

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

① Getting Started

② Arrays & Strings

- Two Pointer + Sliding Window
- Greedy Algorithms
- Searching & Sorting

Time & Space Complexity

* 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

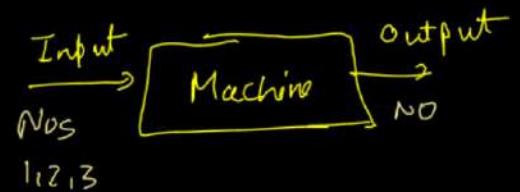
⑨ Graphs

* ⑩ Dynamic Programming

⑪ Number Theory & Bit Manipulation

⑫ Trie

Computer → Human convenience
↳ Machine



Communication

Computer understands Binary lang (0 and 1s)

Humans understand Human Lang → instructions
+ emotions

Prog Lang



① Computer → Software that converts

Prog Lang to Binary Lang

Input / Output -

```
1 // "static void main" must be defined in a public class.  
2 public class Main {  
3     public static void main(String[] args) {  
4  
5     }  
6 }
```

→ comments

```
1 import java.util.*;  
2 public class Main {  
3     public static void main(String[] args) {  
4         //this is a comment  
5     }  
6     //this is also a comment  
7 }
```

Print in Java

```
1 import java.util.*;  
2 public class Main {  
3     public static void main(String[] args) {  
4         System.out.print("DSA classes");  
5     }  
6 }
```

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         System.out.print("Guneet Malhotra");
5         System.out.print("DSA Classes");
6     }
7 }
```

Finished in 69 ms
Guneet MalhotraDSA Classes

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         System.out.print("Guneet Malhotra ");
5         System.out.print("DSA Classes");
6     }
7 }
```

Finished in 82 ms
Guneet Malhotra DSA Classes

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         System.out.print("Guneet Malhotra");
5         System.out.print(" DSA Classes");
6     }
7 }
```

Finished in 84 ms
Guneet Malhotra DSA Classes

System.out.print('Rohan');

R I Rohan
I

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         System.out.println("Guneet Malhotra");
5         System.out.print("DSA Classes");
6     }
7 }
```

Finished in 122 ms
Guneet Malhotra
DSA Classes

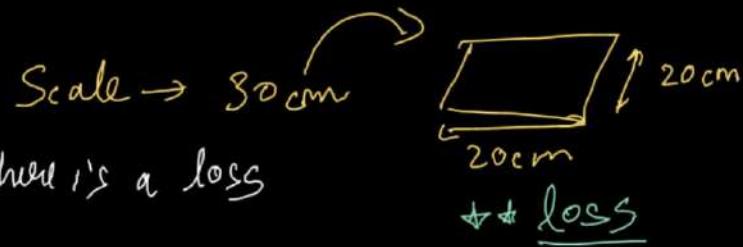
stdin

Data types in Java

Type of
containers.



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



Size in bytes

Integral (w/o decimal)	{	byte short int long	{	1 2 4 8	}	30, 50, 700 etc.
---------------------------	---	------------------------------	---	------------------	---	------------------

floating pt type (with decimal)	{	float double	{	4 8	3.141, 8.632 etc	
characters → char			2 → A-Z, a-z, 0-9, #, *, etc.			
True/false → boolean			Not fixed			

11/1

Range of Datatype

$$1 \text{ byte} = 8 \text{ bits}$$

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

Range of Short

$$\text{Size} = 2 \text{ bytes} = 2 \times 8 = 16 \text{ bits}$$

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

0000...0
16 bits

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

32768 → 17 bit

Data loss

```
1 * import java.util.*;
2 * public class Main {
3 *     public static void main(String[] args) {
4 *         int x = 100;
5 *         System.out.println(x);
6 *     }
7 * }
```

Finished in 62 ms
100

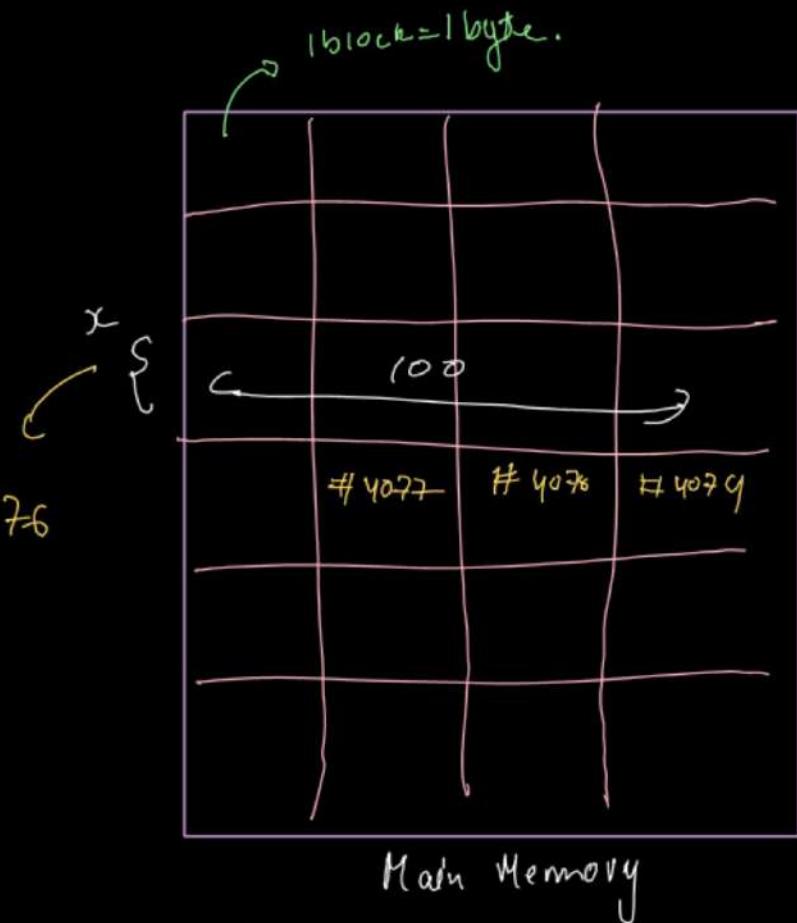
variable
int x = 100;
datatype constant

Binary Repres.
→ 32

```
int x = (100);
```

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         int x = 100; ①
5         System.out.println(x);
6     }
7 }
```

#4076



```
1+ import java.util.*;
2+ public class Main {
3+     public static void main(String[] args) {
4+         int x = 100;
5+         System.out.println(x);
6+
7+         char ch = 'a';
8+         System.out.println(ch);
9+     }
10 }
```

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

Operators in Java

```
1+ import java.util.*;
2+ public class Main {
3+     public static void main(String[] args) {
4+         int x = 100;
5+         System.out.println(x);
6+
7+         char ch = 'a';
8+         System.out.println(ch);
9+     }
10 }
```

$\text{int } x = 100;$

LHS

assignment operator

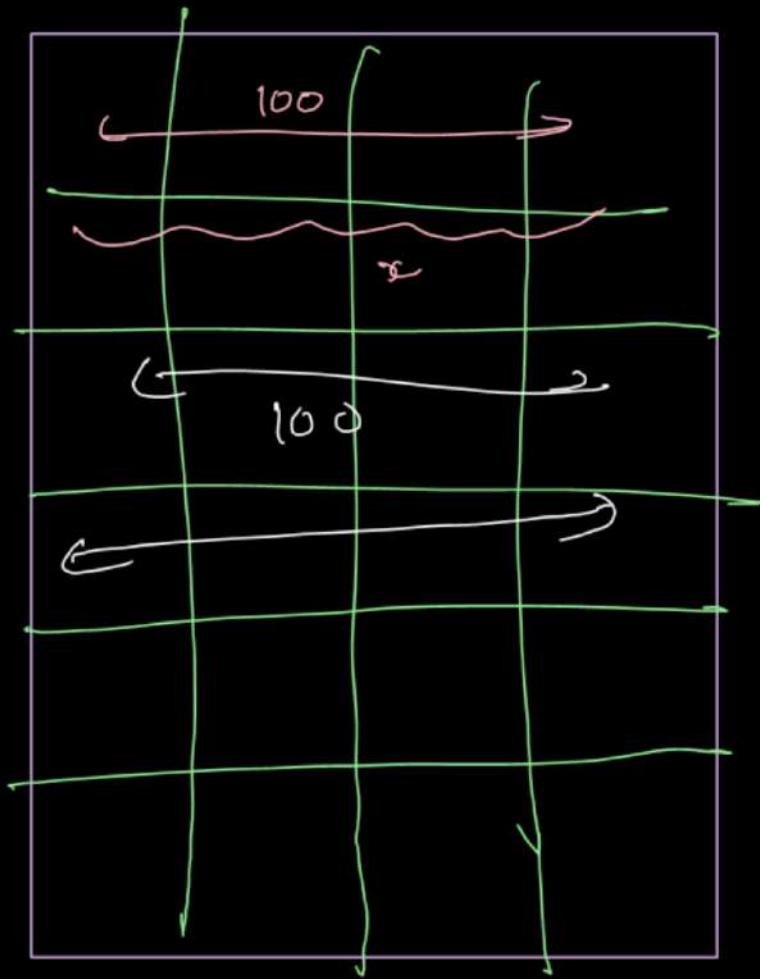
```
1+ import java.util.*;
2+ public class Main {
3+     public static void main(String[] args) {
4+         int x = 100;
5+         int y = x;
6+         System.out.println(x);
7+         System.out.println(y);
8+     }
9+ }
```

RHS gets assigned to LHS

$\text{int } y = x;$

$\text{int } y = \odot;$

y



```
1 * import java.util.*;
2 * public class Main {
3 *     public static void main(String[] args) {
4 *         int x;
5 *         x = 100;
6 *         System.out.println(x);
7 *     }
8 * }
```

Finished in 50 ms
100

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

* Initialization of values is a must in Java .

