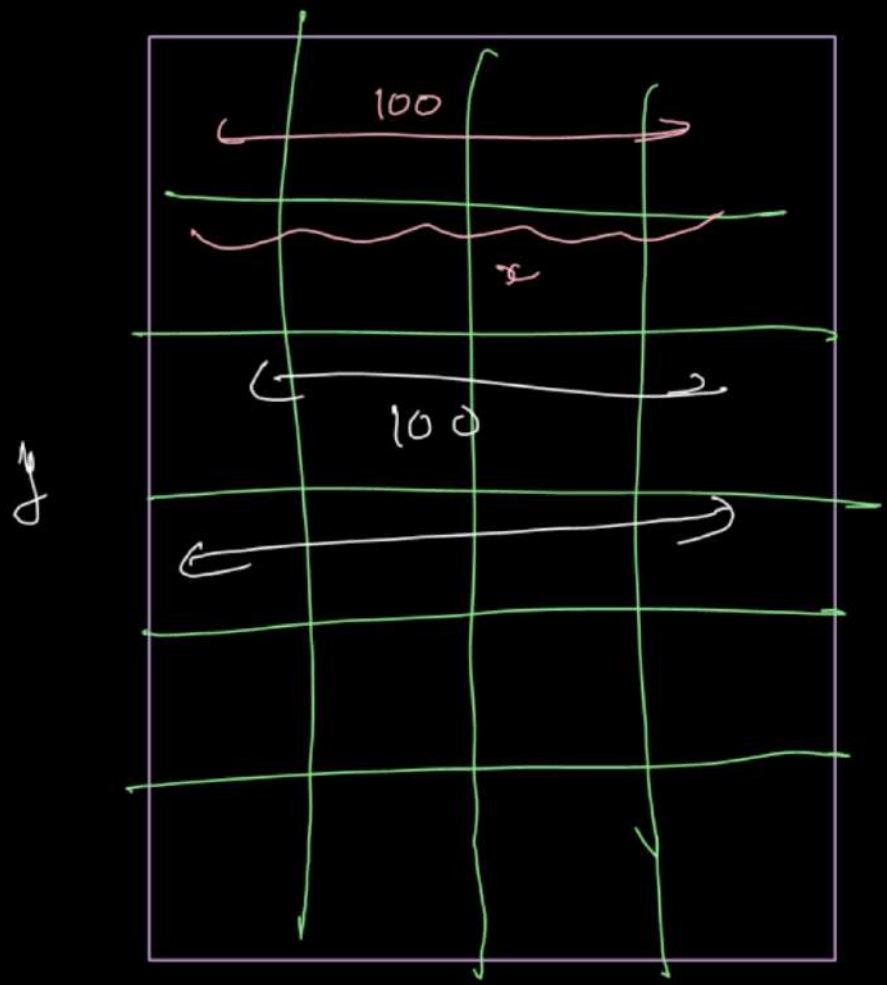
assignment operator

```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);
    }
}
```

RHS gets assigned to LHS

int y = x;


int y = 9;


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);
    }
}
```

$x = \cancel{?} \; 100$

```
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

int x = 1.99

= 1

= 0

```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 IDT 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 /.

10 → 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.

#1 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");
    }
}
```

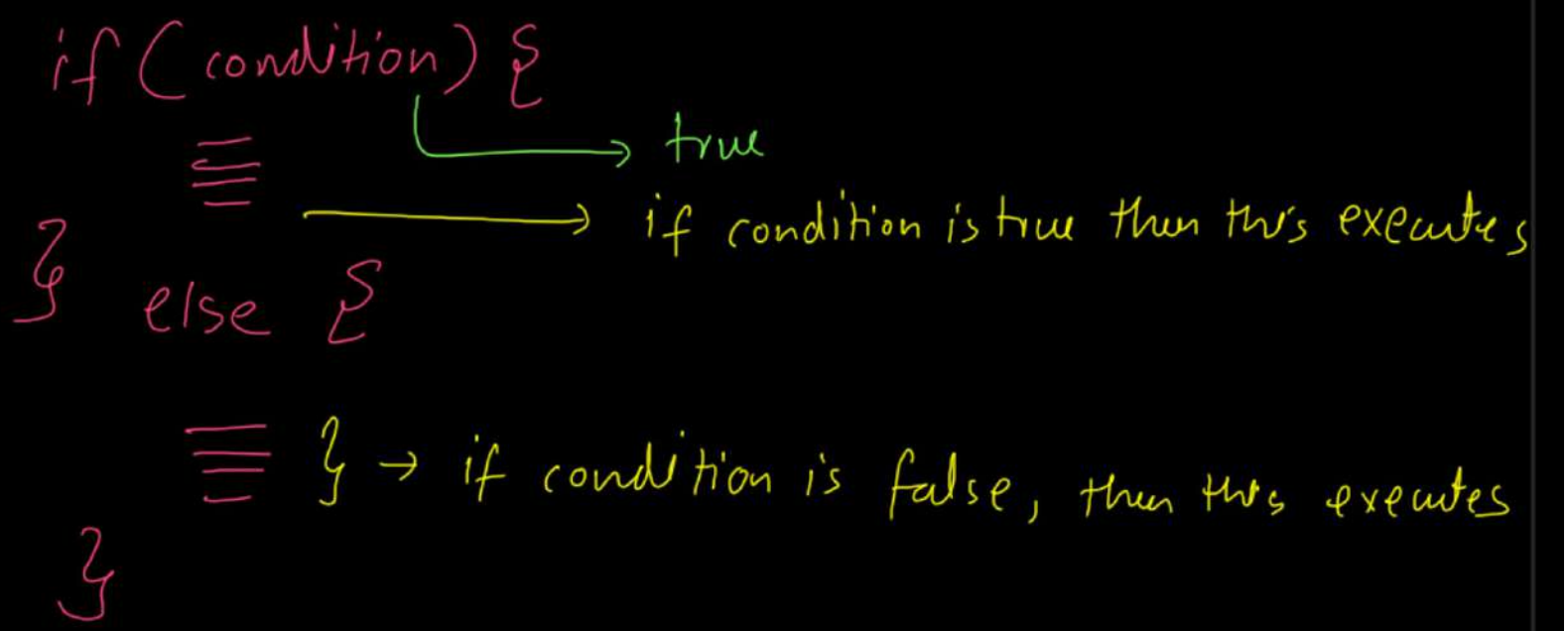
true

```
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

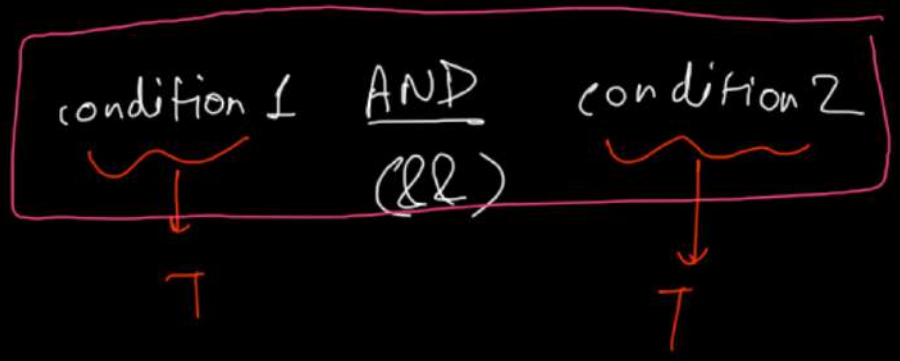

if (condition) {
true
if condition is true then this executes
} else {
if condition is false, then this executes
}

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");
    }
}
```

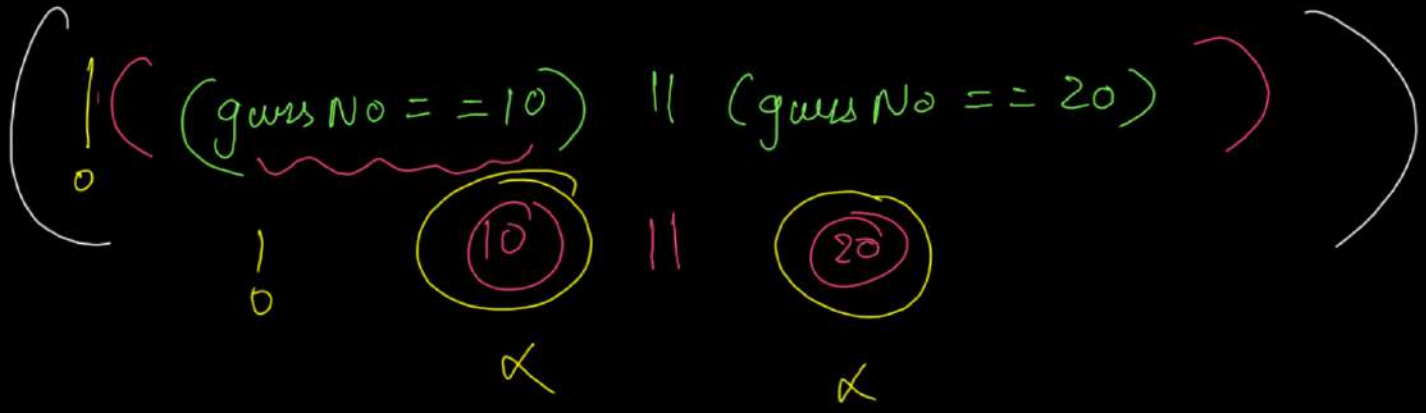
Logical not

! (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");
    }
}
```

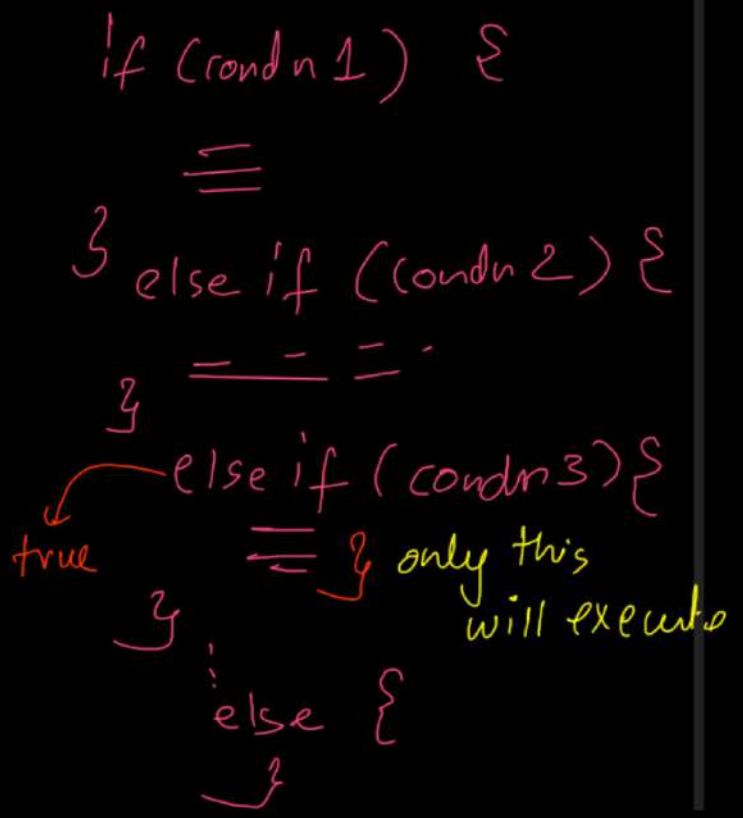
Not equal to

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 (condition 3) {
}
```

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
```

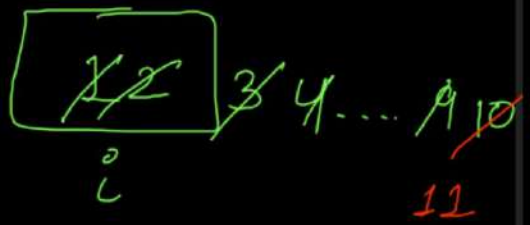
Generally used when exact no of iterations is known.

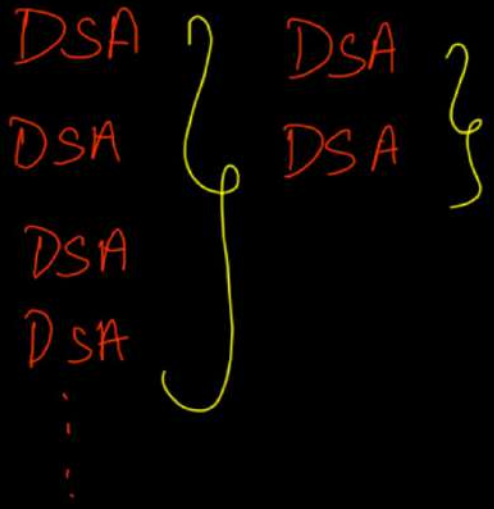
initial condition

condition

