```cpp
#include <iostream>
#include "HugeInteger.h"
using namespace std;

int main()
{
    HugeInteger num1("63434");
    HugeInteger num2("4534");
    HugeInteger total;

    num1.add(num2, total);
    cout << "num1 + num2: ";
    num1.output(total);
    num1.subtract(num2, total);
    cout << "\nThe difference between two integers: ";
    num1.output(total);
    num1.multiply(num2, total);
    cout << "\nnum1 * num2: ";
    num1.output(total);
    //num1.divide(num2, total);
    cout << "\n";
    //num1.modulus(num2, total);

    cout << "num1 is equal to num2: ";
    num1.isEqualTo(num2);

    cout << "num1 is not equal to num2: ";
    num1.isNotEqualTo();

    cout << "num1 is greater than num2: ";
    num1.isGreaterThan();

    cout << "num1 is less than num2: ";
    num1.isLessThan();

    cout << "num1 is greater than or equal to num2: ";
    num1.isGreaterThanOrEqualTo();

    cout << "num1 is less than or equal to num2: ";
    num1.isLessThanOrEqualTo();

    cout << "num1 is zero: ";
    num1.isZero();

    return 0;
}
```

```
#ifndef HugeInteger_H
#define HugeInteger_H
#include <string>
using namespace std;

class HugeInteger{
    public:
        static const int SIZE = 40;
        int digits[SIZE] = {};

        HugeInteger(string = "0");
        void input(string);
        void output(HugeInteger &);

        void add(HugeInteger &, HugeInteger &);
        void subtract(HugeInteger &, HugeInteger &);
        void multiply(HugeInteger &, HugeInteger &);
        void divide(HugeInteger &, HugeInteger &);
        void modulus(HugeInteger &, HugeInteger &);

        void isEqualTo(HugeInteger &);
        void isNotEqualTo();
        void isGreaterThan();
        void isLessThan();
        void isGreaterThanOrEqualTo();
        void isLessThanOrEqualTo();
        void isZero();

    private:
        string inputValue;
        int valueLength;
        int largerLength;
        int lengthAfterCalculation;
        int largerInteger;
        int equalCounter = 0;
        int zeroCounter = 0;

        void beforeSubtract(HugeInteger &);
};

#endif
```

```
#include <iostream>
#include <string>
#include "HugeInteger.h"
using namespace std;

HugeInteger::HugeInteger(string inputValue){
    input(inputValue);
}

void HugeInteger::input(string inputValue){

    valueLength = inputValue.length();

    for (int i=39, j=inputValue.length()-1; i>=0, j>=0; i--, j--){
        digits[i] = inputValue[j] - '0';
    }

}

void HugeInteger::output(HugeInteger &total){
    if (lengthAfterCalculation != 0){
        for (int i=40-lengthAfterCalculation; i<40; i++){
            cout << total.digits[i];
        }
    }
    else if (lengthAfterCalculation == 0){
        cout << 0;
    }
    lengthAfterCalculation = 0;
    cout << endl;

}
```

```
void HugeInteger::add(HugeInteger &num2, HugeInteger &total){

    // 清空儲存格
    for (int i=0; i<40; i++){
        total.digits[i] = 0;
    }

    // 判斷哪個integer的位數比較大
    if (valueLength >= num2.valueLength){
        largerLength = valueLength;
    }
    else{
        largerLength = num2.valueLength;
    }

    // 相加
    for (int i=39; i>=39-largerLength; i--){
        total.digits[i] += digits[i] + num2.digits[i];
    }

    // 進位處理
    for (int i=39; i>=0; i--){
        if (total.digits[i] >= 10){
            total.digits[i] = total.digits[i] % 10;
            total.digits[i-1] += 1;
        }
    }

    // 算相加後的位數
    for (int i=0; i<40; i++){
        if (total.digits[i] > 0){
            lengthAfterCalculation = 40 - i;
            break;
        }
    }

}
```

```
