CS 610 852 Programming Assignment #2 (SP17) Arbitrary Length Integers

Using C, C++, or Java implement the following data structures and associated algorithms. Implement and automatically run the required test cases. Make sure your submission compiles and runs on AFS.

Many applications, such as cryptography, require very large integers (much larger than would fit into 32- or 64-bit integer types). The standard library of many languages includes a “Big Integer” or ”Long Integer” data type which can store arbitrarily large integers.

In this project you’ll implement an array-based **Long Integer Data Structure (“LongInt”)**, where there is no predefined limit on the maximum value that can be stored. Such a LongInt can have an arbitrary number of digits (e.g., tens of thousands of digits, or millions of digits). Additionally you will implement operations for adding, subtracting, and comparing long integers.

Your array-based LongInt data structure will store each number in 8 digit chunks an array of size *numberOfDigits/8* (e.g., a number with 23 digits will require an array of size three). The least significant digits will be stored at *array*[0] and the most significant digits will be stored at *array*[n-1]. The array will store the absolute value of the number and the sign of the number will be stored separately as an int with a value of 1 or -1.

A = 14100568100000545934529077115683593
B = -35000327459484745431789

These can be represented using the following arrays:

0	1	2	3	4
11568359	34529077	00005459	10056810	14

0	1	2
54317890	45948474	35000327

Implement the following operations for the Long Int data structure:

- LongInt(int sign, int [] array) – Initializes a LongInt via an array of integers and an integer containing 1 or -1, indicating the sign of the overall LongInteger
- void print() – Prints the String representation of the LongInt to standard output
- boolean equalTo(LongInt other) – Determines if the LongInt is equal to the LongInt other
- boolean lessThan(LongInt other) – Determines if the LongInt is less than the LongInt other
- boolean greaterThan(LongInt other) – Determines if the LongInt is greater than the LongInt other
- LongInt add(LongInt other) – Adds the LongInt and returns the result as a new LongInt
- LongInt subtract(LongInt other) – Subtracts the LongInt and returns the result as a new LongInt

Example skeleton code (in Java, not compiled or tested):

```
public class LongInt {
      private final int [] array;
      private final int sign;

      public LongInt(int sign, int [] array) {  }
      public void print() {  }
      public boolean equalTo(LongInt other) { }
      public boolean lessThan(LongInt other) { }
      public boolean greaterThan(LongInt other) { }
      public LongInt add(LongInt other) { }
      public LongInt subtract(LongInt other) { }
}
```

Test cases:

A = 1423550000000010056810000054593452907711568359
B = -350003274594847454317890
C = 293023902347029734029734209374209734209374209372348723498729348728934728937492
D = -98534342983742987342987339234098230498203894209928374662342342342356723423423
E =
84364131684386183513516846948354348943643518468434351684843516846843153846843138468
13153843135138413513843813513813138438435153454154515151513141592654543515316848613
242587561516511233246174561276521672162416274123076527612

1. Initialize Long Integers A-E.
2. Print each Long Integer to standard output using print().
3. For each Long Integer compare it to A-I using equalTo, lessThan, greaterThan (i.e., apply all three methods pair-wise, including on itself).
4. For each LongInt add it to every other LongInt (one at a time) and print the result.
5. For each LongInt subtract it from every other LongInt (one at a time) and print the result.

Other Notes:

- Think of this project as math in Base 10,000,000. All of the operation you know math from Base 10 are more or less the same.
- Note that when adding two eight digit numbers you may get a ninth digit (e.g., 99999999 + 99999999 = 199999998). In such a case you will need to handle the overflow digit properly in your algorithms. Similarly, you may need to borrow a digit when doing subtraction.
- Use of the long primitive type (or double, or long long, or anything similar) is not permitted at any time. Only ints can be used.
- A LongInt can NEVER be converted into a String. No exceptions. Strings can only be used in print(), if needed.
- A LongInt can never be converted into another data structure or data type. Only arrays of ints are allowed.