```
for (int i=1; i<=10; i++) {
    Syso("DSA"); ②
}
```

① ③





i++ → i = i+1

(1)+1

(2)

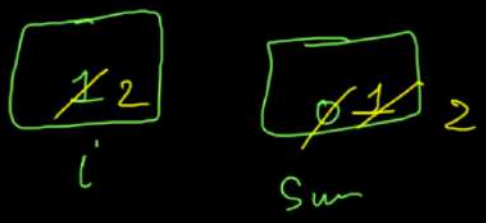
Counting 1 to N

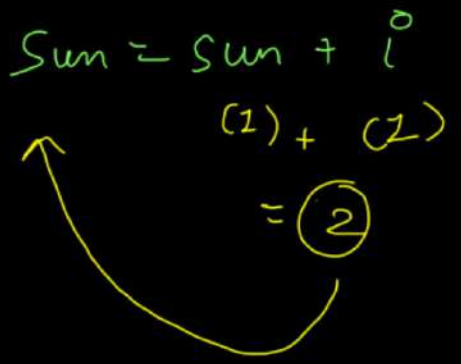
```java
import java.util.*;
public class Main {

    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);
    }
}
```

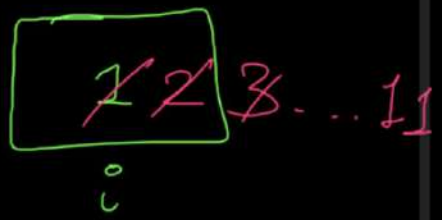




```java
import java.util.*;
public class Main {

    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)

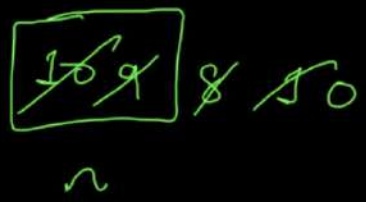
```java
import java.util.*;
public class Main {

    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)
        }
    }
}
```

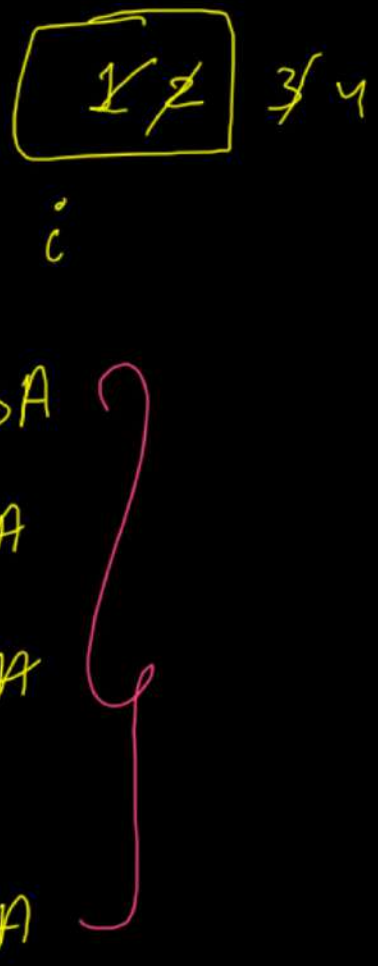


10
9
⋮
1

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++;
}
```

11 ≤ 10 → false

Loop not executed

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

DSA

i = 12

12 ≤ 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);
        }
    }
}
```

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

1
2
4

i % 3 == 0

↓ Skips the current iteration

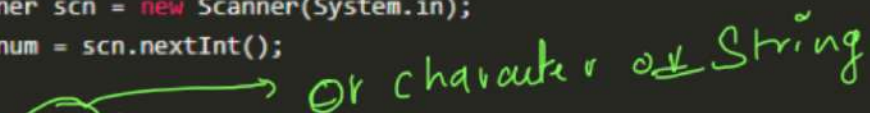
Break in loops

```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");
        }

```



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

or

Operand operator

+
-
++ → pre / post
-- → pre / post

+ (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

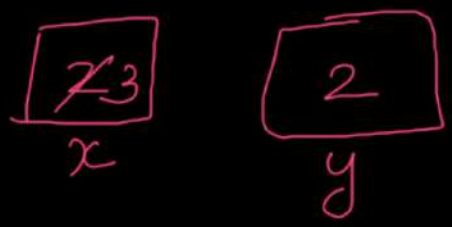
++x; ⟶ x = x + 1

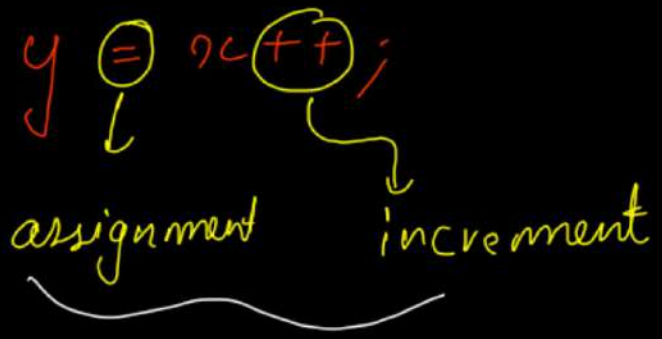
y = x++ ⟶ These ⟵ y = ++x

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);
    }
}
```

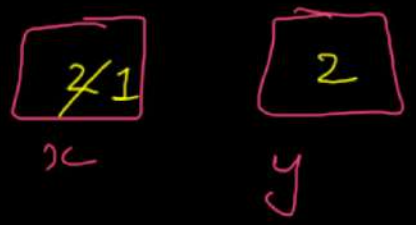
$y = ++x;$

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

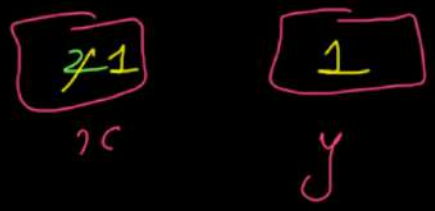
② x = x - 1

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

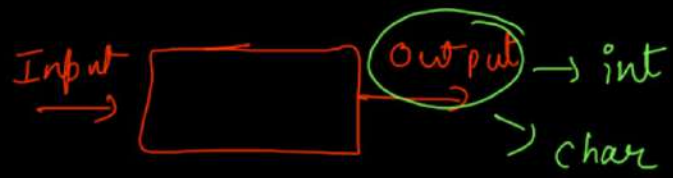
② y = x

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)

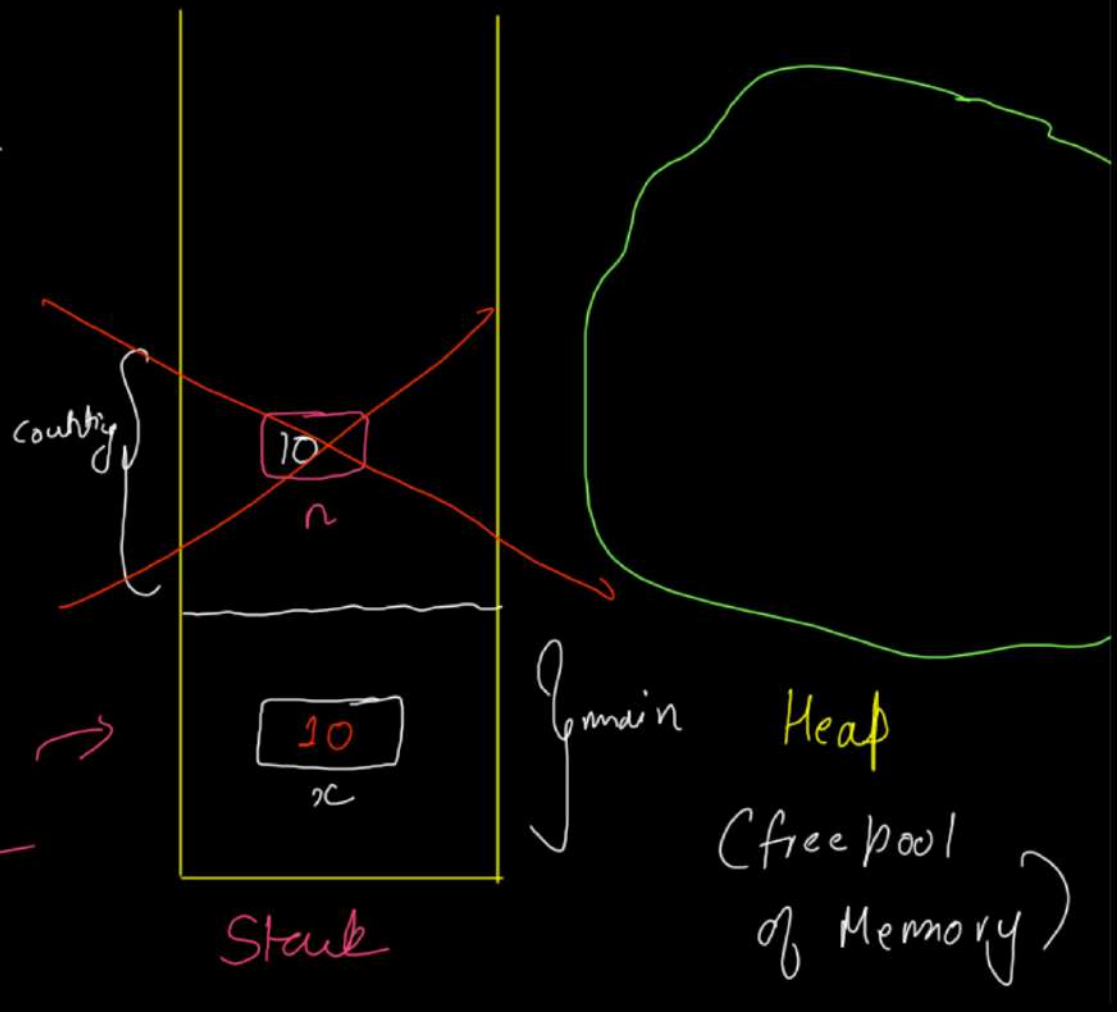
```java
import java.util.*;
public class Main {
    public static void counting(int n) {
        //code
        for(int i=1;i<=n;i++) {
            System.out.println(i);
        }
    }

