Chi Nguyen Moderation project report

Using the rational class from Data Structures, I added two more operations: divide and tothepowerof. I also implemented the BigInteger class from Java.BigInteger class is used for complex mathematical operations which involve very big integer calculations that are outside the limit of all available primitive data types. By default, the int data type is a 32-bit signed two's complement integer, which has a minimum value of -2^31 and a maximum value of 2^31-1. The long data type is a 64-bit two's complement integer. The signed long has a minimum value of -2^63 and a maximum value of 2^63-1. However, with Java.BigInteger class there is no theoretical limit on the upper bound of the range because memory is allocated dynamically but practically as memory is limited you can store a number which has 2^31-1 (Int max value) number of bits in it which should be sufficient to store mostly all large values.

The BigInteger class has a large method library and it is also used a lot in competitive programming. BigInteger class provides analogs to all of Java's primitive integer operators, and all relevant methods from java.lang.Math. Furthermore, BigInteger also has operations for modular arithmetic, GCD, primality testing, prime generation, bit manipulation, and other operations. All of the BigInteger methods and constructors throw NullPointerException when passed a null object reference for any input parameter.

**Constructor**

*BigInteger(String val)*

* Take the input of a string type of a number value. The String representation consists of an optional minus sign followed by a sequence of one or more decimal digits not containing any extraneous characters (whitespace, for example).
* Throws: NumberFormatException when val is null (zero-byte long).

**Method Use**

*Int compareTo(BigInteger val)*

* Compare this BigInteger with the specified BigInteger. Using boolean comparison operators.
* Return: 1, 0, or 1 as this BigInteger is numerically less than, equal to, or greater than val.

*BigInteger gcd(BigInteger val)*

* The greatest common divisor of the 2 BigInt
* Returns a BigInteger whose value is the greatest common divisor of abs(this) and abs(val).

Adding Big int

*BigInteger add(BigInteger val)*

* Adding 2 BigInt
* Returns a BigInteger whose value is (this + val).

*BigInteger subtract(BigInteger val)*

* Subtracting 2 BigInt
* Returns a BigInteger whose value is (this - val).

*BigInteger multiply(BigInteger val)*

* Multiplying 2 BigInt
* Returns a BigInteger whose value is (this \* val).

*BigInteger divide(BigInteger val)*

* Dividing 2 BigInt
* Returns a BigInteger whose value is (this / val).

*BigInteger mod(BigInteger m)*

* Returns a BigInteger whose value is (this mod m).

*BigInteger pow(int val)*

* Returns a BigInteger whose value is this to the power of val.

**Resources**

<https://docs.oracle.com/javase/7/docs/api/java/math/BigInteger.html>

<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>

<https://www.geeksforgeeks.org/biginteger-class-in-java/>