```
1 * import java.util.*;
2 * public class Main {
3 *     public static void main(String[] args) {
4 *         int x;
5 *         System.out.println(x);
6 *     }
7 * }
```

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

Arithmetic Operators

+

plus

-

minus

*

multiply

/

divide

%

modulus

; int x = 1.99

= 1

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         int x = 10;
5         int y = 30;
6         int res = x / y;
7         System.out.println(res);
8     }
9 }
```

$$10/30 = 113 = 0\overline{33} \dots$$

X

= ⑥

```

1 * import java.util.*;
2 * public class Main {
3 *     public static void main(String[] args) {
4 *         int x = 10;
5 *         int y = 30;
6 *         float res = (float)x / y; //typecasting
7 *         System.out.println(res);
8 *     }
9 * }

```

→ process of converting IDT to another.



integer operator → decimal truncates

x is converted to float 10 → 10.00

float/int = float

$10/30 \rightarrow$

$$30 \overline{)10}^0$$
 $\underline{-0}$

Quotient is the ans of integer / .

10 → remainder is the ans of modulus .

```

1 * import java.util.*;
2 * public class Main {
3 *     public static void main(String[] args) {
4 *         int x = 10;
5 *         int y = 30;
6 *         int res = x % y;
7 *         System.out.println(res);
8 *     }
9 * }

```

Finished in 70 ms

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         Scanner scn = new Scanner(System.in);
5         int num = scn.nextInt();
6         System.out.println(num);
7     }
8 }
```

Finished in 108 ms

256

stdin

256

Java code editor window showing the first Java program.

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         Scanner scn = new Scanner(System.in);
5         int num = scn.nextInt();
6         System.out.println(num);
7
8         float f = scn.nextFloat();
9         System.out.println(f);
10    }
11 }
```

Finished in 164 ms

256

10.987

Java code editor window showing the second Java program.

```
1+ import java.util.*;
2+ public class Main {
3+     public static void main(String[] args) {
4+         Scanner scn = new Scanner(System.in);
5+         int num = scn.nextInt();
6+         System.out.println(num);
7+
8+     }
9+ }
```

```
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

```
1+ import java.util.*;
2+ public class Main {
3+     public static void main(String[] args) {
4+         Scanner scn = new Scanner(System.in);
5+         char ch = scn.nextChar();
6+         System.out.println(ch);
7+     }
8+ }
```

```
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
```



NextChar

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

```
1 import java.util.*;
2 public class Main {
3
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in); → ① ✓
6         boolean b = scn.nextBoolean(); → ② ✓
7         System.out.println(b); → ③ ✓
8     }
9 }
```

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

If - Else

```
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+         if(guessNo == 10) { → true
10+             System.out.println("You have won the game");
11+         } else {
12+             System.out.println("You have lost the game");
13+         }
14+
15+         System.out.println("end");
16+     }
17+ }
```

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

guessNo = 10
LHS ↓ RHS
comparison
operator

if LHS = RHS
→ value of this
statement is true
else it is false

$\text{if}(\text{condition}) \{$

\equiv  true

$\}$  if condition is true then this executes

$\} \text{ else } \{$

$\equiv \}$  if condition is false, then this executes

$\}$

```
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 }
```

Greater than or equal to

Less than or equal to

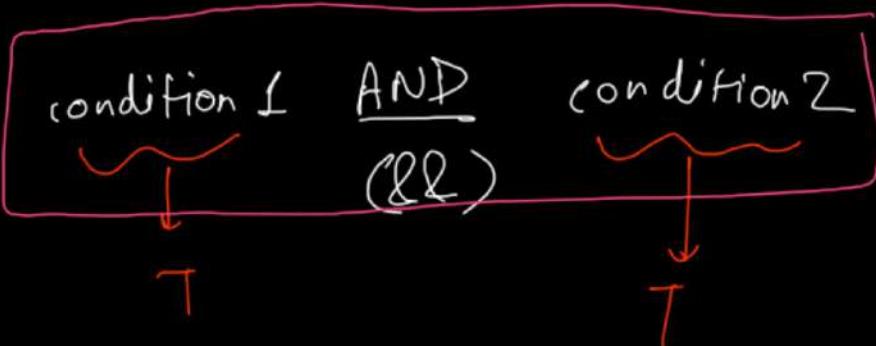
H W

> → Greater
than

< → Less than

logical AND

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) { // Logical OR
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 * }
```

cond 1 or cond 2



if min one is true
answer is true

Logical Not

```
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 }
```

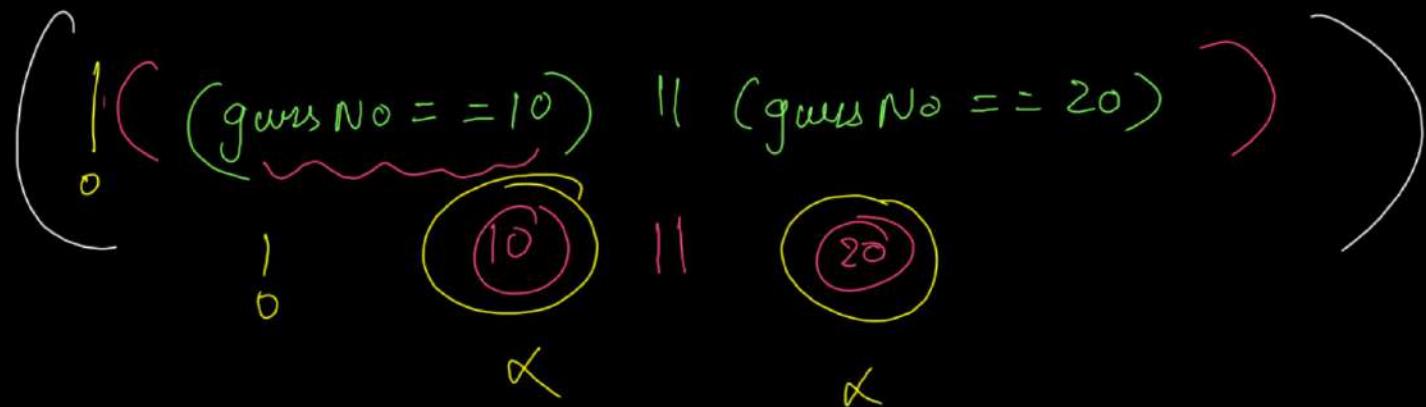
Logical not

! (condition)

↳ it negates
the condition

True → false

false → True



Not Equal to

```

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

```
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");
    }
}
```

if (condn1) {
 ≡
} else if (condn2) {
 ≡
 ≡
} else if (condn3) {
 true
 ≡
 ≡
 only this
 will execute
 :
 else {
 ≡

Multiple Else if

```
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) {

y \equiv

if ()

{

y

if (condn1) {

y \equiv

else if (condn2) {

}

if (condn3) {

y

✓

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

```
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 x = scn.nextInt();
7         int y = scn.nextInt();
8
9         System.out.println("Start");
10
11+     if(x == 10) {
12+         System.out.println("You have won the game");
13+     } else if(x == 20) {
14+         System.out.println("You have lost the game");
15+     }
16+     if(x == 100) {
17+         System.out.println("You have won the game");
18+     }
19
20         System.out.println("end");
21     }
22 }
```

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

for loop

Print your name 10 times

```
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
```

Generally used when exact no of iterations is known.