    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

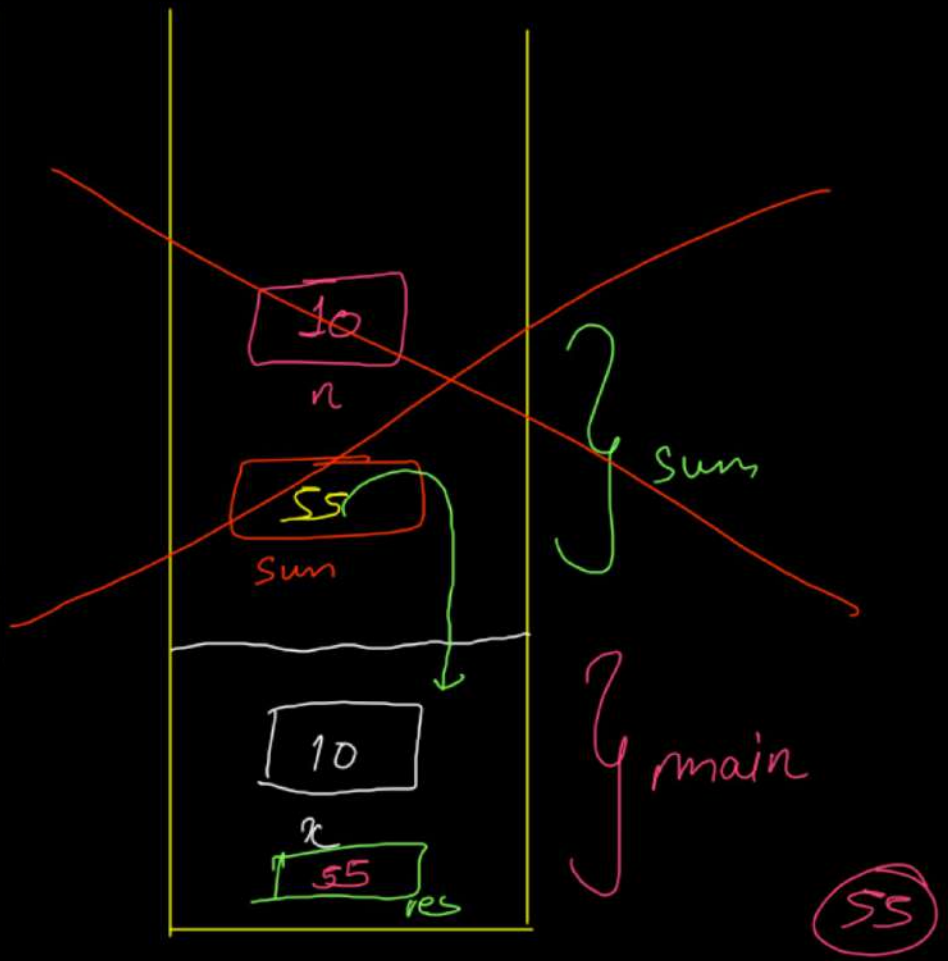
function call Stack

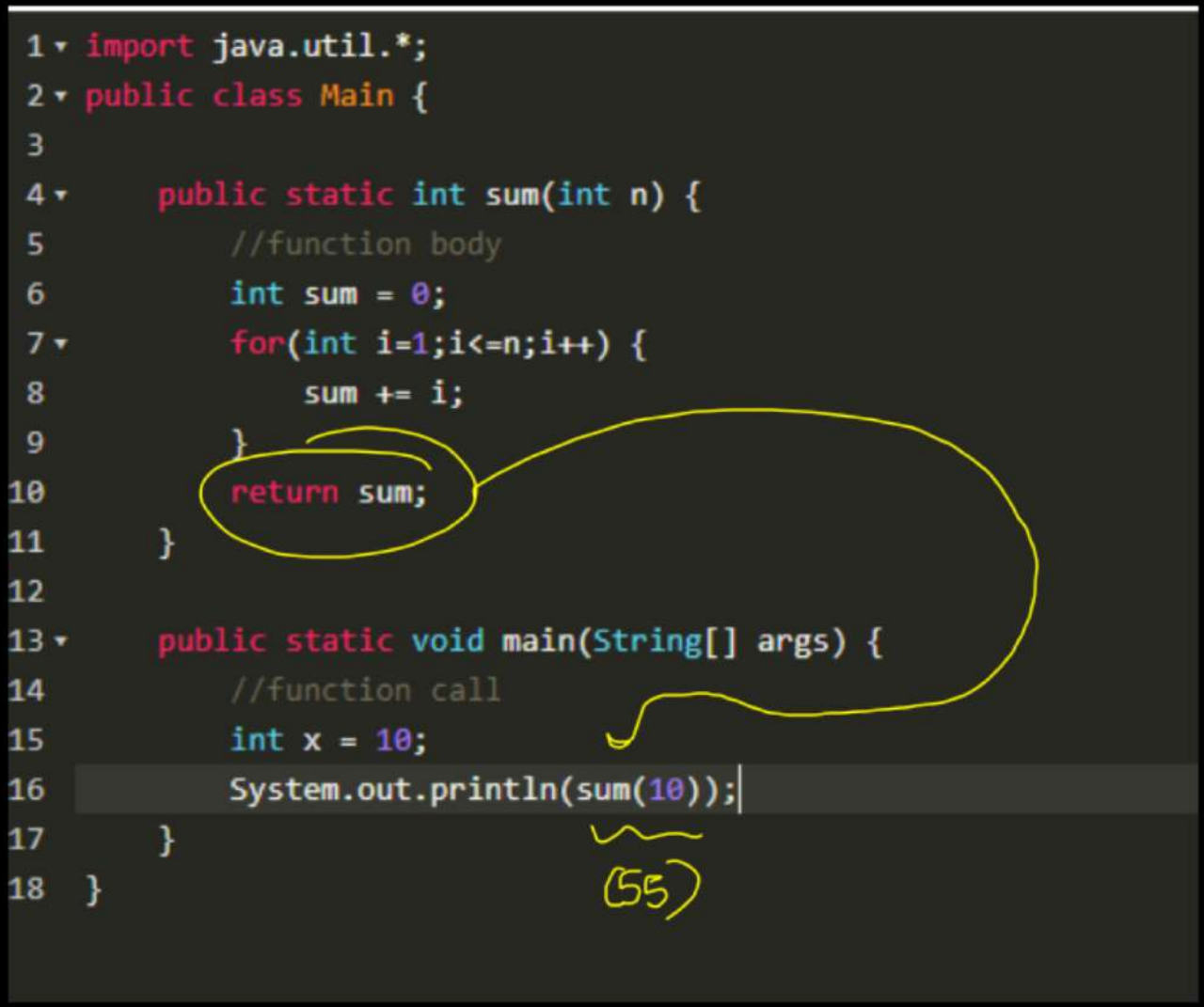
① All primitive datatypes (8) are allocated memory inside the Stack.

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);
    }
}
```




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));
}
}
55

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

```
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 5 20 ~~60~~ 120 — product

$p = p * i$

$(5)*(4)$

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

$N(N+1)/2$

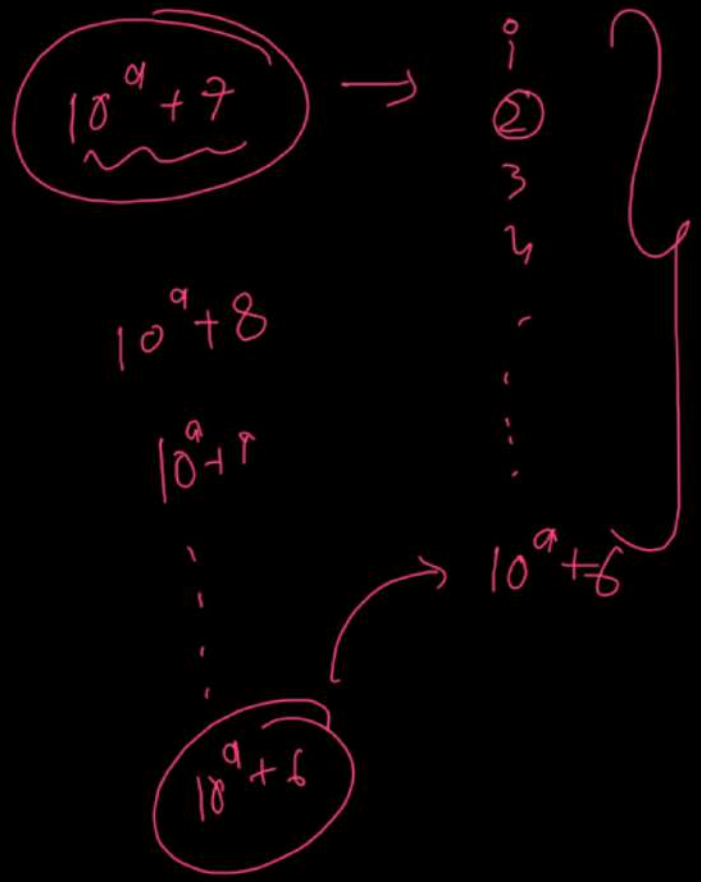
$10^9+7 = (1000000007)$

0 to 10^9+6

$7 \overline{)1}$ → 0 → (1)

$7 \overline{)8}$ → 1, −7 → 1

$1 \% 7 = 1$
$2 \% 7 = 2$
$3 \% 7 = 3$
$4 \% 7 = 4$
$5 \% 7 = 5$
$6 \% 7 = 6$
$7 \% 7 = 0$
$8 \% 7 = 1$
$9 \% 7 = 2$
⋮



INT * INT → Out of Range

(∞) (∞-1)

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

```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);
        }
    }
```

Getting Started - Lecture 4

Basic Syntax

- → Sum of Even Nos Till N
- → Swap 2 Nos ** → 3rd variable
  - → w/o 3rd variable
- → Switch Case

Basic Maths

- → Check Prime → N, N/2, $\sqrt{N}$
- → Prime Nos in Range
- → GCD & LCM → Normal Loop, Euclidean
- → N^{th} fibonacci No
- → Bulb Switcher
- → Power of 2

Java is always pass by value

Sum Of Even Numbers Till N

71

Difficulty: EASY

Avg. time to solve
10 min

Success Rate
90%

Contributed By
Ayush Thakur | Level 1

Problem Statement Suggest Edit

You have been given a number 'N'. Your task is to find the sum of all even numbers from 1 to 'N' (both inclusive).

Example :

```
Given 'N' : 6
Sum of all even numbers till 'N' will be : 2 + 4 + 6 = 12
```

10

1 2 3 4 5 6 7 8 9 10

2+4+6+8+10 → ans

if (condn) {} ∝

{} ∝

```java
public class Solution {
    public static long evenSumTillN(int n) {
        // Write your code here.
        long sum = 0;
        for(int i=1;i<=n;i++) {
            if(i % 2 == 0) sum += i;
        }

        return sum;
    }
}
```

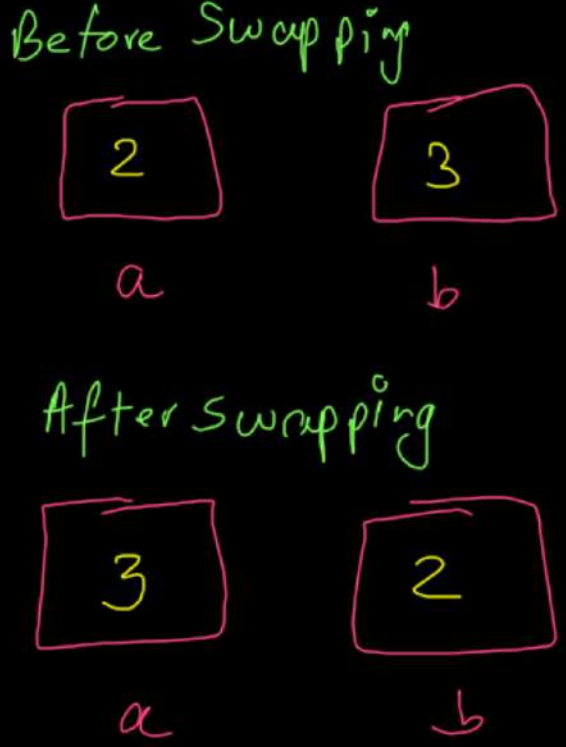
Swap two numbers

School Accuracy: **69.58%** Submissions: **14283** Points: **0**

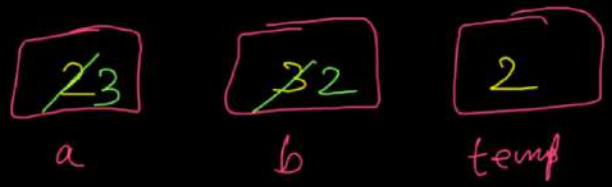
Swap given two numbers and print them. (Try to do it without a temporary variable.) and return it.

Example 1:

Input: a = 13, b = 9
Output: 9 13
Explanation: after swapping it becomes 9 and 13.

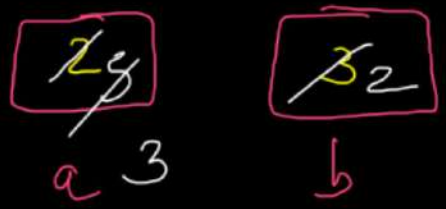


Using 3rd variable



① int temp = a;
② a = b;
③ b = temp;

Without 3rd Variable



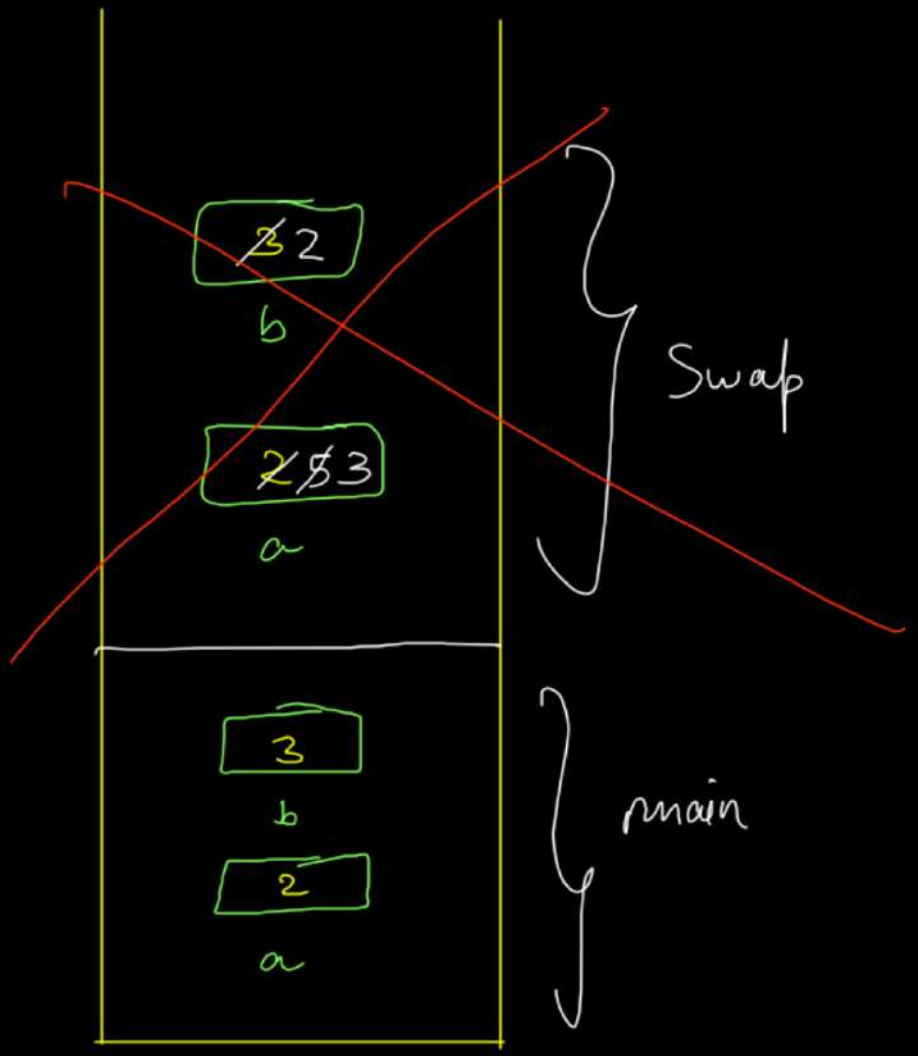
① a = a + b
(7) (3) = (5)

