UEE1303 S18: Object-Oriented Programming

EXERCISE

What you will learn from Lab 7

EXERCISE:7-1 (BIGNUMBER)

- ✓ In C++, the largest `int` value is 2,147,483,647. So, an integer larger than this cannot be stored and processed as an integer. Similarly, if the sum or product of two positive integers is greater than 2,147,483,647, the result will be incorrect. One way to store and manipulate large integers is to store each individual digit of the number in an array.
- ✓ Create a class `HugeInteger` that uses a 40-element array of digits to store integer as large as 40 digits each. Provide member function `add`, `subtract`, `output`. For comparing `HugeInteger` objects, provide functions `isEqualTo`, `isNotEqualTo`, and `isLessThan` – each of these is a function that simply returns `true` if the relationship holds between the two `HugeInteger` and returns `false` if the relationship does not hold. Also please add the constructor and destructor to the class `HugeInteger`.

- ✓ The sample output is shown as follows

```
7654321 + 100000000000000 = 100000007654321
100000000000000 - 5 = 99999999999995
99999999999995 + 5 = 100000000000000
100000000000000 is equal to 100000000000000
7654321 is not equal to 100000000000000
5 is less than 100000000000000
```

- ✓ In this exercise, your `HugeInteger` class has defined three private data member as follow:

```
// HugeInteger.h
#ifndef HUGEINTEGER_H
#define HUGEINTEGER_H
class HugeInteger
{
public:
    // put your member function prototype
private:
    int *data;
    int length;

}
#endif
```

```
// HugeInteger.cpp
#include <iostream>
#include "HugeInteger.h"
using namespace std;
// Member-function definitions for class HugeInteger.
```

✓ In this exercise, your `HugeInteger` class has defined three private data member as follow:

```
// ex7-1.cpp
#include <iostream>
#include "HugeInteger.h"
using namespace std;
int main()
{
	HugeInteger n1( 7654321 ); // HugeInteger object n1
	// HugeInteger object n2
	HugeInteger n2( "100000000000000" );
	HugeInteger n4( 5 ); // HugeInteger object n4
	HugeInteger n5; // HugeInteger object n5
	n1.output();
	cout << endl;
	n2.output();
	cout << endl;
	n4.output();
	cout << endl;
	// outputs the sum of n1 and n2
	n5 = n1.add( n2 );
	n1.output();
	cout << " + ";
	n2.output();
	cout << " = ";
	n5.output();
	cout << endl;
	// assigns the difference of n2 and n4 to n5 then outputs n5
	n5 = n2.subtract( n4 );
	n2.output();
	cout<< " - ";
	n4.output();
	cout << " = ";
	n5.output();
	cout << endl;
	HugeInteger n3(n5); // call copy constructor
	// outputs the sum of n3 and n4
	n2 = n3.add ( n4 );
	n3.output();
	cout<< " + ";
	n4.output();
	cout << " = ";
	n2.output();
	cout << endl;
```

```
    // checks for equality between n2 and n2
    if ( n2.isEqualTo( n2 ) == true )
    {
        n2.output();
        cout << " is equal to ";
        n2.output();
        cout << endl;
    } // end if
    // checks for inequality between n1 and n2
    if ( n1.isNotEqualTo( n2 ) == true )
    {
        n1.output();
        cout << " is not equal to ";
        n2.output();
        cout << endl;
    } // end if

    // tests for smaller number between n4 and n2
    if ( n4.isLessThan( n2 ) == true )
    {
        n4.output();
        cout << " is less than ";
        n2.output();
        cout << endl;
    } // end if*/
    return 0;
}
```