Computer Science and Engineering

Fifth Semester

CS6308 / JAVA PROGRAMMING LAB TEST

(Regulation 2018 - RUSA)

Time: 1Hr:15 mins **Date:24/11/23** **Marks :15**

Create a class called LargeInteger, which will represent an integer that has the capabilities to store large values as well as values within the range of ints. To represent a large integer two fields are used such as sign and magnitude. Sign is an int that stores -1, if the number is negative, 0 if the number is zero, and 1 if the number is any nonzero positive number. Magnitude is an int array where each element represents a digit in the long integer. For example, the values 123456 and -123456 will have the magnitude {1, 2, 3, 4, 5, 6}. LargeInteger also contains methods that will perform integer operations on these LargeInteger objects. The methods are listed below:

1) public LargeInteger(String numStr) (1)

This constructor creates a LargeInteger object with the integer value represented by numStr. Negative numbers are represented with a negative sign in the beginning (ex. "-123456"). Zero will be represented as "0". All nonzero positive numbers will not have any sign (ex. "123456"). Numbers with preceding zeros ("0000123", " -0000123", "0000") are invalid. Write custom exception to handle this data.

2) public LargeInteger(LargeInteger otherNum) (1)

This is a copy constructor that will create a LargeInteger object. This constructor creates a deep copy of otherNum.

3) public String toString() (1)

This method will return a String that represents the value of the LargeInteger

4) public boolean equals(Object obj) (1)

This method overrides the Java Object's equals(Object obj) method. This method returns true if this LargeInteger is equivalent to obj. It first checks whether the object is of type LargeInteger. use instanceof to check this. Then it checks whether the value of sign and magnitude are the same. If the fields have the same value, it returns true. Otherwise, return false.

5)public LargeInteger multiply10() (1)

This method multiplies 10 to this LargeInteger object and returns a new LargeInteger object that represents this value. It has to append a 0 to the magnitude array.

6)public boolean isEven() (1)

This method returns true if this LargeInteger is even and returns false otherwise.

7) public LargeInteger negate() (1)

This method returns the negated value of this LargeInteger

8)public int compare(LargeInteger otherNum) (1)

This method compares this LargeInteger object with otherNum and returns either -1, 0, or 1. It returns -1 if *this* LargeInteger is smaller, 1 if *this* is large, and 0 if *this* is the same as otherNum

9)public static LargeInteger add(LargeInteger left, LargeInteger right) (1)

This method adds left and right and returns a new LargeInteger representing the sum of left + right. Do not modify the fields of left or right. Instead, create a new LargeInteger.

10) LargeInteger modulus(LargeInteger otherNum) (1)

This method computes the modulus operation on this LargeInteger with otherNum (this % otherNum). Dont write a divide method for this function. To compute the mod operation simply keep subtracting this by otherNum (this - otherNum) until this is less than otherNum (this < otherNum). Use implementation of subtract() and compare() here

11) void Createfile() (5)

Create a Textfile named “LargeInt.txt” which will store the set of large intergers. Read the file and print numbers which has the magnitude of more than 10.

Create a LargeIntegerTester class to fully test the correctness of LargeInteger implementation.