② b = a - b ;
(5) (3) = (2)

③ a = a - b ;
(5) - (2) = (3)

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

    public static void swap(int a, int b) {
        //swapping
        a = a + b;
        b = a - b;
        a = a - b;
    }

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

        System.out.println("Before swapping a = " + a + " b = " + b);
        swap(a,b);
        System.out.println("After swapping a = " + a + " b = " + b);
    }
}
```

Pass by Value



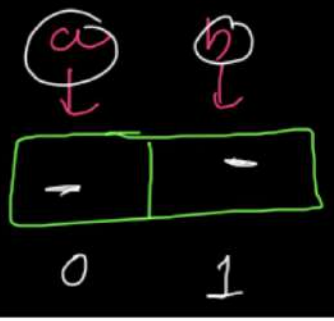
* Java is always pass by value

① Swap with 3rd variable

② W/o 3rd variable

③

④ Put the nos in a data structure (Array, List, LL etc)



→ list (changes will reflect across function)

```java
class Solution{
    static List<Integer> get(int a,int b)
    {
        // code here
        a = a + b;
        b = a - b;
        a = a - b;

        List<Integer> list = new ArrayList<>();

        list.add(a);
        list.add(b);

        return list;
    }
}
```

Java Switch Case statement

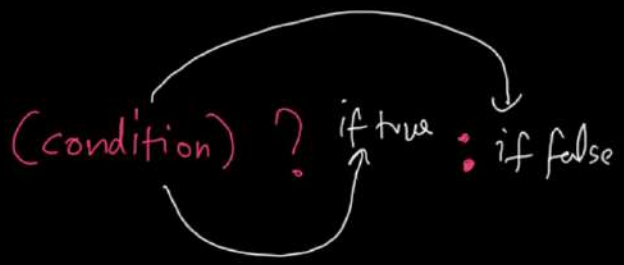
School Accuracy: **66.06%** Submissions: **2535** Points: **0**

Given an integer choice denoting the choice of the user and a list containing the single value R or two values L and B depending on the choice.
If the user's choice is 1, calculate the area of the circle having the given radius(**R**).
Else if choice is 2, calculate the area of the rectangle with given length(**L**) and breadth(**B**).

```java
class Solution{
    static double switchCase(int choice, List<Double> arr){
        // code here

        double r = arr.get(0);
        double l = r;
        double b = (arr.size() == 2) ? arr.get(1) : 0;

        switch(choice) {
            case 1:
                return Math.PI *r *r;
            case 2:
                return l * b;
            default:
            return 0;
        }
    }
}
```

Ternary Operator

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

    public static void swap(int a, int b) {
        //swapping
        a = a + b;
        b = a - b;
        a = a - b;
    }

    public static void main(String[] args) {
        boolean b = true;
        int a = (b == true) ? 1 : 0;
        System.out.println(a);
    }
}
```



Prime Number

Basic Accuracy: **49.19%** Submissions: **56391** Points: **1**

For a given number **N** check if it is prime or not. A prime number is a number which is only **divisible by 1 and itself**.

Example 1:

Input:
N = 5
Output:
1
Explanation:
5 has 2 factors 1 and 5 only.

2 → 2, 1

3 → 3, 1

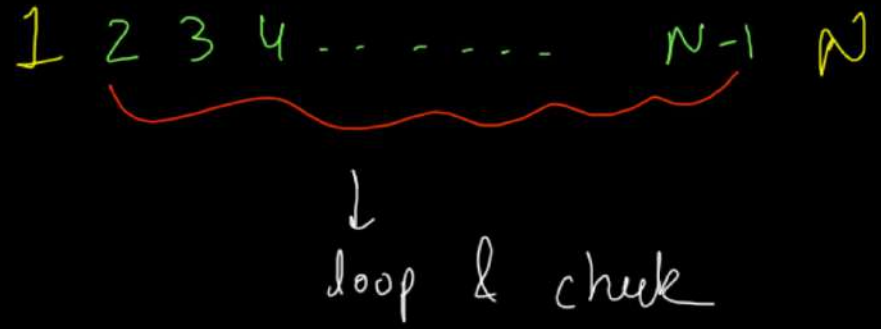
4 → 2, 4, 1 ✗

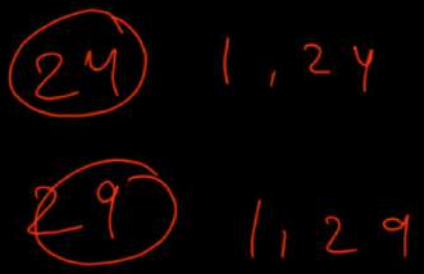
5 → 5, 1

6 → 2, 3, 6, 1 ✗

7 → 2, 7

⋮






```java
class Solution{
    static int isPrime(int N){
        // code here
        if(N == 1) return 0;

        for(int i=2;i<N;i++) {
            if(N % i == 0) return 0;
        }

        return 1;
    }
}
```

→ This code gives TLE.

O(N) solution

27 and 24

27 → (9) less than N/2

24 → (12) equal to N/2

$1 \times 27 = 27$

$3 \times 9 = 27$

(1), 3, (9), (27)

$1 \times 24 = 24$

$2 \times 12 = 24$

$3 \times 8 = 24$

$4 \times 6 = 24$

(1), 2, 3, 4, 6, 8, (12), (24)

24 / 2 = 12

smallest * largest → N/2 is the largest factor in case of even Nos

(2) (12)

25

smallest * largest

```java
class Solution{
    static int isPrime(int N){
        // code here
        if(N == 1) return 0;

        for(int i=2;i<N/2;i++) {
            if(N % i == 0) return 0;
        }

        return 1;
    }
}
```

#Square Root Solution

1x36 = 36
2x18 = 36
3x12 = 36
4x9 = 36
6x6 = 36

1, 2, 3, 4, 6, 9, 12, 18, 36

1x24 = 24
2x12 = 24
3x8 = 24
4x6 = 24

1, 2, 3, 4, 6, 8, 12, 24

1x25 = 25
5x5 = 25

1, 5, 25

29 → 5

$1 \times 29 = 29$

1, 29

5 ↑ → Since there are No factors equal to or before 5 (i.e. $\sqrt{N}$) hence, 29 is a prime No.

Code Conversion

625

2 to 624 1st Method

2 to 317 2nd Method

2 to 25 3rd Method

$i \leq \sqrt{N}$

Squaring both sides

$i^2 \leq N$

$\Rightarrow \boxed{i * i \leq N}$

```java
class Solution{
    static int isPrime(int N){
        // code here
        if(N == 1) return 0;

        for(int i=2;i * i <=N;i++) {
            if(N % i == 0) return 0;
        }

        return 1;
    }
}
```

Find Prime numbers in a range

Medium Accuracy: **63.56%** Submissions: **17866** Points: **4**

Given two integers M and N, generate all primes between M and N including M and N.

```java
boolean isPrime(int n) {

    if(n == 1) return false;

    for(int i=2;i*i<=n;i++) {
        if(n % i == 0) return false;
    }

    return true;
}
```

```java
ArrayList<Integer> primeRange(int lo, int hi) {
    // code here

    ArrayList<Integer> res = new ArrayList<>();

    for(int j=lo;j<=hi;j++) {
       if(isPrime(j) == true) res.add(j);
    }

    return res;
}
```

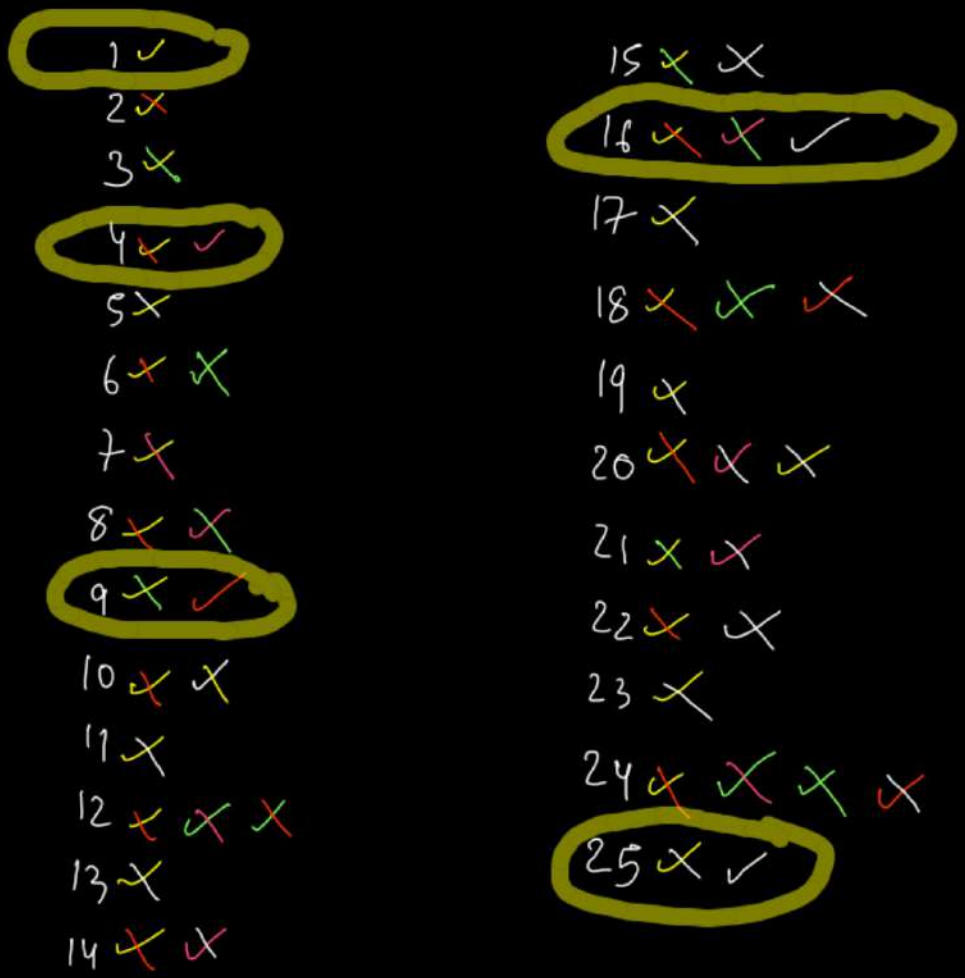
319. Bulb Switcher

Medium 1033 1751 Add to List Share

There are `n` bulbs that are initially off. You first turn on all the bulbs, then you turn off every second bulb.

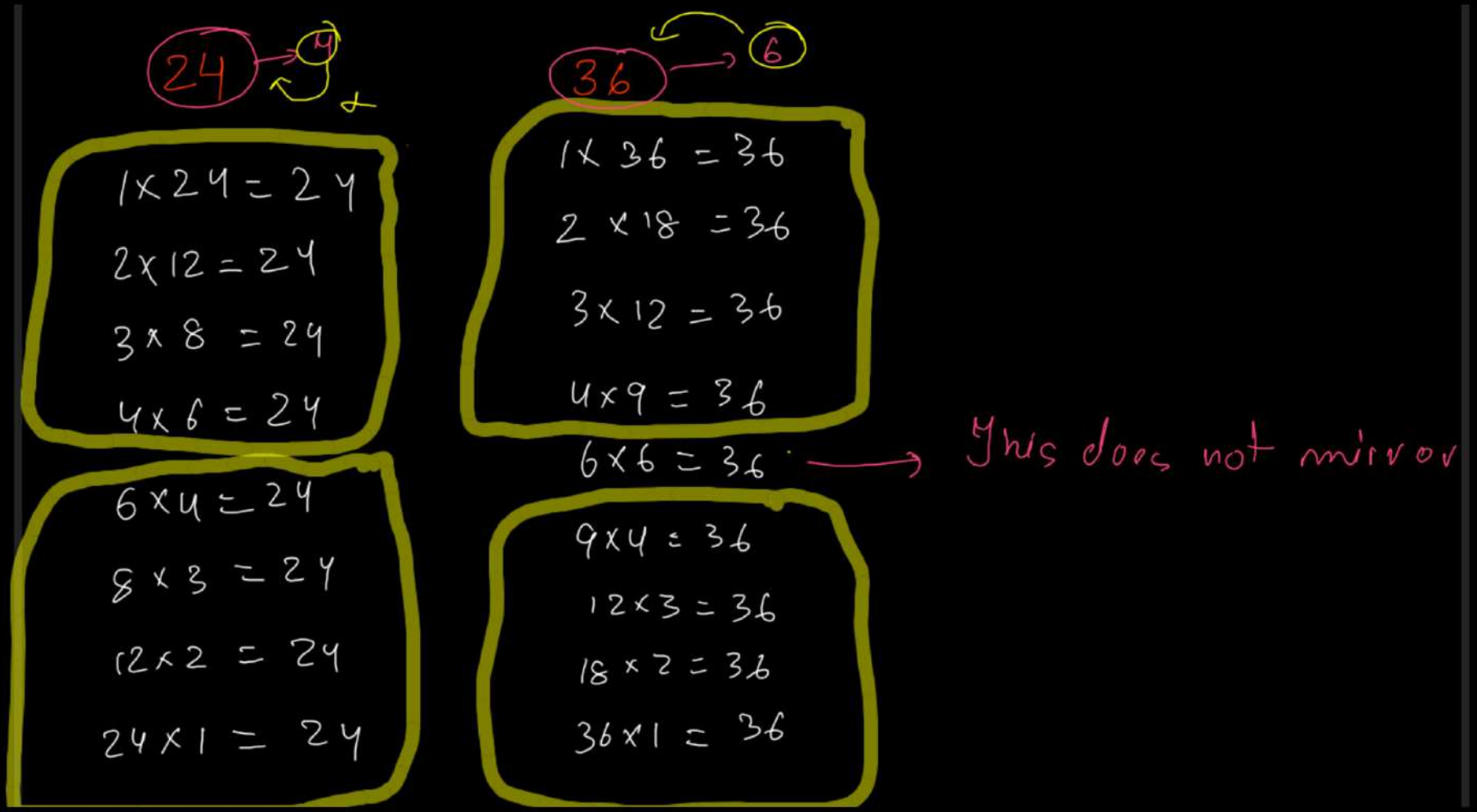
On the third round, you toggle every third bulb (turning on if it's off or turning off if it's on). For the `i^th` round, you toggle every `i` bulb. For the `n^th` round, you only toggle the last bulb.

