National University of Computer and Emerging Science
Lahore Campus
Object Oriented Programming

Assignment # 3

Due Date: Sunday, May 14, 2023 by 11:59pm.

1. If there is a syntax error in code, zero marks will be awarded in that part of assignment.

2. Your code must be generic

Overflow Management

The native int type in C++ cannot store large integers of more than 10 decimal digits. In this project, you will implement a bigint class that behaves like the int type, but can handle unsigned integers of up to 45 digits.

1. You are to build a class named bigint that can deal with up to 45 decimal digits of unsigned integers. The bigint class is specified as follows.

2. The bigint class should have a five-element array (int v[5];) as its private date member. Each element should store 9 decimal digits.
   For example to store integer 1,300,000,000,000,900,000,000,100,000,000. You should have v[0] = 100000000, v[1] = 900000000, v[2] =0, v[3]= 1300, and v[4] = 0.

3. The bigint class should have the following constructors: bigint(); // default constructor, set the value to be 0 bigint(int x0); // set the value to be x0
   bigint(int x0, int x1); // set the value to be $x1*10^9+x0$

   bigint(int x0, int x1, int x2); // set the value to be $x2*10^{18}+x1*10^9+x0$ bigint(int x0, int x1, int x2, int x3); bigint(int x0, int x1, int x2, int x3, int x4);

4. To make bigint behave like int, you must overload the following operators

   - The extraction operator » for reading a bigint number from an input stream. The input number can have up to 45 digits. Following are two legitimate bigint numbers that can appear as input:
     1300000000000900000000100000000
     999999999999999999999999999999999999999999999

   - The insertion operation « for output a bigint number to an output stream. The number must be output like a regular int number with the first digit being a non zero.

   - 2 arithmetic operators + and – (*, and / are omitted for this type to simplify the program).

   - 6 comparison operators <, >, <=, >=, ==, and !=.

5. Once the class is implemented, you can test your implementation using the driver.cpp program. You should also write other driver programs to do a thorough test of your bigint implementation. Put your bigint class declaration in bigint.h and your bigint class implementation in bigint.cpp. Redirect test.txt as the standard input, the output of the program should be identical to ouputt.txt.

Submission

Single .cpp file. Name of the file should be your roll number.

Grading Policy

1. Program with no compiler error (20 points)

2. The default constructor and the overloaded « operator (20 points)

3. Other four constructors (3 points each, 12 points total)

4. Six comparison operators (2 points each, 12 points total)

5. + and – operators (8 points each, 16 point total)

6. The » operator (10 points)

Hints

1. You should then implement the default construct, and write a simple operator«() function that simply prints out the values of v[0], v[1], v[2], v[3], v[4].

2. After that you can implement the functions in the following order: the « operator, other constructors, the comparison operators, the arithmetic operators, and the » operator. Remember to compile and run the program at least once everytime you add one function.

3. To implement the « operator, you will need to output potentially 5 integers. You must use setw(9) and setfill('0') to output each value besides the most significant non-zero integer. You must have 'include <iomanip>' in the file in order to use these functions. The following line is an example to use these functions. s « setfile('0') « setw(9) « vv.v[j];

4. To implement the » operator, you can first read the string of digits into a character array and then manually convert it into a bigint number.

5. To implement the + and – operators, you must pay special attention to the carry bit in the operations.

Honor Policy

This task is a learning opportunity that will be evaluated based on your ability.**Plagiarized reports or code will get a zero**. If in doubt, ask the course instructor.