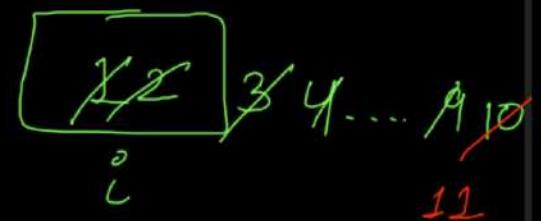
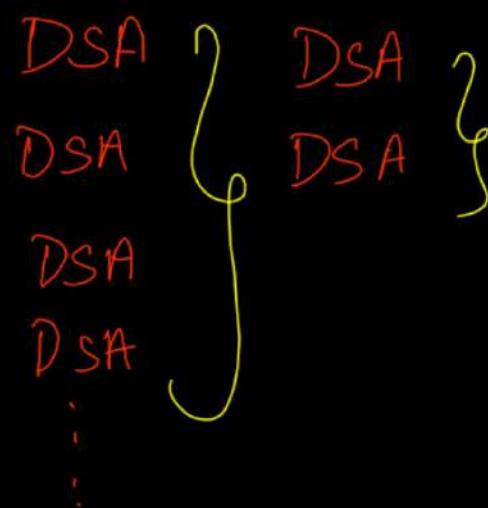
initial condition

for (int $i = 1$; $i \leq 10$; $i++$) {
 Sys("DSA"); ②
 }
 ① ③

condition

$$i++ \rightarrow i = i + 1$$

$(1) + 1$
 \uparrow
 (2)



Counting I to N

```
* 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

```
1 import java.util.*;
2 public class Main {
3
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6
7         int num = scn.nextInt();
8         int sum = 0; ① ③
9         for(int i=1;i<=num;i++) {
10             sum = sum + i; ②
11         }
12
13         System.out.println(sum);
14     }
15 }
```

12
l

0 1 2
Sum

$$\text{Sum} = \text{Sum} + i$$

$$\begin{aligned} & (1) + (2) \\ & = (2) \end{aligned}$$

```
1 import java.util.*;
2 public class Main {
3
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6
7         int num = scn.nextInt();
8         int sum = 0;
9         for(int i=1;i<=num;i++) {
10             sum += i; //sum = sum + i
11         }
12
13         System.out.println(sum);
14     }
15 }
```

While Loop

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

```
1 import java.util.*;
2 public class Main {
3
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6
7         int i = 1; ↳ Initialization
8         ① while(i <= 10){ ↳ condition
9             System.out.println("DSA Classes"); ↳ ②
10            i++; ③
11        }
12    } ↳
13 }
```

123...11
o
c

DSA

DSA

:

DSA

Count N to 1

```
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
        }
    }
}
```

$$n-- \rightarrow n = n - 1$$

10 9 8 7 6 5 4 3 2 1

~

10

9

8

7

6

5

4

3

2

1

Do while loop

```
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
    }
}
```

✓ ↗
i

DSA
DSA
DSA
DSA
DSA
DSA

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

$$11 \leq 10 \rightarrow \text{false}$$

$\cancel{\text{yx}}$ loop not executed

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

$$\boxed{\text{DSA}} \\ i = \cancel{11} \ 12$$

$$12 \leq 10 \ \cancel{x}$$

Continue in Loops

```
1 import java.util.*;
2 public class Main {
3
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         for(int i=1;i<=10;i++) {
7             if(i % 3 == 0) {
8                 continue;
9             }
10            System.out.println(i);
11        }
12    }
13 }
14 }
```

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

12 3 4

1
2
4

$i \% 3 == 0$

Skips the current iteration

Break in loops

```
1 * import java.util.*;
2 * public class Main {
3 *
4 *     public static void main(String[] args) {
5 *         Scanner scn = new Scanner(System.in);
6 *
7 *         for(int i=1;i<=10;i++) {
8 *             if(i % 3 == 0) {
9 *                 break;
10 *             }
11 *             System.out.println(i);
12 *         }
13 *     }
14 * }
```

→ break from the loop

i.e exit the loop.

Switch Case

```
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 num = scn.nextInt();
7 *         switch(num){ → Or character or String
8 *             case 10:
9 *                 System.out.println("Option 10");
10 *                 break;
11 *             case 20:
12 *                 System.out.println("Option 20");
13 *                 break;
14 *             case 30:
15 *                 System.out.println("Option 30");
16 *                 break;
17 *             default:
18 *                 System.out.println("Please select out of 10, 20 or 30");
19 *         }
20 *     }
21 *
```

Getting Started - Lecture 3

- ① Unary Operators
- ② functions and their Memory Mappings

DSA Sheet Questions

- ① Even or Odd - 1 ④ Sum or Product
- ② Even or odd - 2 ⑤ Sum of Even Nos Till N
- ③ factorial

Unary Operators

Operand 1 operator Operand 2
+ , - , * , / , % , && , || etc

operator Operand

uv

Operand operator

+
-
++ ↗ pre
 ↘ post
-- ↗ pre
 ↘ post

+ (Unary Operator)

```
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 x = +3 + 5;
7         System.out.println(x);
8     }
9 }
```

Unary Binary

- (Unary Operator)

```
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 x = -3 + 5;
7         System.out.println(x);
8     }
9 }
```

Unary Binary

$++$ (Increment Operator)

Post

$$\underbrace{x++}_{\text{over}} ; \rightarrow x = x + 1$$

Pre

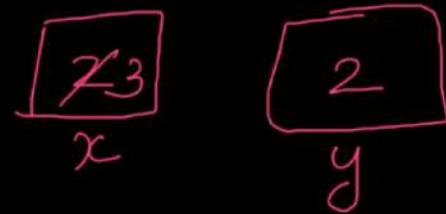
$$++x ; \rightarrow x = x + 1$$

$y = x++ \rightarrow$ True $\leftarrow y = ++x$
2 statements are different.

Post Increment

```
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);
    }
}
```



$y = x++$

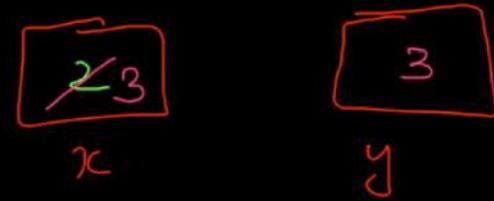
assignment increment

$$\textcircled{1} \quad y = x$$

$$\textcircled{2} \quad x = x + 1$$

Pre Increment

```
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 x = 2;
7         int y = ++x;
8         System.out.println(x);
9         System.out.println(y);
0     }
1 }
```



$$y = ++x ;$$

① $x = x + 1$

② $y = \textcircled{x}$

Post Decrement

```
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);
    }
}
```

2
1

2
y

$$y = x - - ;$$

① $y = x$

② $x = x - 1$

$$x - - ;$$

$\hookrightarrow x = x - 1 ;$

Pre decrement

```
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 x = 2;
7         int y = --x;
8         System.out.println(x);
9         System.out.println(y);
10    }
11 }
```

21

20

1

y

$$y = --x /$$

① $x = x - 1$

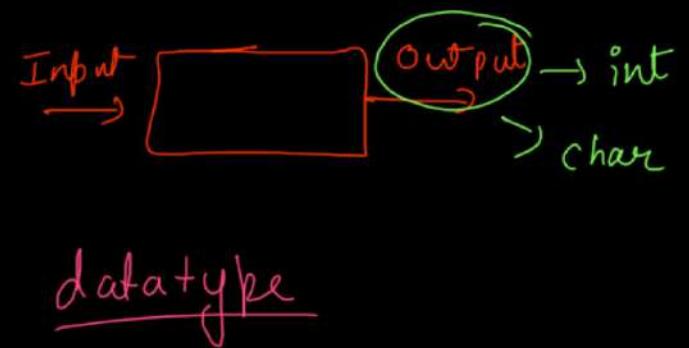
② $y = x$

functions

```
import java.util.*;  
public class Main {  
    public static void counting() {  
        //code  
    }  
    public static void main(String[] args) {  
        //some code  
    }  
}
```

Return type

If void means function does not return anything .