Return *the number of bulbs that are on after* `n` *rounds.*



✓ → ON

✗ → off


24
4
36
6
1 x 24 = 24
2 x 12 = 24
3 x 8 = 24
4 x 6 = 24
6 x 4 = 24
8 x 3 = 24
12 x 2 = 24
24 x 1 = 24
1 x 36 = 36
2 x 18 = 36
3 x 12 = 36
4 x 9 = 36
6 x 6 = 36
This does not mirror.
9 x 4 = 36
12 x 3 = 36
18 x 2 = 36
36 x 1 = 36

Bulb Switcher

```java
class Solution {
    public int bulbSwitch(int n) {

        int count = 0;
        for(int i=1;i<=n;i++) {
            int sqrt = (int)Math.sqrt(i);
            if(sqrt * sqrt == i) count++;
        }

        return count;
    }
}
```

Power of 2

```java
class Solution {
    public boolean isPowerOfTwo(int n) {
        if(n < 0) return false;

        for(int i=0;i<=30;i++) {
            int power = (int)Math.pow(2,i);
            if(power == n) return true;
        }

        return false;
    }
}
```

→ negative nos cannot be represented as

$n = 2^x$

Nth Fibonacci Number

Easy Accuracy: **41.85%** Submissions: **76409** Points: **2**

Given a positive integer n, find the nth fibonacci number. Since the answer can be very large, return the answer modulo 1000000007.

Example 1:

Input: n = 2
Output: 1
Explanation: 1 is the 2nd number of fibonacci series.

Example 2:

0th 1st 2nd 3rd 4th 5th ...

0 1 1 2 3 5 ...

** Every term is the sum of previous 2

first = 0 (if N==0)

second = 1 (if N==1)

third = 0

```
for(int i=2; i<=N ; i++)
{   third = first + second;
    first = second;
}   second = third;
```

```
//User function Template for Java
class Solution {
    static long nthFibonacci(long n){
        // code here

        long a = 0L;
        long b = 1L;
        long c = 0L;

        for(int i=2;i<=n;i++) {
            c = (a + b) % 1000000007;
            a = b;
            b = c;
        }

        return c;
    }
}
```

5th

a = 0 1 1 2 3

b = 1 1 2 3 5

c = 0 1 2 3 5 (5)

i = 2 3 4 5

c = a + b

2 + 3 = (5)

0 1 1 # This is DP Space Optimized Approach.

LCM And GCD

Basic Accuracy: **55.13%** Submissions: **14347** Points: **1**

Given two numbers **A** and **B.** The task is to find out their LCM and GCD.

$$A * B = LCM * GCD$$

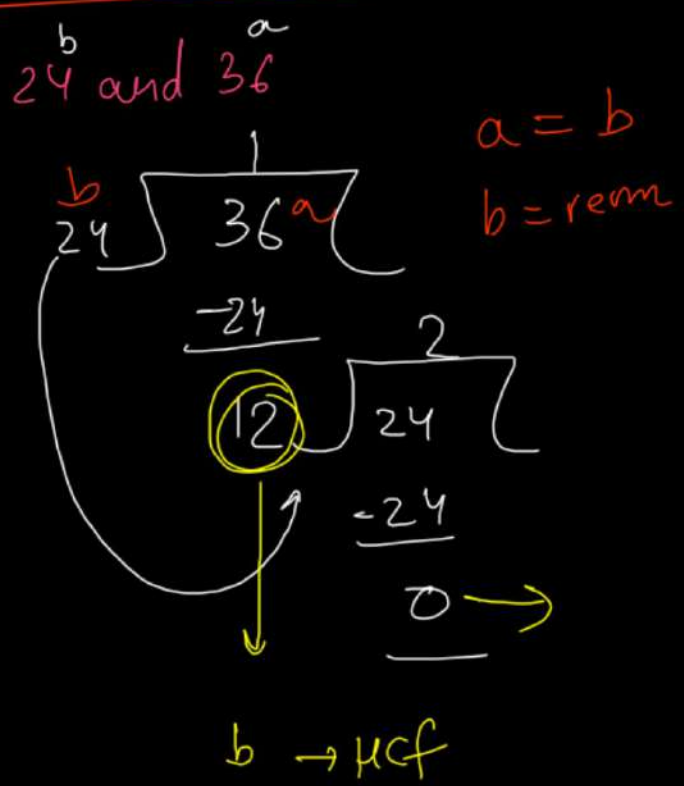
$$LCM = (A * B) / GCD$$

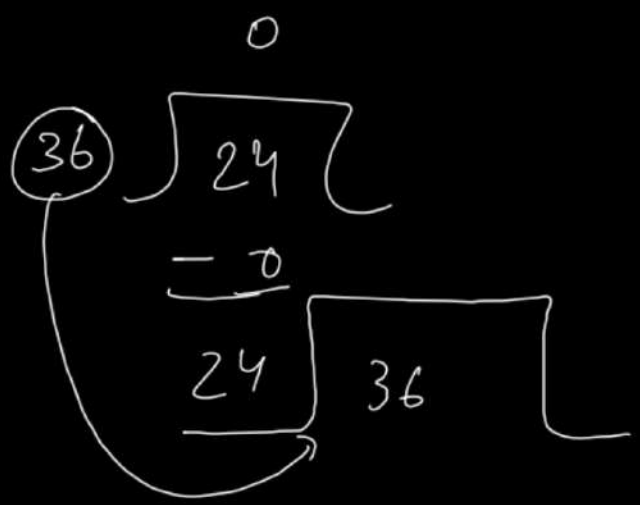
```java
static long gcd(long a,long b) {

    long originalA = a;

    while(a >= 1) {
        if(originalA % a == 0 && b % a == 0) return a;
        a--;
    }

    return 1L;
}
```

→ TLE

```java
static Long[] lcmAndGcd(Long A , Long B) {
    // code here

    long gcd = gcd(A,B);
    long lcm = (A*B)/gcd;

    Long[] res = new Long[2];
    res[0] = lcm;
    res[1] = gcd;
    return res;
}
```

Euclidean Method






```java
static long divisionMethod(long a, long b) {
    while(a % b != 0) {
        long rem = a%b;
        a = b;
        b = rem;
    }

    return b;
}
```

Getting Started - Lecture 4

Digit Traversal

1. Digits of a Number
2. Count Digits of a Number C(ondition Based)
3. Subtract Product & Sum of Digits
4. Rotate a Number
5. Reverse a Number
6. Palindrome Number
7. Armstrong Number
8. Inverse of a Number

Digits of a Number

6 5 7 8 4 3 8 3

3
8
3
4
8
7
5
6

Reverse A Number

Easy

Prev Next

1. You've to display the digits of a number in reverse.
2. Take as input "n", the number for which digits have to be display in reverse.
3. Print the digits of the number line-wise, but in reverse order.

Input Format

"n" where n is any integer.

Input Format

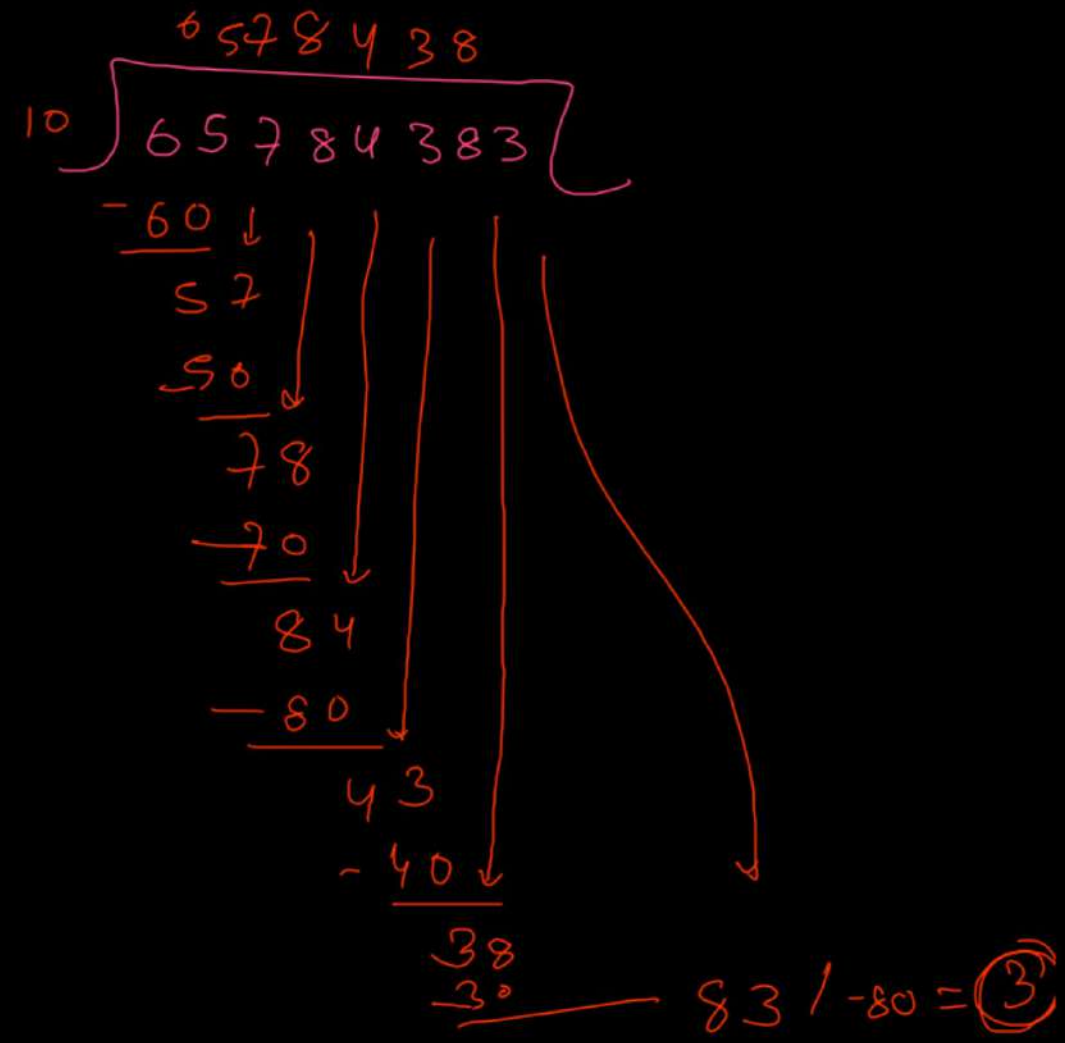
"n" where n is any integer.

Output Format

d1
d2
d3
... digits of the number in reverse

65784383 → n

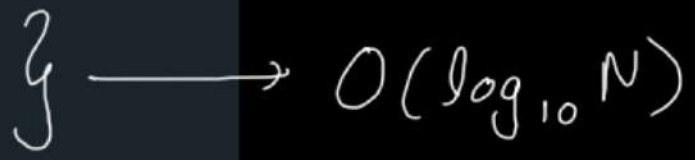
```
int digit = n%10;
n = n/10;
```



	digit (N%10)	N/10
65784383	3	6578438
6578438	8	657843
657843	3	65784
65784	4	6578
6578	8	657
657	7	65
65	5	6
6	6	0
0 → end point		

6/10 = 0.6


```java
public static void main(String[] args) {
  // write your code here
  Scanner scn = new Scanner(System.in);

  int n = scn.nextInt();

  //digits of a number

  if(n == 0) {
      System.out.println(0);
      return;
  }


  while(n > 0) {
     int digit = n % 10;
     System.out.println(digit);
     n /= 10;
  }

 }
}
```



Count Digits

School Accuracy: **42.91%** Submissions: **9265** Points: **0**

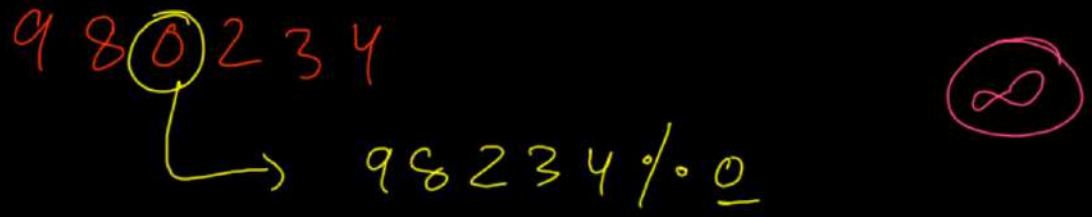
Given a number **N**. Count the number of digits in N which evenly divides N.

