Abhishek Sharma

Results-driven civil engineer with expertise in project management and a track record of successful outcomes. Transitioning to the analyst field to utilize my analytical mindset, attention to detail, and problem-solving abilities. Skilled in data analysis, statistical modeling, and driving informed decision-making. Effective communicator and collaborator with a strong technical background. Seeking a challenging analyst role to contribute to organizational growth and success.

sharma.abhishek1408@gmail.com
+91-8375898754
Varanasi, U.P.
linkedin.com/in/abhisheksharma1408

EDUCATION

B.Tech
Dr.A.P.J.Abdul Kalam Technical University (Inderprastha Engineering College)
08/2016 - 09/2020 *Ghaziabad, U.P.*
Civil Engineering
- 75.2%

Intermediate
W.H.Smith Memorial School
04/2014 - 05/2015 *Varanasi, U.P.*
PCM
- 62.6%

High School
W.H.Smith Memorial School
04/2012 - 05/2013 *Varanasi, U.P.*
Science
- 69%

WORK EXPERIENCE

Civil Site Engineer
M/S Bhangal Construction Company
12/2021 - 12/2022 *Dhubri, Assam*
Job Responsibility
- Worked on the Dhubri-Phulbari Bridge Project. My role was to supervise the execution of the Well Foundation, ensuring compliance with the technical and drawing specifications. Additionally, I was responsible for providing all necessary machinery and equipment to ensure safe and uninterrupted progress without any issues or delays.

Trainee
Lucknow Metro Rail Corporation
06/2019 - 07/2019 *Lucknow, U.P.*
Task
- I worked as an intern Civil Engineer at the casting yard of the Lucknow Metro, where I had the opportunity to observe the process involved in the casting of U-Girders.

Trainee
S.P.Singhla Construction Pvt. Ltd.
01/2019 - 02/2019 *Buxar, Bihar*
Tasks
- I worked as an intern for a three-week training period, during which I had the opportunity to learn about the fabrication of cutting edges.

SKILLS

SQL · OOPS · Data Analytics · Tableau · Python · Core Java · C · MS Excel · Power BI · ETabs · Staad Pro · Oracle Primavera · Time Management · Communication Skill

PERSONAL PROJECTS

Intelligent Door Lock (05/2020 - 09/2020)
- An Alexa enabled door lock with face recognition and remote control powered by Amazon Web Services.

Echo Music Player (07/2019 - 08/2019)
- An Android app development project using Kotlin. It is a basic music player app made under the Internshala training Program

Conducting Seismic Analysis and Design of Steel Structures with diverse Bracing Systems using ETabs. (09/2019 - 05/2020)
- In the present study, three steel structure buildings were chosen and modeled using different bracing systems. Through response spectrum analysis, a comparison was made between the top storey displacement, inter-storey drift, base shear, and stiffness of the modeled buildings and the original ones.

CERTIFICATES

Google Cloud Fundamentals: Core Infrastructure Coursera

Google Data Analytics Coursera

SQL Essentials Linkedin

Excel Essential Linkedin

SQL Basics HackerRank

Essentials of Power BI Digisaksham

LANGUAGES

English	Hindi
Full Professional Proficiency	*Native or Bilingual Proficiency*

BITWISE OPERATOR

In Java, bitwise operators perform operations on integer data at the individual bit-level. Here, the integer data includes byte, short, int, and long types of data.

There are 7 operators to perform bit-level operations in Java: -

1. **BITWISE OR**
2. **BITWISE AND**
3. **BITWISE XOR**
4. **BITWISE COMPLEMENT**
5. **LEFT SHIFT**
6. **SIGNED RIGHT SHIFT**
7. **UNSIGNED RIGHT SHIFT**

