Biginteger in Java

Jayasimha Y

Biginteger:

Primitive data types have fixed range and size in handling the integer values. But when the numbers exceeding the limit of primitive data types has to be stored “Biginteger” data type is used. Unlike primitive types biginteger doesn’t have fixed range and size to hold the integers.

Let us look at an example for performing the arithmetic operation on huge figures using biginteger data type.

Java code for Biginteger data type:

```
 class BigInteger {
   public static void main(String[] args) {
       BigInteger  a=12345678901234567890;
       BigInteger  b=98765432109876543210;
       BigInteger sum, BigInteger difference;
       BigInteger sum = a+b;
       BigInteger difference = a-b;
       System.out.println("Sum: " + sum);
       System.out.println("Difference: " + difference);


   }
}
```

The BigInteger class in Java is implemented as an array of integers, where each element of the array stores a portion of the number. The size of each element in the array is typically 32 bits (4 bytes) on most systems.

To calculate the number of bytes required for a BigInteger object, you can use the following formula:

Number of bytes = (bitLength + 7) / 8.

For example, if you have a BigInteger object representing the number 1000, the bitLength would be 10 (since it can be represented in 10 bits). Applying the formula, you would get:

Number of bytes = (10 + 7) / 8 = 17 / 8 = 2.125 bytes.