Example 1:

Input:
N = 12
Output:
2
Explanation:
1, 2 both divide 12 evenly

N = 124

N	digits	N/10	original N (124)
124	4	12	count = 1
12	2	1	count = 2
1	1	0	count = 3



Arithmetic Exception

↳ / by zero exception

```java
class Solution{
    static int evenlyDivides(int N){
        // code here
        int originalN = N;
        int count = 0;

        while(N > 0) {
            int digit = N % 10;
            if(digit != 0 && originalN % digit == 0) {
                count++;
            }
            N /= 10;
        }

        return count;
    }
}
```

$O(\log_2 N)$

Rotate A Number

Easy

Prev Next

1. You are given two numbers n and k. You are required to rotate n, k times to the right. If k is positive, rotate to the right i.e. remove rightmost digit and make it leftmost. Do the reverse for negative value of k. Also k can have an absolute value larger than number of digits in n.
2. Take as input n and k.
3. Print the rotated number.
4. Note - Assume that the number of rotations will not cause leading 0's in the result. e.g. such an input will not be given

   n = 12340056

   k = 3

   r = 05612340

562984 → N → nod = (6)

→ k (No of rotations)

k % nod ?

(0) k = 0 562984 ⟷ k = 6 562984 (0)

(1) k = 1 456298 ⟷ k = 7 456298 (1)

k = 2 845629 k = 8

k = 3 984562 k = 9

k = 4 298456 k = 10

k = 5 629845 k = 11

$$
\begin{array}{r}
166 \\
6 \overline{)1000} \\
-6\,\downarrow \\
\hline
40 \\
-36 \\
\hline
40 \\
-36 \\
\hline
4
\end{array}
$$

✓

① k = k % nod (To convert large values of k to small values)

(0 to Nod-1)

Nod = 6

#② if (k < 0)

k = k + nod

k = 0 562984

k = ① 456298 ⟷ k = -5 456298

k = ② 845629 ⟷ k = -4 845629

k = ③ 984562 ⟷ k = ③ 984562

k = ④ 298456 ⟷ k = -2 298456

k = ⑤ 629845 ⟷ k = -1 629845

$N = 562984$

$k = 2$

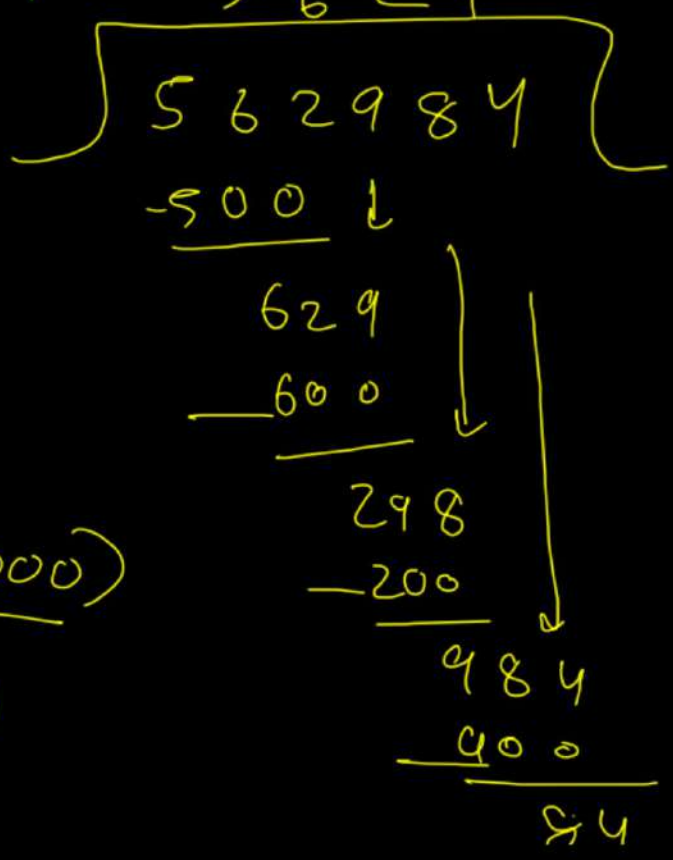
845629

$Last = N \% 10^k$ (84)

$N = N / 10^k$ (5629)

$Last = Last * 10^{nod-k}$ (840000)

$N = Last + N$ (845629)



```java
public static void main(String[] args) {
  // write your code here

  Scanner scn = new Scanner(System.in);
  int n = scn.nextInt();
  int k = scn.nextInt();

  int temp = n;
  int nod = 0;
  while(temp > 0) {
      nod++;
      temp /= 10;
  }

  k = k % nod;
  if(k < 0) k = k + nod;

  int last = n % (int)Math.pow(10,k);
  n = n / (int)Math.pow(10,k);
  last = last * (int)Math.pow(10,nod-k);
  n = n + last;

  System.out.println(n);

 }
}
```

$-1000 < k$

$k = k \% nod$

$-(1000 \% 6)$

$k = -4$

1281. Subtract the Product and Sum of Digits of an Integer

Easy 1506 196 Add to List Share

Given an integer number `n`, return the difference between the product of its digits and the sum of its digits.

Example 1:

```
Input: n = 234
Output: 15
Explanation:
Product of digits = 2 * 3 * 4 = 24
Sum of digits = 2 + 3 + 4 = 9
Result = 24 - 9 = 15
```

No = 234

N	digits	Sum (0)	Product (1)
234	4	4	4
23	3	7	12
2	2	9	24
0			

24 - 9 = 15

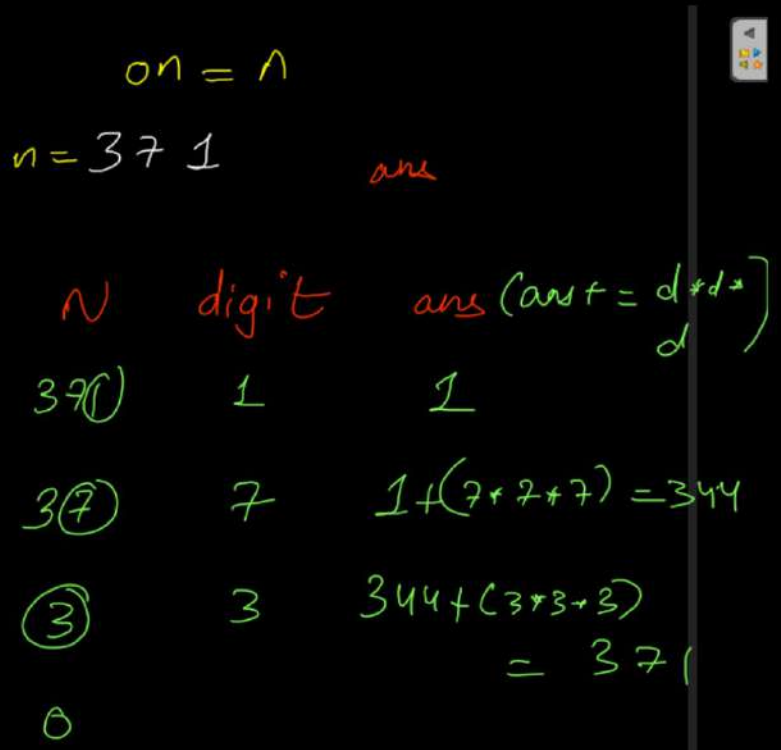
```java
class Solution {
    public int subtractProductAndSum(int n) {
        int sum = 0;
        int product = 1;

        while(n > 0) {
            int digit = n % 10;
            sum += digit;
            product *= digit;
            n /= 10;
        }

        return product - sum;
    }
}
```

Armstrong Numbers

School Accuracy: **53.34%** Submissions: **21079** Points: **0**

For a given **3** digit number, find whether it is armstrong number or not. An **Armstrong number** of three digits is an integer such that the sum of the cubes of its digits is equal to the **number** itself. Return "**Yes**" if it is a armstrong number else return "**No**".

NOTE: 371 is an **Armstrong number** since $3^3 + 7^3 + 1^3 = 371$



```java
//User function Template for Java
class Solution {
    static String armstrongNumber(int n){
        // code here

        int originalN = n;
        int ans = 0;
        while(n > 0) {
            int d = n % 10;
            n/=10;
            ans += d*d*d;
        }

        if(ans == originalN) return "Yes";
        return "No";
    }
}
```

7. Reverse Integer

Medium 8234 10570 Add to List Share

Given a signed 32-bit integer `x`, return `x` *with its digits reversed.* If reversing `x` causes the value to go outside the signed 32-bit integer range `[-2^31, 2^31 - 1]`, then return `0`.

Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

```
Input: x = 123
Output: 321
```

Example 2:

```
Input: x = -123
Output: -321
```

1234

rev = 0

0

rev = (rev * 10) + digit

No	digit	N/10	Rev
1234	4	123	4
123	3	12	43
12	2	1	432
1	1	0	4321

```java
class Solution {
    public int reverse(int x) {
        int rev = 0;

        while(x != 0) {
            int digit = x % 10;

            rev = (rev * 10) + digit;
            x /= 10;
        }

        return rev;
    }
}
```

Integer

$\infty * 10 \rightarrow$ out of Range

$-\infty * 10 \rightarrow$ out of Range

$rev * 10 > Integer.MAX\ value$

$rev > \frac{Integer.MAX\_VALUE}{10}$

$rev * 10 < Integer.MIN\_VALUE$

$rev < \frac{Integer.MIN\_VALUE}{10}$

```java
class Solution {
    public int reverse(int x) {
        int rev = 0;

        while(x != 0) {
            int digit = x % 10;

            if((rev > Integer.MAX_VALUE/10) || (rev < Integer.MIN_VALUE/10))
                return 0;

            rev = (rev * 10) + digit;
            x /= 10;
        }

        return rev;
    }
}
```

9. Palindrome Number

Easy 7012 2274 Add to List Share

Given an integer `x`, return `true` if `x` is palindrome integer.

An integer is a **palindrome** when it reads the same backward as forward.

- For example, `121` is a palindrome while `123` is not.

Example 1:

```
Input: x = 121
Output: true
Explanation: 121 reads as 121 from left to right and
from right to left.
```

123

```java
class Solution {

    public int reverse(int x) {
        int rev = 0;

        while(x != 0) {
            int digit = x % 10;
            rev = (rev * 10) + digit;
            x /= 10;
        }

        return rev;
    }

    public boolean isPalindrome(int x) {
        if(x < 0) return false;
        int reverse = reverse(x);
        if(reverse == x) return true;
        return false;
    }
}
```

Inverse Of A Number

Easy

Prev Next

1. You are given a number following certain constraints.
2. The key constraint is if the number is 5 digits long, it'll contain all the digits from 1 to 5 without missing any and without repeating any. e.g. 23415 is a 5 digit long number containing all digits from 1 to 5 without missing and repeating any digit from 1 to 5.Take a look at few other valid numbers - 624135, 81456273 etc.Here are a few invalid numbers - 139, 7421357 etc.
3. The inverse of a number is defined as the number created by interchanging the face value and index of digits of number.e.g. for 426135 (reading from right to left, 5 is in place 1, 3 is in place 2, 1 is in place 3, 6 is in place 4, 2 is in place 5 and 4 is in place 6), the inverse will be 416253 (reading from right to left, 3 is in place 1, 5 is in place 2,2 is in place 3, 6 is in place 4, 1 is in place 5 and 4 is in place 6) More examples - inverse of 2134 is 1243 and inverse of 24153 is 24153
4. Take as input number "n", assume that the number will follow constraints.
5. Print it's inverse.

Input Format

"n" where n is any integer, following constraints defined above.