Here we will discuss about the Shift Operators: -

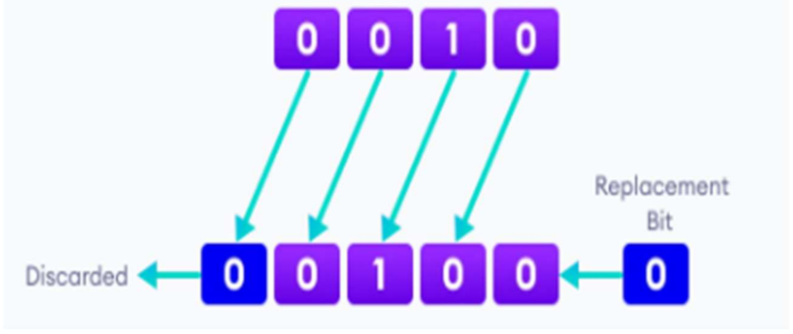
- **Left Shift or Signed Left Shift Operator: -**

The signed left shift operator (<<) shifts a bit pattern to the left. It is represented by the symbol **<<.** It also preserves the leftmost bit (sign bit).

In general, if we write a<<n, it means to shift the bits of a number toward the left with specified position (n). In the terms of mathematics, we can represent the signed right shift operator as follows:

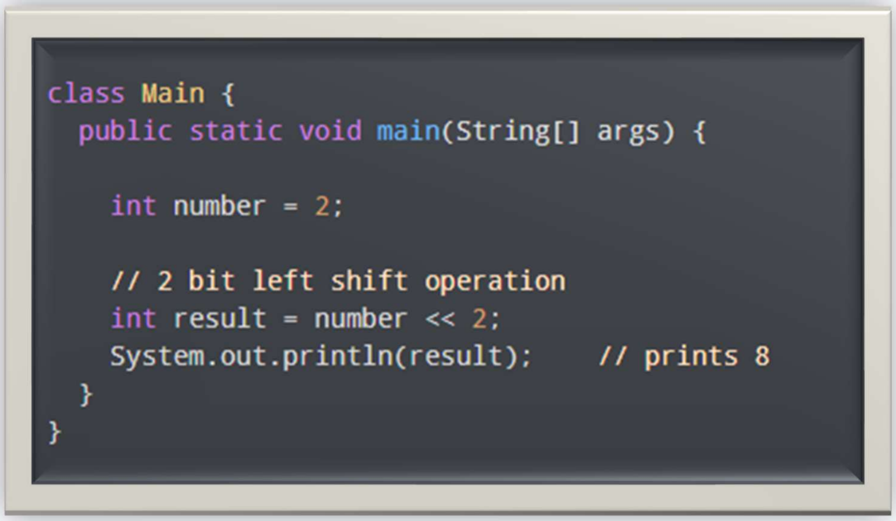
$$b = a >> n \longrightarrow b = a^{*} (2^{n})$$

The left shift operator shifts all bits towards the left by a certain number of specified bits.



As we can see from the image above, we have a 4-digit number. When we perform a 1-bit left shift operation on it, each individual bit is shifted to the left by **1** bit.
As a result, the left-most bit (most-significant) is discarded and the right-most position(least-significant) remains vacant. This vacancy is filled with **0s**.

Program: -



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

    int number = 2;