```
1 * import java.util.*;
2 * public class Main {
3
4     public static void counting(int n) {parameter
5         //code
6     }
7
8     public static void main(String[] args) {
9         //some code
10    }
11 }
```

```
* import java.util.*;
* public class Main {
*
*     public static void counting(int n) {function body
*         //code
*         for(int i=1;i<=n;i++) {
*             System.out.println(i);
*         }
*     }
*
*     public static void main(String[] args) {
*         //function call
*         counting(10); → function call
*     }
* }
```

```
1+ import java.util.*;
2+ public class Main {
3
4+     public static void counting(int n) {
5         //code
6+         for(int i=1;i<=n;i++) {
7             System.out.println(i);
8         }
9     }
10
11+    public static void main(String[] args) {
12        //function call
13        counting(10);
14        System.out.println();
15        counting(20);
16    }
17 }
```

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

```

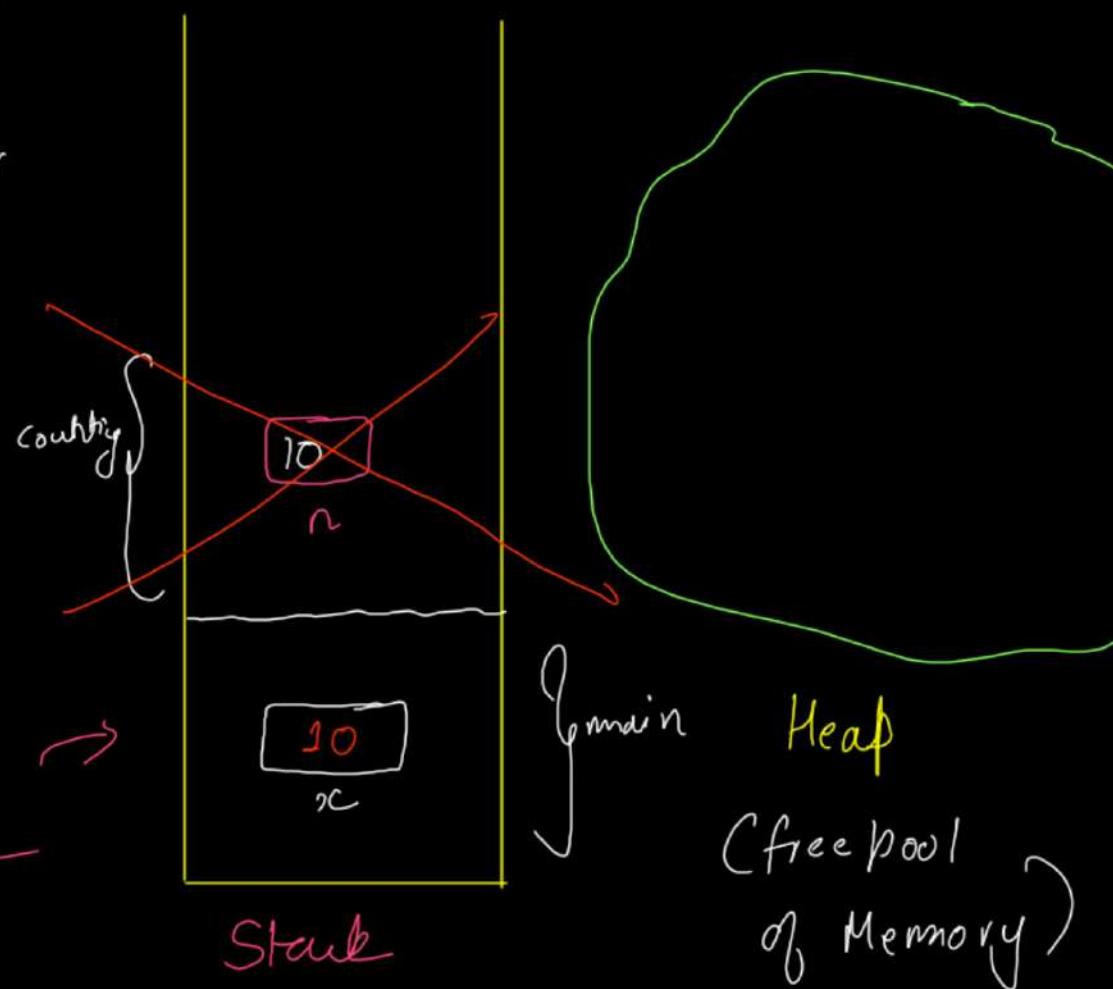
import java.util.*;
public class Main {
    > function definition
    public static void counting(int n) {
        //code
        for(int i=1;i<=n;i++) {
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        //function call
        int x = 10; → ①
        counting(x); → ② Pause
        System.out.println(); → ③
    }
} } argument

```

1
2
3
4
5
6
7
8
9
10

function call Stack

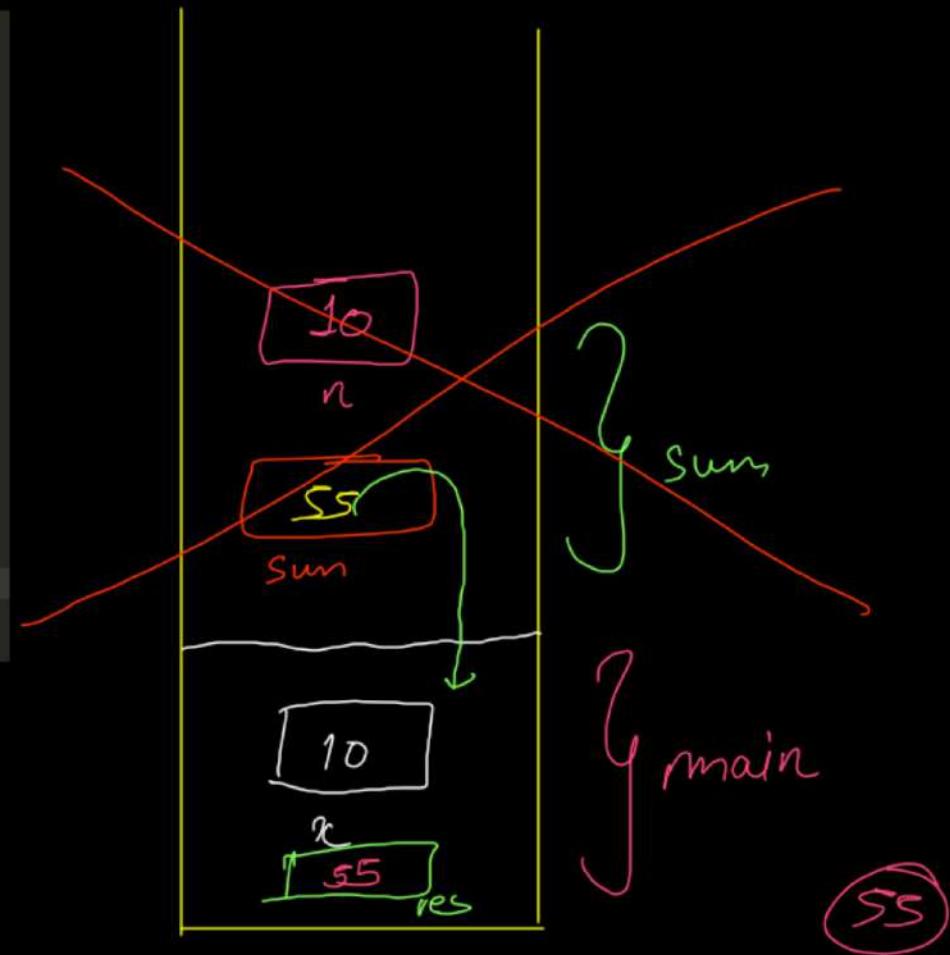


-
- ① All primitive datatypes (8) are allocated memory inside the Stack.
 - ② As soon as all the lines of code of a function are executed, it gets wiped out of the stack.
 - ③ In Java, functions are called Methods because in OOPS any function inside a class is called a Method & in Java everything is inside a class.

```

1+ import java.util.*;
2+ public class Main {
3
4+     public static int sum(int n) {
5         //function body
6         int sum = 0;
7         for(int i=1;i<=n;i++) {
8             sum += i;
9         }
10        return sum; ✓ ✓
11    }
12
13+    public static void main(String[] args) {
14        //function call
15        int x = 10; → ①
16        int res = sum(10); → ② Pause
17        System.out.println(res); → ③
18    }
19 }

```



```
1 * import java.util.*;
2 * public class Main {
3 *
4 *     public static int sum(int n) {
5 *         //function body
6 *         int sum = 0;
7 *         for(int i=1;i<=n;i++) {
8 *             sum += i;
9 *         }
10 *         return sum;
11 *     }
12 *
13 *     public static void main(String[] args) {
14 *         //function call
15 *         int x = 10;
16 *         System.out.println(sum(10));
17 *     }
18 * }
```

(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.

```
//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

$$N \% 2 \neq 0 \quad (\in 1)$$

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.