N → 1 to N digits

8 7 6 5 4 3 2 1
2 8 3 4 6 7 5 1 → place value

→ face value

Number

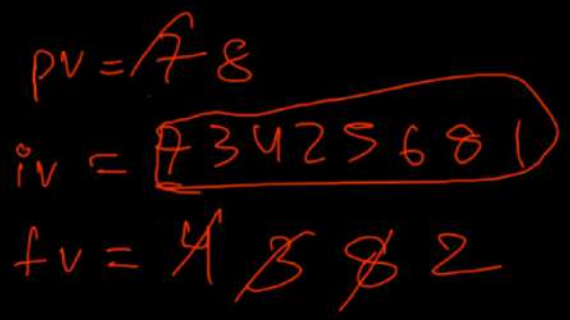
$$\sum fv * 10^{pv-1}$$

4 3 2 1
1 2 7 8

8 7 6 5 4 3 2 1
7 3 4 2 5 6 8 1 → Inverse

$$\sum pv * 10^{fv-1} \rightarrow \text{Inverse}$$

$$8 * 10^{0} + 7 * 10^{1} + 2 * 10^{2} + 1 * 10^{3}$$



$8 * 10^{1}$

$= 80$

```java
import java.util.*;

public class Main{

public static void main(String[] args) {
  // write your code here
  Scanner scn = new Scanner(System.in);
  int n = scn.nextInt();

  int pv = 1;
  int inverse = 0;

  while(n > 0) {
      int fv = n % 10;
      n /= 10;

      inverse += pv * (int)Math.pow(10,fv-1);
      pv++;
  }

  System.out.println(inverse);

 }
}
```

Getting Started - Lecture 6
(Pattern Printing)

Concepts ↓

1. Pattern Printing GfG
2. Right Angled Triangle
3. Patternify
4. Right Angled Triangle -2
5. Yet Another Pattern
6. Diamond of Stars
7. Void of Diamond
8. Backward Slash
9. forward Slash
10. Cross of Stars

Pattern - Printing (GfG)

Pattern Printing

School Accuracy: **65.5%** Submissions: **2776** Points: **0**

Given a number **N**. The task is to print a series of asterisk(*****) from **1 till N** terms with increasing order and difference being 1.

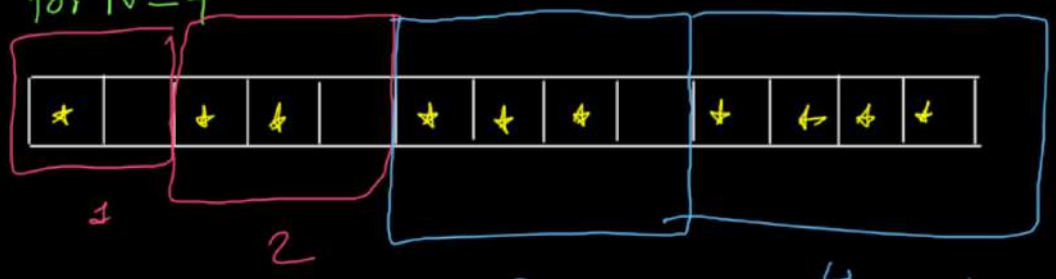
Example 1:

Input:
N = 3
Output:
* ** ***
Explanation:
First, print 1 asterisk then space after
that print 2 asterisk and space after that
print 3 asterisk now stop as N is 3.

for N=3

Task → Print *
Print space

for N=4



↳ ith iteration → istars

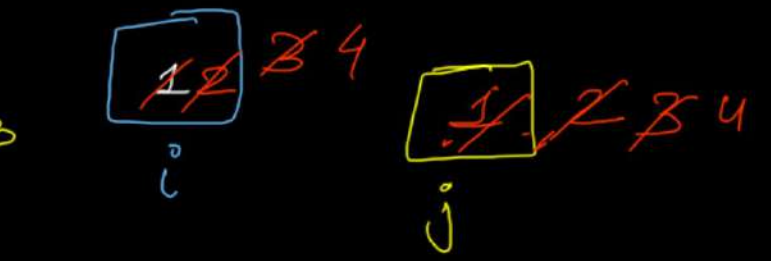
```java
//User function Template for Java
class Solution{
    static void printPattern(int n){
        // code here

        for(int i=1;i<=n;i++) {

            for(int j=1;j<=i;j++) {
                System.out.print("*");
            }

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

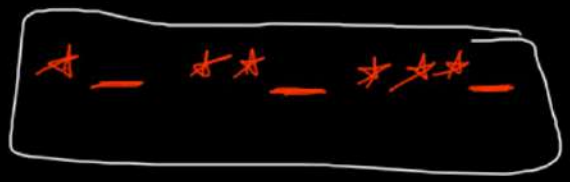
1 3

2

Nested Loop → One or many loops inside another

N = 3





Right Angled Triangle

Pattern 1

Easy

Prev Next

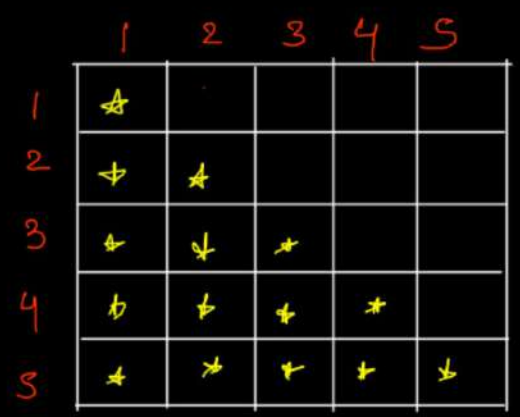
1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

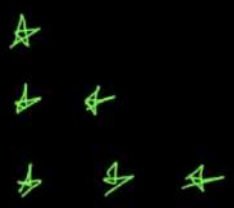
A number n

Output Format

for N=5

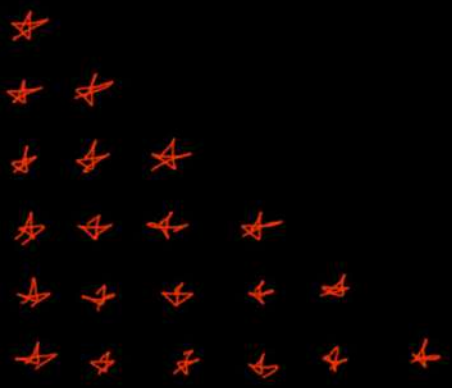


for N=3



for N=3

for N=6



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

    // write ur code here
    int n = scn.nextInt();

     for(int i=1;i<=n;i++) {

        for(int j=1;j<=i;j++) {
            System.out.print("*\t");
        }

        System.out.println();
    }

}
```

#Patternify

Pattern 2

Easy

Prev Next

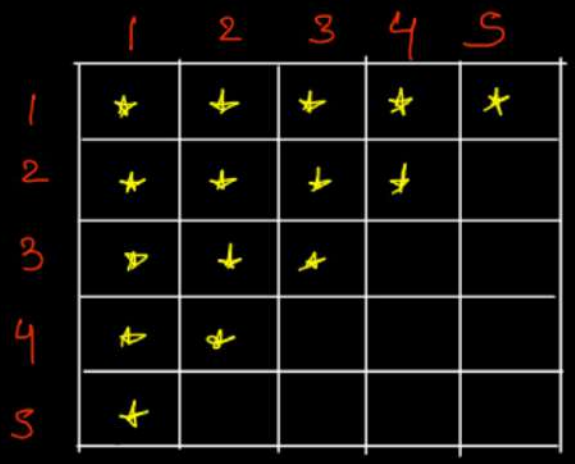
1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

A number n

Output Format

for N = 5



N to 1 → loop

↓

i^{th} iteration → i stars

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

    // write ur code here
    int n = scn.nextInt();

    for(int i=n;i>=1;i--) {

        //stars
        for(int j=1;j<=i;j++) {
            System.out.print("*\t");
        }

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

1 to N Loop With External Variable

1 to N → Loop

for N=5 Stars 5 4 3 2 1

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

    // write ur code here
    int n = scn.nextInt();

    int stars = n;
    for(int i=1;i<=n;i++) {

        //stars
        for(int j=1;j<=stars;j++) {
            System.out.print("*\t");
        }

        System.out.println();
        stars--;
    }
}
```

#1 to N Loop Without External Variable

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

    // write ur code here
    int n = scn.nextInt();

    for(int i=1;i<=n;i++) {

        //stars
        for(int j=1;j<=n-i+1;j++) {
            System.out.print("*\t");
        }

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

#Right Angled Triangle -2

Pattern 3

Easy

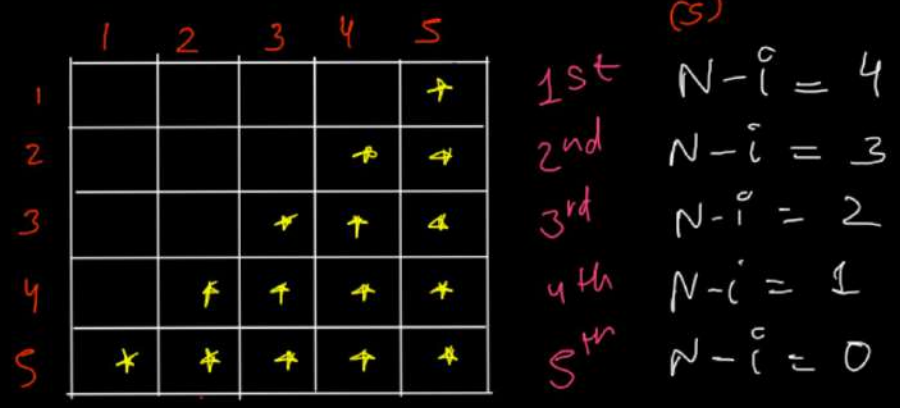
Prev Next

1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

A number n

Output Format




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

    // write ur code here
    int n = scn.nextInt();

    for(int i=1;i<=n;i++) {
        //space
        for(int sp=1;sp<=n-i;sp++) {
            System.out.print("\t");
        }

        //star
        for(int j=1;j<=i;j++) {
            System.out.print("*\t");
        }

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

Yet another Pattern

11 Difficulty: EASY

Avg. time to solve
15 min

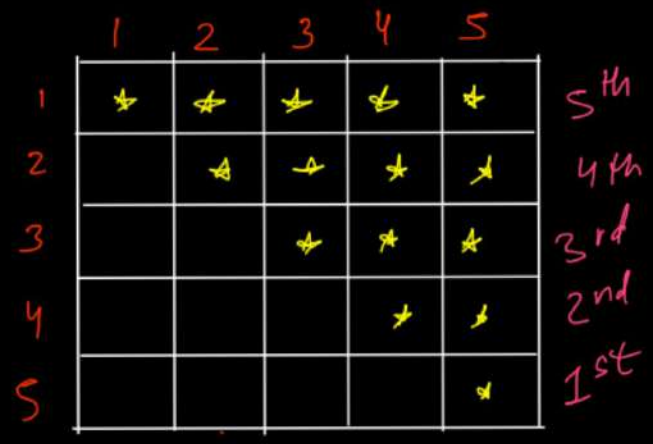
Success Rate
85%

Problem Statement Suggest Edit

Ninja was playing with her sister to solve a puzzle pattern. However, even after taking several hours, they could not solve the problem.

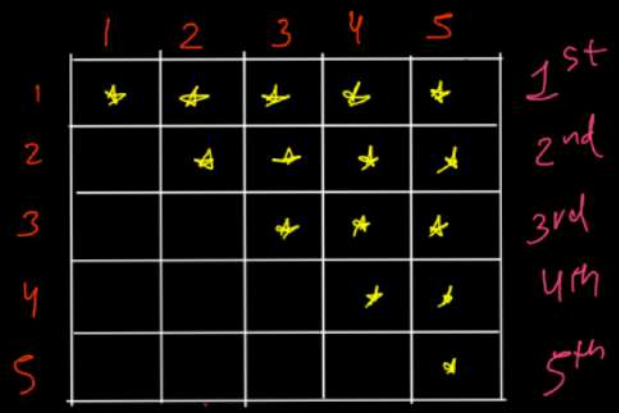
A value of N is given to them, and they are asked to solve the problem. Since they are stuck for a while, they ask you to solve the problem. Can you help solve this problem?

```
Example : Pattern for N = 4

****
 ***
  **
   *
```



N to 1 Loop with external variable for space

```java
// Write your code here.

int space = 0;

for(int i=n;i>=1;i--) {

    for(int sp=1;sp<=space;sp++) {
        System.out.print(" ");
    }

    for(int j=1;j<=i;j++) {
        System.out.print("*");
    }

    space++;
    System.out.println();
}
}
```



(N - i + 1) Stars

Spaces → i - 1

```java
// Write your code here.

for(int i=1;i<=n;i++) {
    //space

    for(int sp=1;sp<=i-1;sp++) {
        System.out.print(" ");
    }

    //stars

    for(int j=1;j<=n-i+1;j++) {
        System.out.print("*");
    }

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

Diamond of Stars

17

Difficulty: EASY

Contributed By
Ankush Gupta | Level 1

Avg. time to solve
10 min

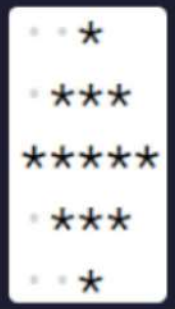
Success Rate
90%

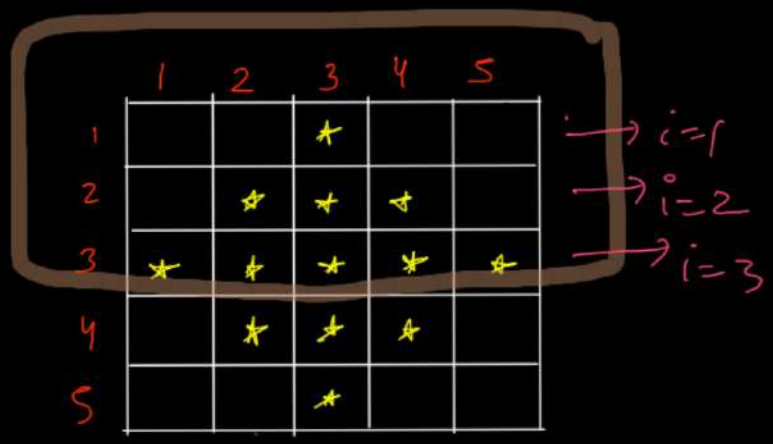
Problem Statement Suggest Edit

You are given an integer 'N'. Your task is to print the following pattern for the 'N' number of rows.