    // 2 bit left shift operation
    int result = number << 2;
    System.out.println(result);    // prints 8
  }
}
```

- **Signed Right Shift: -**

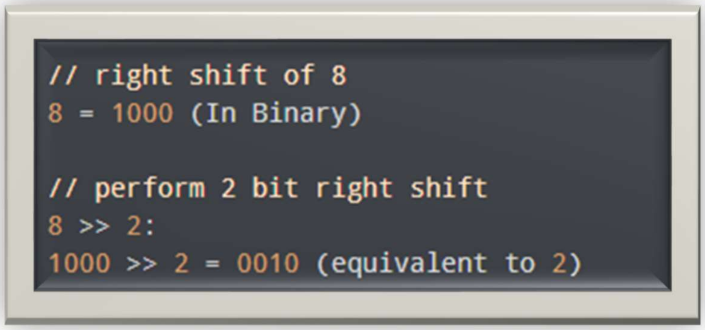
The signed right shift operator shifts a bit pattern of a number towards the **right** with a specified number of positions and fills 0. The operator is denoted by the symbol **>>.** It also preserves the leftmost bit (sign bit). If **0** is presented at the leftmost bit, it means the number is **positive**. If **1** is presented at the leftmost bit, it means the number is **negative**.

In general, if we write a>>n, it means to shift the bits of a number toward the right with a specified position (n). In the terms of mathematics, we can represent the signed right shift operator as follows:

$$b = a >> n \longrightarrow b = a / 2^n$$

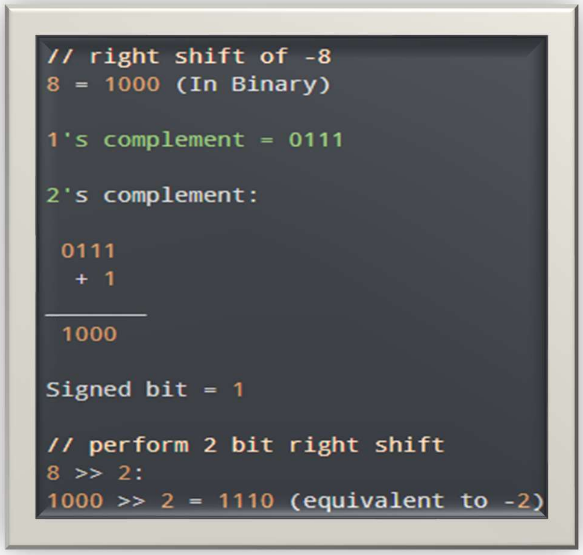
Note: When we apply right shift operator on a positive number, we get the positive number in the result also. Similarly, when we apply right shift operator on a negative number, we get the negative number in the result also.

The signed right shift operator shifts all bits towards the right by a certain number of specified bits. When we shift any number to the right, the least significant bits (rightmost) are discarded and the most significant position (leftmost) is filled with the sign bit.



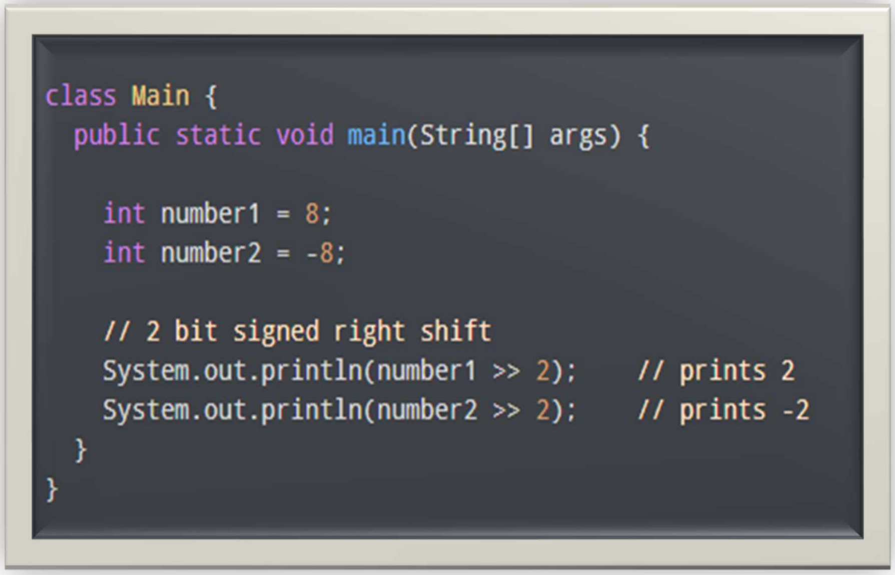
```
// right shift of 8
8 = 1000 (In Binary)

// perform 2 bit right shift
8 >> 2:
1000 >> 2 = 0010 (equivalent to 2)
```

Here, we are performing the right shift of 8 (i.e., sign is positive). Hence, there no sign bit. So, the leftmost bits are filled with 0 (represents positive sign).