System.out.print/
ln("↓");

This is a

String

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

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:

$5 \times 4 \times 3 \times 2 \times 1 = 120$

```
class Solution{
    static long factorial(int n){
        // code here
        if(n == 0) return 1L;
        long product = 1L; ②
        for(int i=n;i>=1;i--) {
            product = product * i; ③
        }
        return product;
    }
}
```

factorial

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

$$= 1 \times 2 \times 3 \times \dots \times (N-1) \times (N-2)$$

54
i

13 20 68
product
120

$$P = P \times i$$

(13)(4)



Problem

Submissions

Solution

New Discuss



Sum or Product

1006 Difficulty: EASY

Contributed By
Ankush Gupta | Level 1

Avg. time to solve

15 min



Success Rate

80%

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

$$N(N+1)/2$$

$$10^9 + 7 = (10000000007)$$

$$\begin{array}{r} \overline{1} \\ \times 7 \\ \hline \overline{0} \end{array}$$

$$\begin{array}{r} \overline{8} \\ \times 7 \\ \hline \overline{1} \end{array}$$

$$\begin{array}{r} \checkmark \\ 0 \rightarrow 10^9 + 6 \end{array}$$

$$\textcircled{1} \cdot 7 = \textcircled{1}$$

$$\textcircled{2} \cdot 7 = \textcircled{2}$$

$$\textcircled{3} \cdot 7 = \textcircled{3}$$

$$\textcircled{4} \cdot 7 = \textcircled{4}$$

$$\textcircled{5} \cdot 7 = \textcircled{5}$$

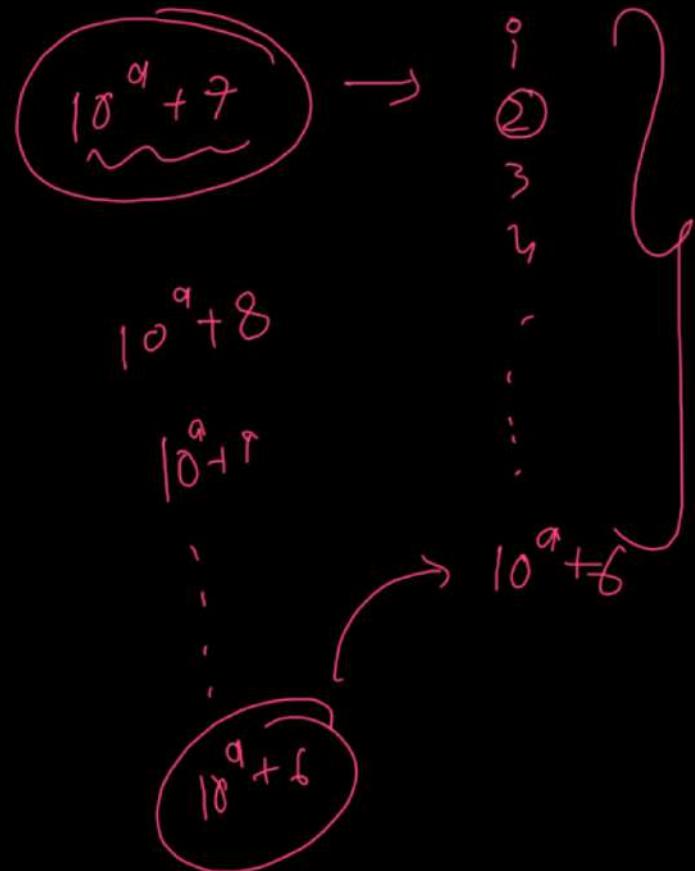
$$\textcircled{6} \cdot 7 = \textcircled{6}$$

$$7 \cdot 1 \cdot 7 = 0$$

$$8 \cdot 1 \cdot 7 = 1$$

$$9 \cdot 1 \cdot 7 = 2$$

⋮



INT \times INT
 \downarrow \downarrow
 (∞) $(\infty - 1)$

$N \times (\underline{N} - 1) \dots 1$

of
 range

```
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;
    }
}
```

```
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 ~~++~~

→ Switch Case

→ # Java is always
pass by value

Basic Maths

→ Check Prime

$$\begin{array}{c} \nearrow N \\ \searrow N^{1/2} \\ \swarrow \sqrt{N} \end{array}$$

→ Prime Nos in Range

→ GCD & LCM

variable

→ Nth fibonaci No

→ Bulb Switcher

→ Power of 2

Normal Loop

Euclidean

Sum Of Even Numbers Till N

71

Difficulty: EASY



Contributed By
Ayush Thakur | Level 1

Avg. time to solve

10 min

Success Rate

90%



10

1 2 3 4 5 6 7 8 9 10

$$2 + 4 + 6 + 8 + 10 \rightarrow \text{ans}$$

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 : $2 + 4 + 6 = 12$

if (condn) \sum

$\{$

```
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.

Before Swapping

2

a

3

b

After Swapping

3

a

2

b

Using 3rd variable

23

a

22

b

2

temp

① int temp = a;

② a = b;

③ b = temp;

Without 3rd Variable

$$\begin{array}{c} \boxed{x_1} \\ \diagup \\ a \end{array} \quad \begin{array}{c} \boxed{x_2} \\ \diagdown \\ b \end{array}$$

$$\textcircled{1} \quad a = a + b \\ (1) + (2) = (3)$$

$$\textcircled{2} \quad b = a - b \\ (1) - (2) = (3)$$

$$\textcircled{3} \quad a = a - b \\ (1) - (2) = (3)$$

```

import java.util.*;
public class Main {

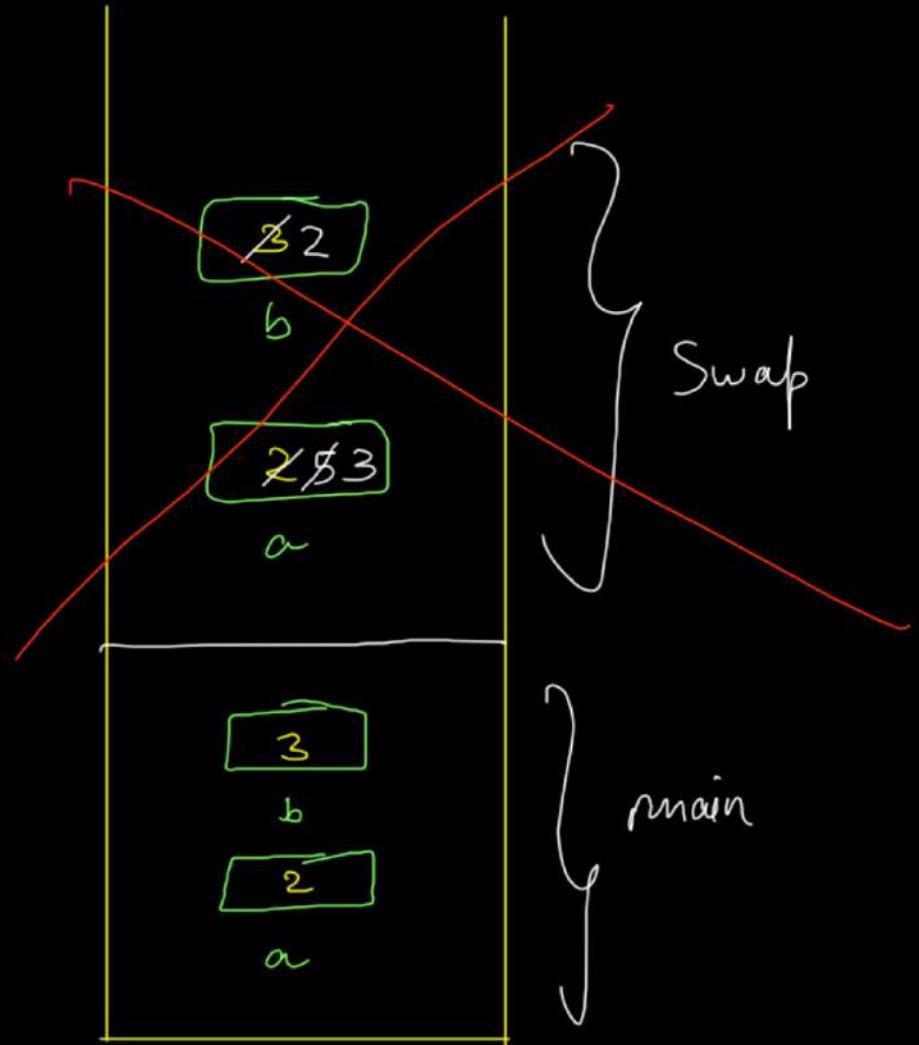
    public static void swap(int a, int b) {
        //swapping
        a = a + b; ✓
        b = a - b; ✓
        a = a - b; ✓
    }

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in); ①
        int a = scn.nextInt(); ②
        int b = scn.nextInt(); ③

        System.out.println("Before swapping a = " + a + " b = " + b); ④
        swap(a,b); ⑤ Pause
        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.)