For Example:

```
Pattern for 'N' = 5:
```



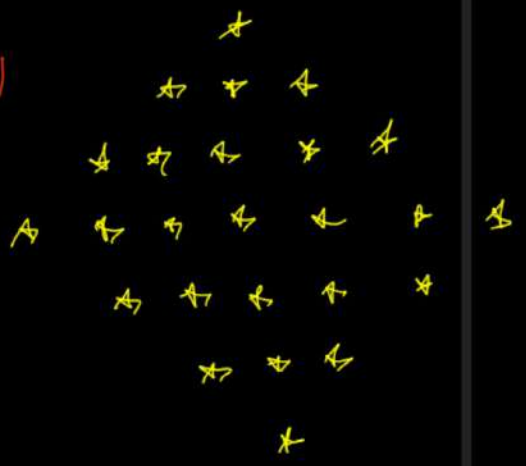


Upper half

let k = n/2 + 1; → 3

Spaces → k - i

Stars → 2i - 1



Second Half

Stars → start from N-2 &
keep decrementing by 2

Spaces = i

```java
int k = n/2 + 1;

//first half
for(int i=1;i<=k;i++) {
    //spaces
    for(int sp=1;sp<=k-i;sp++) {
        System.out.print(" ");
    }

    //stars
    for(int j=1;j<=2*i-1;j++) {
        System.out.print("*");
    }

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

```java
//second half
int stars = n-2;
for(int i=1;i<=n-k;i++) {

    //spaces
    for(int sp=1;sp<=i;sp++) {
        System.out.print(" ");
    }

    //stars

    for(int j=1;j<=stars;j++) {
        System.out.print("*");
    }

    System.out.println();
    stars -= 2;
}
```

Backward Slash

Pattern 7

Easy

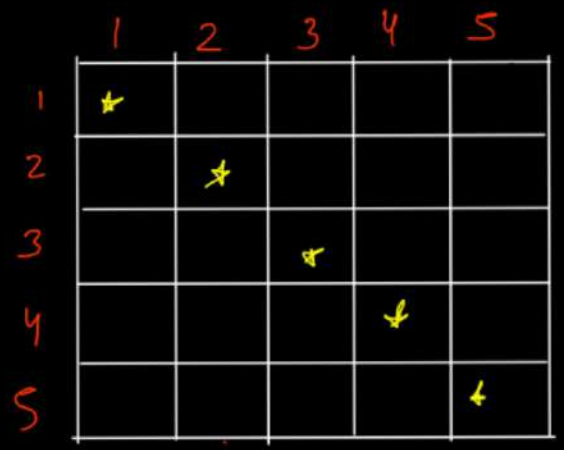
Prev Next

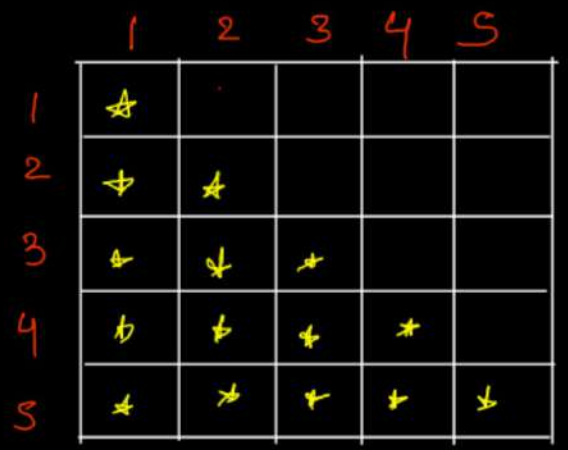
1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

A number n

Output Format





Compare them to simplify

1st
2nd
3rd
4th
5th

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

    // write ur code here

    int n = scn.nextInt();

    for(int i=1;i<=n;i++) {

        for(int j=1;j<=i;j++) {
            if(i == j) {
                System.out.print("*\t");
            } else {
                System.out.print("\t");
            }
        }

        System.out.println();
    }

}
```

forward Slash

Pattern 8

Easy

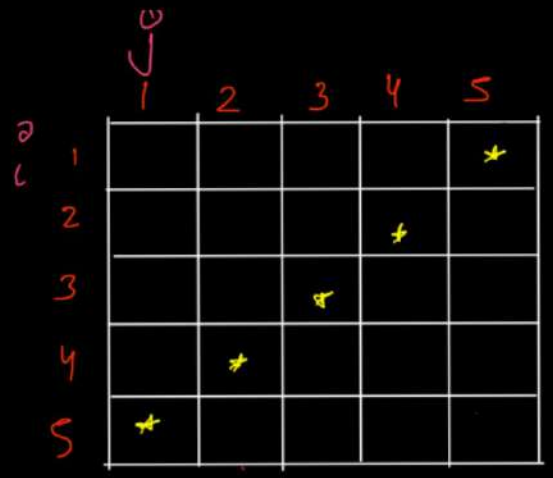
Prev Next

1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

A number n

Output Format



(i+j == N+1) → * break

else → Space

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

    // write ur code here
    int n = scn.nextInt();

    for(int i=1;i<=n;i++) {
        for(int j=1;j<=n;j++) {
            if(i+j == n+1) {
                System.out.print("*\t");
                break;
            } else {
                System.out.print("\t");
            }
        }
        System.out.println();
    }
}
```

Cross of Stars

Pattern 9

Easy

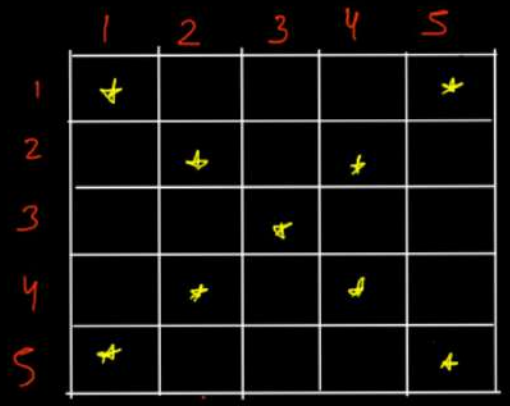
Prev Next

1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

A number n

Output Format




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

    // write ur code here
    int n = scn.nextInt();

    for(int i=1;i<=n;i++) {
        for(int j=1;j<=n;j++) {
            if(i == j || i+j == n+1) {
                System.out.print("*\t");
            } else {
                System.out.print("\t");
            }
        }

        System.out.println();
    }

}
```

Void of Diamond

Pattern 6

Easy

Prev Next

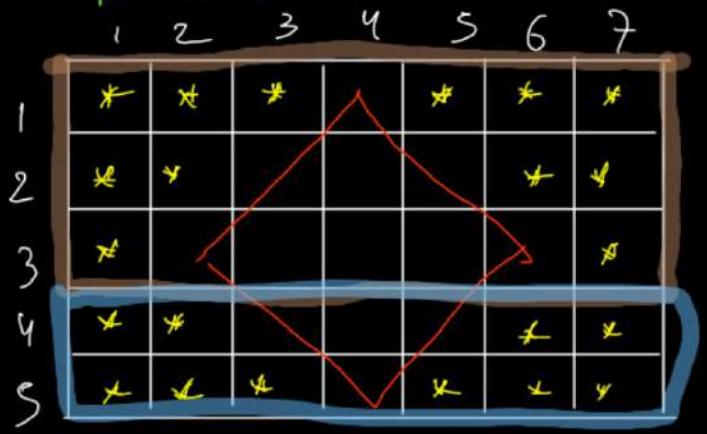
1. You are given a number n.
2. You've to create a pattern of * and separated by tab as shown in output format.

Input Format

A number n

Output Format

for n=5

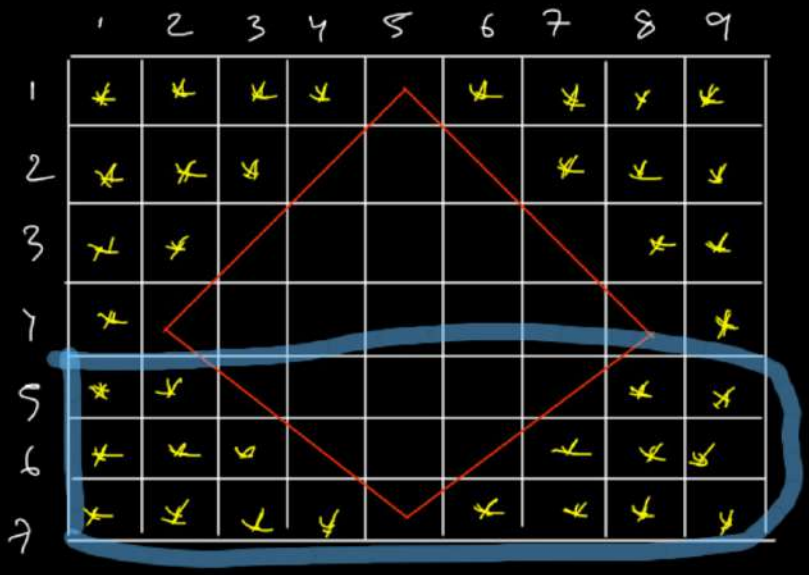


$k = n/2 + 1$

↓

3

for n=7



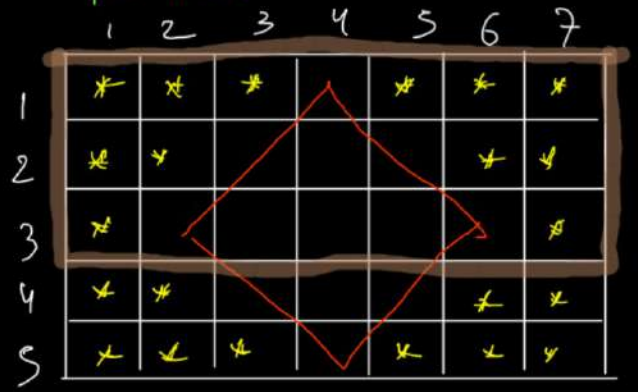
k = 4

First Half

→ Stars → Start from k

& dec by 1

for n=5



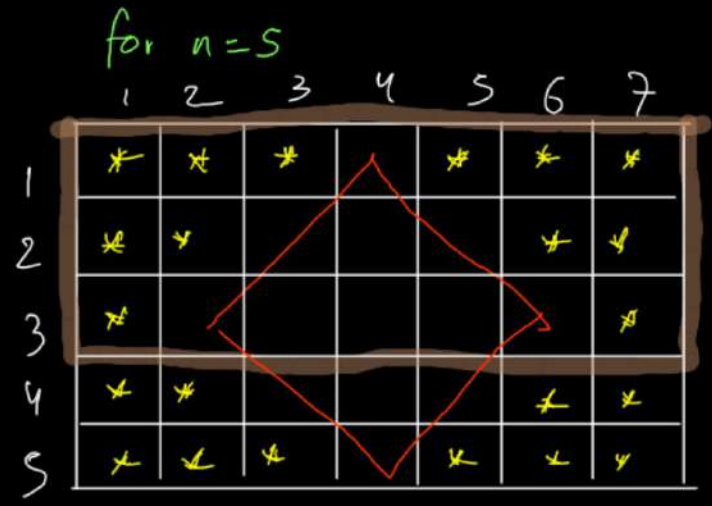
Second Half

Stars → start from 2

& inc by 1

Spaces → Start from n-2

& dec by 2



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

    // write ur code here
    int n = scn.nextInt();

    int k= n/2 + 1;

    int stars = k;
    for(int i=1;i<=k;i++) {

        for(int j=1;j<=stars;j++) {
            System.out.print("*\t");
        }

        //spaces

        for(int sp=1;sp<=2*i-1;sp++) {
            System.out.print("\t");
        }

        for(int j=1;j<=stars;j++) {
            System.out.print("*\t");
        }

        System.out.println();
        stars--;
    }

    stars = 2;
    int space = n-2;

    for(int i=1;i<=n-k;i++) {
        //stars
        for(int j=1;j<=stars;j++) {
            System.out.print("*\t");
        }

        for(int sp=1;sp<=space;sp++) {
            System.out.print("\t");
        }


        for(int j=1;j<=stars;j++) {
            System.out.print("*\t");
        }

        System.out.println();
        stars++;
        space -=2;
    }

}
```