```
// right shift of -8
8 = 1000 (In Binary)

1's complement = 0111

2's complement:

 0111
  + 1
_______
 1000

Signed bit = 1

// perform 2 bit right shift
8 >> 2:
1000 >> 2 = 1110 (equivalent to -2)
```

Program: -



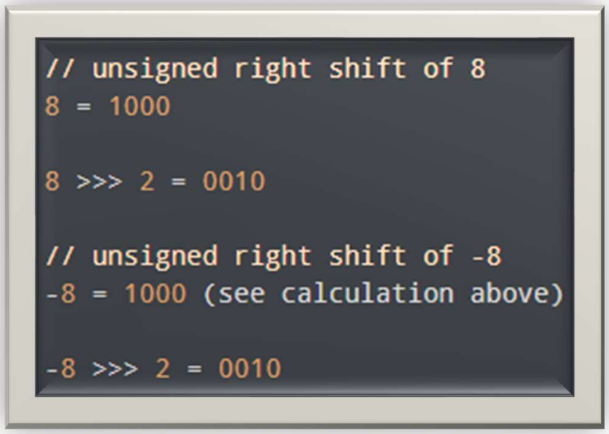
```
class Main {
  public static void main(String[] args) {

    int number1 = 8;
    int number2 = -8;

    // 2 bit signed right shift
    System.out.println(number1 >> 2);    // prints 2
    System.out.println(number2 >> 2);    // prints -2
  }
}
```

- **<u>Unsigned Right Shift</u>: -**

It shifts a zero at the leftmost position and fills 0. It is denoted by the symbol >>>. Note that the leftmost position after >> depends on the sign bit. It does not preserve the sign bit.

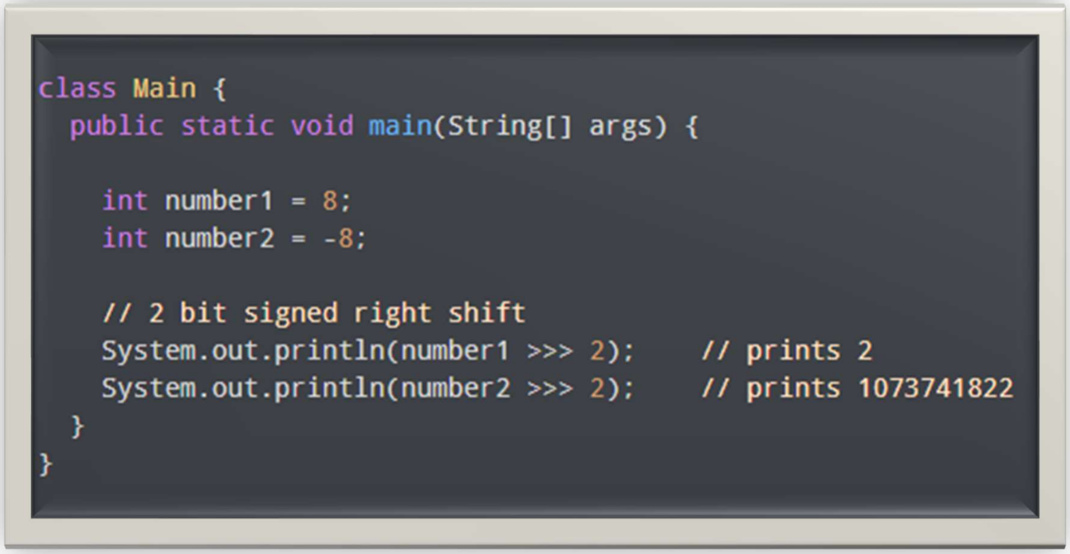
Example: If a=11110000 and b=2, find a>>>b?
a >>> b = 11110000 >>> 2 = **00111100**

The left operand value is moved right by the number of bits specified by the right operand and the shifted bits are filled up with zeros. Excess bits shifted off to the right are discarded. Therefore, before shifting the bits the decimal value of a is 240, and after shifting the bits the decimal value of a is 60.



```
// unsigned right shift of 8
8 = 1000

8 >>> 2 = 0010

// unsigned right shift of -8
-8 = 1000 (see calculation above)

-8 >>> 2 = 0010
```

Program: -



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

    int number1 = 8;
    int number2 = -8;

    // 2 bit signed right shift
    System.out.println(number1 >>> 2);    // prints 2
    System.out.println(number2 >>> 2);    // prints 1073741822
  }
}
```