```
+ 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).

```
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

```
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);
    }
}
```

(condition) ? if true : if false

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

;

1 2 3 4 N-1 N

↓
loop & check

(24) 1, 2 4

(29) 1, 2 9

```
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 N12 24 → ⑫ equal to N12

$$1 \times 27 = 27$$

$$1 \times 24 = 24$$

$$3 \times 9 = 27$$

$$2 \times 12 = 24$$

$$3 \times 8 = 24$$

$$4 \times 6 = 24$$

①, 3, ⑨, ⑪, 27

①, 2, 3, 4, 6, 8, ⑫, 24

$$\textcircled{24} / \textcircled{2} = 12$$

smallest * largest \rightarrow $n/2$ is the largest factor in case of even Nos

(2) (12)

25

smallest * largest

```
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;
    }
}
```

Y

iterations have
reduced to half.

O(N)

Square Root Solution

(36) → (6)

$$\left\{ \begin{array}{l} 1 \times 36 = 36 \\ 2 \times 18 = 36 \\ 3 \times 12 = 36 \\ 4 \times 9 = 36 \\ 6 \times 6 = 36 \end{array} \right.$$

(24) → (4)

$$\left\{ \begin{array}{l} 1 \times 24 = 24 \\ 2 \times 12 = 24 \\ 3 \times 8 = 24 \\ 4 \times 6 = 24 \end{array} \right.$$

(25)

$$\left\{ \begin{array}{l} 1 \times 25 = 25 \\ 5 \times 5 = 25 \end{array} \right.$$

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

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

② 29 → ⑤

$$1 \times 29 = 29$$

① 29

\sqrt{n}

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

Code conversion

$$i \leq \sqrt{N}$$

Squaring both sides

$$i^2 \leq N$$

$$\Rightarrow i + i \leq N$$

(625)

2 to 624 1st Method

2 to 317 2nd Method

2 to 25 3rd Method

```
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;
    }
}
```

$\# O(\sqrt{N})$

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.

```
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;
}
```

```
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.

1 ✓	15 ✗ ✗	16 ✗ ✗ ✓	17 ✗
2 ✗	16 ✗ ✗ ✓	17 ✗	18 ✗ ✗ ✗
3 ✗	17 ✗	19 ✗	20 ✗ ✗ ✗
4 ✗ ✓	18 ✗ ✗ ✗	21 ✗ ✗	22 ✗ ✗
5 ✗	19 ✗	22 ✗ ✗	23 ✗
6 ✗ ✗	20 ✗ ✗ ✗	24 ✗ ✗ ✗ ✗	25 ✗ ✗
7 ✗	21 ✗ ✗	25 ✗ ✗	
8 ✗ ✗	22 ✗ ✗		
9 ✗ ✓	23 ✗		
10 ✗ ✗	24 ✗ ✗ ✗ ✗		
11 ✗	25 ✗ ✗		
12 ✗ ✗ ✗			
13 ✗			
14 ✗ ✗			

✓ → ON

✗ → off

24

$$1 \times 24 = 24$$

$$2 \times 12 = 24$$

$$3 \times 8 = 24$$

$$4 \times 6 = 24$$

$$6 \times 4 = 24$$

$$8 \times 3 = 24$$

$$12 \times 2 = 24$$

$$24 \times 1 = 24$$

36

$$1 \times 36 = 36$$

$$2 \times 18 = 36$$

$$3 \times 12 = 36$$

$$4 \times 9 = 36$$

$$6 \times 6 = 36$$

$$9 \times 4 = 36$$

$$12 \times 3 = 36$$

$$18 \times 2 = 36$$

$$36 \times 1 = 36$$

This does not mirror.

#Bulb Switcher

```
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

```
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 n th 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;

5th

```
// 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;
    }
}
```

$$a = \cancel{0} \cancel{1} \cancel{2} \cancel{3}$$

$$b = \cancel{1} \cancel{2} \cancel{3} \cancel{5}$$

$$c = \cancel{0} \cancel{1} \cancel{2} \cancel{3} \cancel{5}$$

$$c = \cancel{2} \cancel{3} \cancel{4} \cancel{5} \quad \textcircled{S}$$

$$c = a + b$$

$$2^{+3} - \textcircled{S}$$

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.

$$A * B = \text{LCM} * \text{GCD}$$

$$\text{LCM} = (A * B) / \text{GCD}$$

24, 36 ✓
28
32
24
:
⑫ → HCF

```
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

```
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

2^4 and 3^6

$$\begin{array}{r} b \\ \overline{)2^4} \\ 36 \\ -2^4 \\ \hline 12 \end{array}$$

$a = b$
 $b = \text{rem}$

$$\begin{array}{r} 2 \\ \overline{)2^4} \\ 24 \\ -2^4 \\ \hline 0 \end{array}$$

$b \rightarrow \text{lcm}$

$$\begin{array}{r} 36 \\ \overline{)2^4} \\ = 0 \\ 24 \\ \overline{)36} \end{array}$$

```
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 (Condition 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

Reverse A Number

Easy

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

1 1 1 1 1 1 1
6 5 7 8 4 3 8 3

3

8

3

4

8

7

5

6

$65784383 \rightarrow n$

$\text{int digit} = n \% 10;$

$n = n / 10;$

$$\begin{array}{r} 6578438 \\ 10 \overline{)65784383} \\ -60 \\ \hline 57 \\ -50 \\ \hline 78 \\ -70 \\ \hline 84 \\ -80 \\ \hline 43 \\ -40 \\ \hline 38 \\ -30 \\ \hline 8 \end{array}$$

$$83 \mid -80 = \underline{\underline{3}}$$

	digit ($N \% 10$)	$N / 10$
6578438③	3	6578438
6578438	8	657843
65784③	3	65784
6578④	4	6578
657⑧	8	657
6⑦	7	65
6⑤	5	6

6 6 0

0 → end point

$6 /_{10}$
~~-⑥~~

$N = \textcircled{0} \rightarrow 0$ # corner case

```
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;
    }
}
```

} $\longrightarrow O(\log_{10} N)$

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	$N / 10$	$\text{original } N (124)$
124	4	12	count = 1
12	2	1	count = 2
1	1	0	count = 3

9 8 0 2 3 4

→ 9 8 2 3 4 % 0



Arithmetic Exception

↳ / by zero exception

```
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;
    }
}
```

7

→ $O(\log_2 N)$

Rotate A Number



◀ 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

$n = 12340056$

$k = 3$

$r = 05612340$

5 6 2 9 8 4 → N → nod = ⑥

→ k (No of rotations)

k % nod ?

(0) k = 0 5 6 2 9 8 4 ← k = 6 5 6 2 9 8 4 (0)

(1) k = 1 4 5 6 2 9 8 ← k = 7 4 5 6 2 9 8 (1)

k = 2 8 4 5 6 2 9

k = 8

k = 3 9 8 4 5 6 2

k = 9

k = 4 2 9 8 4 5 6

k = 10

k = 5 6 2 9 8 4 5

k = 11

$$6 \overline{)1000}$$

$$\begin{array}{r} -6 \\ \hline 40 \end{array}$$

$$\begin{array}{r} -30 \\ \hline 10 \end{array}$$

✓

① $k = k \text{ mod } n$ (To convert large values of
 k to small values)

(0 to Nod-1)

$$\text{Nod} = 6$$

#② if ($k < 0$)

$$k = 0 \quad 562984$$

$$k = k + \text{nod}$$

$$k = 1 \quad 456298 \rightarrow k = -5 \quad 456298$$

$$k = 2 \quad 845629 \rightarrow k = -4 \quad 845629$$

$$k = 3 \quad 984562 \rightarrow k = -3 \quad 984562$$

$$k = 4 \quad 298456 \rightarrow k = -2 \quad 298456$$

$$k = 5 \quad 629845 \rightarrow k = -1 \quad 629845$$

$$N = 562984$$

$$k = 2$$

$$\text{Last} = N \% 10^k \quad (84)$$

$$N = N / 10^k \quad (\underline{\underline{5629}}) -$$

$$\text{Last} = \text{Last} * 10^{\text{nod} - k} \quad (\underline{\underline{840000}})$$

$$N = \text{Last} + N(845629)$$

$$\begin{array}{r} 845629 \\ 100 \sqrt{562984} \\ \underline{-500} \quad | \\ \quad 629 \\ \underline{-600} \quad | \\ \quad 298 \\ \underline{-200} \quad | \\ \quad 984 \\ \underline{-900} \quad | \\ \quad 84 \end{array}$$

```
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);
}
```

-10000k

$k = k \text{ \% } nod$

$-(\underbrace{1000}_{\text{\%}} \text{ \% } 6)$

$k = -4$

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:

$$\text{Product of digits} = 2 * 3 * 4 = 24$$

$$\text{Sum of digits} = 2 + 3 + 4 = 9$$

$$\text{Result} = 24 - 9 = 15$$

$$No = 234$$

N	digits	Sum (0)	Product (1)
234	4	4	4
23	3	7	12
2	2	9	24
0			$24 - 9 = 15$

```

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 $3^3 + 7^3 + 1^3 = 371$

$$on = 1$$

$$n = 371 \quad \text{ans}$$

$$N \quad \text{digit} \quad \text{ans} \quad (\text{ans} = d^3 + d^3 + d^3)$$

3	7	1
---	---	---

$$3^3 + 7^3 + 1^3 = 343 + 343 + 1 = 371$$

$$\textcircled{3} \quad 3 \quad 343 + (3^3 + 3^3 + 3^3) \\ = 371$$

```

//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

```

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;
    }
}

```

$rev * 10 < Integer.MIN_VALUE$

$rev < Integer.MIN_VALUE$

10

Integer
1

$\infty * 10 \rightarrow \text{out of Range}$

$-\infty * 10 \rightarrow \text{out of Range}$

$rev * 10 > Integer.MAX_VALUE$
value

$rev > Integer.MAX_VALUE$

10

```

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.

 A handwritten diagram of the number 121. Above the number, there are two arrows: one pointing from the left '1' to the right '1', and another pointing from the right '1' back to the left '1'. This indicates that the number is a palindrome.

 A handwritten diagram of the number 123. There are two vertical arrows: one pointing down from the top '1' to the bottom '3', and another pointing up from the bottom '3' to the top '1'. This indicates that the number is not a palindrome.

```
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 its inverse.

Input Format

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

$N \rightarrow 1 \text{ to } N \text{ digits}$

8 7 6 5 4 3 2 1
 ↓
 2834675(1)
 ↓
 place value
 ↓
 face value

8 7 6 5 4 3 2 1
 ↓
 73425681 → Inverse

↓ Number

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

4 3 2 1
 ↓
 1278

$\left\{ \sum p_v * 10^{f_v-1} \right\} \rightarrow \text{Inverse}$

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

$$pv = 78$$
$$iv = \boxed{73425681}$$
$$fv = 4582$$

$$8 * 10^1$$

$$\approx 80$$

```
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);
    }
}
```

Concepts ↓

Getting Started - Lecture 6 (Pattern Printing)

- 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

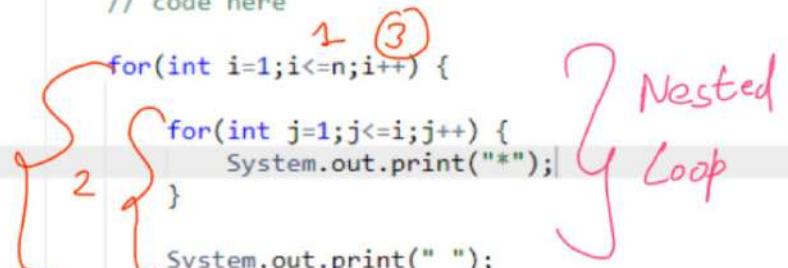
for $N=4$

*	*	*	*	*	*	*	*	*	*	*	*	*
---	---	---	---	---	---	---	---	---	---	---	---	---

1 2 3 4

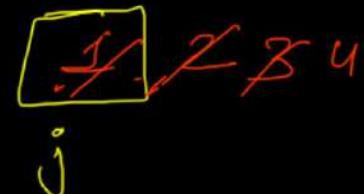
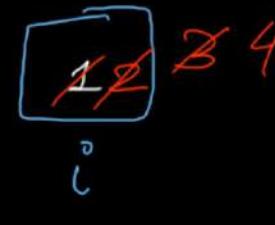
↳ i^{th} iteration → $i\text{stars}$

```
//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(" ");
        }
    }
}
```



→ 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 $N=5$

	1	2	3	4	5
1	*				
2	*	*			
3	*	*	*		
4	*	*	*	*	
5	*	*	*	*	*

for $N=3$

*
* *
* * *

for $N=6$

*
* *
* * *
* * * *
* * * * *
* * * * * *

for $N=3$

*		*	*		*	*	*
---	--	---	---	--	---	---	---

```
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 $N = 5$

$N \rightarrow 1 \rightarrow \text{Loop}$



$i^{\text{th}} \text{ iteration} \rightarrow i \text{ stars}$

	1	2	3	4	5
1	*	*	*	*	*
2	*	*	*	*	
3	*	*	*		
4	*	*			
5	*				

```
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();  
    }  
}
```

I to N Loop with External Variable

1 to N → Loop

for, N=5

stars

~~S~~

A

B

C

O

```
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++) {  
            vol     yet  
            5th 6th  
        //stars  
        for(int j=1;j<=stars;j++) {  
            System.out.print("*\t");  
        }  
        System.out.println();  
        stars--;  
    }  
}
```

* * * * *
* * * *
* * *
* *
*

#1 to N loop without External Variable

```
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

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

	1	2	3	4	5
1					*
2			*	*	*
3		*	*	*	*
4	*	*	*	*	*
5	*	*	*	*	*

(5)

1st $N-i = 4$
 2nd $N-i = 3$
 3rd $N-i = 2$
 4th $N-i = 1$
 5th $N-i = 0$

```
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

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

	1	2	3	4	5
1	*	*	*	*	*
2		*	*	*	*
3			*	*	*
4				*	*
5					*

5th
4th
3rd
2nd
1st

// 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();
}

```

1
2
3
4
5

N to 1 Loop with external variables
for space

	1	2	3	4	5
1	*	*	*	*	*
2		*	*	*	*
3			*	*	
4				*	*
5					*

$(N - i + 1)$ Stars

1st
2nd
3rd
4th
5th

Spaces $\rightarrow i - 1$

```
// 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:

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

	1	2	3	4	5
1		*			
2	*	*	*		
3	*	*	*	*	*
4	*	*	*		
5		*			

Upper half

$$\begin{array}{l} \rightarrow i=1 \\ \rightarrow i=2 \\ \rightarrow i=3 \end{array}$$

$$\text{let } k = n/2 + 1; \rightarrow 3$$

$$\text{Spaces} \rightarrow k - i$$

$$\text{Stars} \rightarrow 2^i - 1$$

*
 * * *
 * * * * *
 * * * * * *
 * * * * * * *
 * * * * * * * *
 *

Second Half

Stars → start from $N-2$

keep decrementing by 2

$$\text{Spaces} = i$$

```
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();
}
```

```
//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

```
*\n* *\n* *\n* *
```

	1	2	3	4	5
1	*				
2		*			
3			*		
4				*	
5					*

	1	2	3	4	5
1	*	.			
2	*	*			
3	*	*	*		
4	*	*	*	*	
5	*	*	*	*	*

Compose them to simplify

1st
2nd
3rd
4th
5th

```
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>i</i>	1	2	3	4	5
<i>j</i>	1	*				
	2		*			
	3			*		
	4				*	
	5	*				

$i+j = N+1$ → & break
else → Space

```

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

The output format shows a 5x5 grid of asterisks (*). The stars are positioned at the following coordinates relative to the top-left corner: Row 1: (1,1), (1,5); Row 2: (2,2), (2,4); Row 3: (3,3); Row 4: (4,2), (4,4); Row 5: (5,1), (5,5). This creates a central cross shape with stars at the corners and the center.

	1	2	3	4	5
1	*				*
2		*		*	
3			*		
4		*		*	
5	*				*

```

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 $n=5$

	1	2	3	4	5	6	7
1	*	*	*		*	*	*
2	*	*			*	*	
3	*					*	
4	*	*			*	*	
5	*	*	*		*	*	*

$$k = n/2 + 1$$

↓

3

for $n=7$

	1	2	3	4	5	6	7	8	9
1	*	*	*	*		*	*	*	*
2	*	*	*				*	*	*
3	*	*						*	*
4	*								
5	*	*						*	*
6	*	*	*				*	*	*
7	*	*	*	*			*	*	*

$$k = 4$$

~~# first half~~

→ Starts → Start from k
& dec by 1

for $n=5$

	1	2	3	4	5	6	7
1	*	*	*		*	*	*
2	*	*			*	*	
3	*						
4	*	*					
5	*	*	*				

Second Half

Stars \rightarrow start from 2
 ℓ inc by 1

Spaces \rightarrow start from $n-2$
 ℓ dec by 2

for $n=5$

	1	2	3	4	5	6	7
1	*	*	*		*	*	*
2	*	*				*	*
3	*					*	*
4	*	*				*	*
5	*	*	*		*	*	*

```
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;
}

}
```

Getting Started Lecture : 7

- ① Number Pattern → PEP
- ② Cross of Numbers → CN
- ③ Empty Diamond
- ④ fibonacci Triangle
- ⑤ Mirror image of Triangle

Number Pattern

Pattern 11

Easy

Prev Next

1. You are given a number n.
2. You've to create a pattern as shown in output format.

Input Format

A number n

Output Format

```
1
2   3
4   5   6
7   8   9   10
..
```

$N=5$

	1	2	3	4	5
1	1				
2	2	3			
3	4	5	6		
4	7	8	9	10	
5	11	12	13	14	15

Count = 1

① → count Print
count + 1

* ↗ * ↗ * ↗ * ↗ * ↗ * ↗ * ↗ * ↗

2 N

$N=5$

	1	2	3	4	5
1	1				
2	2	3			
3	4	5	6		
4	7	8	9	10	
5	11	12	13	14	15

1
→
2 - 3 -

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

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

    int num = 1; ① ③
    for(int i=1;i<=n;i++) {
        //numbers

        for(int j=1;j<=i;j++) {
            System.out.print(num + "\t");
            num++;
        }
        //Leave Line
        System.out.println();
    }
}
```

$N=5$

~~2~~ 2
num 4

~~1~~ 1
c
~~2~~
j
~~3~~ ③

Problem Statement

Ninja wants to build a number pattern for the given integer.

For example, If the given integer 'N' is 4

Pattern:

1
23
345
4567

$N=5$

	1	2	3	4	5
1	1				
2	2	3			
3	3	4	5		
4	4	5	6	7	
5	5	6	7	8	9

1st
2nd
3rd
4th
5th

```
1 import java.util.*;
2 public class Main {
3     public static void main(String[] args) {
4         Scanner scn = new Scanner(System.in);
5         int n = scn.nextInt();
6
7         for(int i=1;i<=n;i++) {
8             //numbers
9             int num = i;
10            for(int j=1;j<=i;j++) {
11                System.out.print(num + "\t");
12                num++;
13            }
14
15            //leave line
16            System.out.println();
17        }
18    }
19 }
```

→ This code is for
Number Pattern
on CN (Print
only)

Cross of Numbers

 Play with Pattern

7 Difficulty: EASY

Avg. time to solve
20 min

Success Rate
80%

Problem Statement

Ninja wanted to go to a party along with his friends. However, his mom wanted him to go only if he completes a task assigned by her.

She gave Ninja a value and asked him to print an X-shaped pattern.

Example : Pattern for N = 3 (No. of rows = 5, No. of columns = 5) :

```
1   1
 2 2
 3
 2 2
1   1
```

	1	2	3	4	5
1	1				1
2		2	2		
3			3		
4		2	2		
5	1				1

3

$$(S - 4 + 1) = 2$$

$$(S - 5 + 1) = 1$$

i j N

* * * * *

* * * *

* * *

* *

*

(N - i + 1)

S

```
public static void ninjaPattern(int N) {  
    // Write your code here.  
    // N= 3  
    // n = 2*N-1  
  
    int n = 2*N-1;  
  
    for(int i=1;i<=n;i++) {  
        for(int j=1;j<=n;j++) {  
            if(i == j || i+j == n+1) {  
                if(i <= n/2 + 1) System.out.print(i);  
                else System.out.print(n-i+1);  
            } else {  
                System.out.print(" ");  
            }  
        }  
        System.out.println();  
    }  
}
```

#Empty Diamond

Pattern 10

● 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 $N=5$

	1	2	3	4	5
1			*		
2		*		*	
3	*				*
4		*	*		
5		*			

1st
2nd
3rd
 $n/2 + 1 (3)$
 \underbrace{k}

first

① $k-i$ spans

② 1 star

③ Space starts from 1 & inc by 2 always

$N=7$

	1	2	3	4	5	6	7
1				*			
2					*		
3	*					*	
4		*					*
5			*				*
6				*			
7					*		

$n/2 + 1 (4)$
 \underbrace{k}

④ 1 star

for $N=5$

1	2	3	4	5
1		*		
2	*		*	
3	*			*
4	*	*	*	
5		*		

1^{st}
 2^{nd}
 $3^{rd} \frac{n}{2} + 1 (3)$
 } k

$N=7$

1	2	3	4	5	6	7
1			*			
2			*		*	
3			*			*
4	*					*
5		*				*
6			*		*	
7				*		

$n/2 + 1 (4)$
 } k

Second Half

① i spans

② 1 star

③ Space = 4;
④ 1 star

last-space = space - 2;

Sec-half-space = last-space / 2;

sp = 1

for (int i = 1; i <= 3; i++) {

 printSpace()

 sp += 2;

}

1 3 5

i = 1 ≠ 3

2 ≠ 5 ≠ 7

space

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

    // write ur code here
    int n = scn.nextInt();
    int space = 1;
    int k = n/2 + 1;

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

        System.out.print("*\t");
        if(i == 1) {
            System.out.println();
            continue;
        }

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

        System.out.println("**");
        space += 2;
    }
}
```

**

```
space -= 4; //this is explained in the class

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

    System.out.print("*\t");
    if(i == n-k) {
        System.out.println();
        continue;
    }

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

    System.out.print("*\t");

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

You are screen

fibonacci Triangle

Pattern 12

● Easy

◀ Prev ▶ Next

1. You are given a number n.
2. You've to create a pattern as shown in output format.

Input Format

A number n

Output Format

```
0
1      1
2      3      5
8      13     21   34
..
```

	1	2	3	4	5
1	0				
2	1	1			
3	2	3	5		
4	8	13	21	34	
5	55	89	144	233	377

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

$$\text{fib}(N) = \text{fib}(N-1) + \text{fib}(N-2)$$

```
// write ur code here
int n = scn.nextInt();
int a = 0;
int b = 1;
int c = 0;
for(int i=1;i<=n;i++) {
    for(int j=1;j<=i;j++) {
        if(i == 1) {
            System.out.print("0\t");
        } else if(i == 2 && j==1) {
            System.out.print("1\t");
        } else {
            c = a + b;
            System.out.print(c + "\t");
            a = b;
            b = c;
        }
    }
    System.out.println();
}
```

$$a = \cancel{0} \cancel{1} \cancel{2} \cancel{3}$$

$$b = \cancel{1} \cancel{2} \cancel{3} \cancel{5}$$

$$c = \cancel{0} \cancel{1} \cancel{2} \cancel{5}$$

$$\boxed{?}$$

l

$$j = \cancel{1} \cancel{2} \cancel{3} \cancel{4}$$

$$0 -$$

$$1 - 1$$

→

$$2 - 3 = 5$$



Mirror image of triangle

3

Difficulty: EASY



Contributed By

Deepanshu_1780 | Level 1

Avg. time to solve

10 min



Success Rate

90%

Problem Statement

[Suggest Edit](#)

Ninja's younger sister got a project to print the pattern for the given number of rows, say 'N'. She finds it difficult to write such a pattern for every 'N'.

Ninja decided to help his sister and decided to write a code for this but wasn't able to write the code for the same. Ninja wants help to write a code and asks you for help. Help Ninja.

Example:

Pattern for N = 2

0

101

21012

for $N=2$

	1	2	3	4	5
0			0		
1		1	0	1	
2	2	1	0	1	2

$[1 \text{ to } N] \rightarrow N$

1 2 3 4 ... N

$[1 \text{ to } 5]$

1 2 3 4 5

$\{0 \text{ to } N\} \rightarrow N+1 \text{ times}$

$[0 \text{ to } N-1] \rightarrow N$

$[0 \text{ to } 4]$

Spaces $\rightarrow N-i$

2^{i+1} Numbers $\rightarrow \text{print}$

0, 1, 2, 3, 4

```

public class Solution {
    public static void NumberPattern(int n) {
        // Write your code here

        for(int i=0;i<=n;i++) {

            int num = i;

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

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

            System.out.println();
        }
    }
}

```

$j \leq 5$

for $N=2$

1	2	3	4	5
0		0		
1	1	0	1	
2	1	0	1	2

$$(2^i + 1) / 2 + 1$$

$$= (2 * 2 + 1) / 2 + 1 \quad 2 \ 1 \ 0 \ 1 \ 2$$

$$\begin{aligned} & 2^1 * 2 + 1 \\ & = 2 + 1 - 3 \end{aligned}$$

$$\begin{aligned} \text{num} &= 2 \\ & 2 \end{aligned}$$

$$\begin{aligned} j &= 1 \\ & 3 \end{aligned}$$