void HugeInteger::beforeSubtract(HugeInteger &num2){
    if (valueLength > num2.valueLength){
        largerInteger = 1;
    }
    else if (valueLength == num2.valueLength and digits[40-valueLength] > num2.digits[40-num2.valueLength]){
        largerInteger = 1;
    }
    else if (valueLength < num2.valueLength){
        largerInteger = 2;
    }
    else if (valueLength == num2.valueLength and digits[40-valueLength] < num2.digits[40-num2.valueLength]){
        largerInteger = 2;
    }
    else if (valueLength == num2.valueLength and digits[40-valueLength] == num2.digits[40-num2.valueLength]){
        largerInteger = 0;
    }

}
```

```
void HugeInteger::subtract(HugeInteger &num2, HugeInteger &total){

    // 清空儲存格
    for (int i=0; i<40; i++){
        total.digits[i] = 0;
    }

    // 相減且做借位處理
    beforeSubtract(num2);
        // num1大
    if (largerInteger == 1 or largerInteger == 0){

        int temp[40];
        for (int i=0; i<40; i++){
            temp[i] = digits[i];
        }

        for (int i=39; i>39-largerLength; i--){
            if (temp[i] - num2.digits[i] < 0){
                total.digits[i] = temp[i] + 10 - num2.digits[i];
                temp[i-1] -= 1;
            }
            else if (temp[i] - num2.digits[i] >= 0){
                total.digits[i] = temp[i] - num2.digits[i];
            }
        }

    }
        // num2大
    else if (largerInteger == 2){

        int temp[40];
        for (int i=0; i<40; i++){
            temp[i] = num2.digits[i];
        }

        for (int i=39; i>39-largerLength; i--){
            if (temp[i] - digits[i] < 0){
                total.digits[i] = temp[i] + 10 - digits[i];
                temp[i-1] -= 1;
            }
            else if (temp[i] - digits[i] >= 0){
                total.digits[i] = temp[i] - digits[i];
            }
        }

    }
```

```
    // 算相減後的位數
    for (int i=0; i<40; i++){
        if (total.digits[i] > 0){
            lengthAfterCalculation = 40 - i;
            break;
        }
    }

}

void HugeInteger::multiply(HugeInteger &num2, HugeInteger &total){

    // 清空儲存格
    for (int i=0; i<40; i++){
        total.digits[i] = 0;
    }

    // num1 * num2
    int k = 39;
    for (int i=39; i>=40-num2.valueLength; i--){
        k = i;
        for (int j=39; j>=40-valueLength; j--){
            total.digits[k] += digits[j] * num2.digits[i];

            // 進位處理
            if (total.digits[k] >= 10){
                total.digits[k-1] += total.digits[k] / 10;
                total.digits[k] %= 10;
            }
            k--;
        }

    }

    // 算相乘後的位數
    for (int i=0; i<40; i++){
        if (total.digits[i] > 0){
            lengthAfterCalculation = 40 - i;
            break;
        }
    }

}
```

```
void HugeInteger::divide(HugeInteger &num2, HugeInteger &total){

    // 清空儲存格
    for (int i=0; i<40; i++){
        total.digits[i] = 0;
    }


}

void HugeInteger::modulus(HugeInteger &num2, HugeInteger &total){

    // 清空儲存格
    for (int i=0; i<40; i++){
        total.digits[i] = 0;
    }



}

void HugeInteger::isEqualTo(HugeInteger &num2){
    for (int i=0; i<40; i++){
        if (digits[i] == num2.digits[i]){
            equalCounter++;
            if (equalCounter == 40){
                cout << "True" << endl;
                break;
            }
        }
        else if (digits[i] != num2.digits[i]){
            cout << "False" << endl;
            break;
        }
    }

}

void HugeInteger::isNotEqualTo(){
    if (equalCounter == 40){
        cout << "False" << endl;
    }
    else{
        cout << "True" << endl;
    }

}
```

```
void HugeInteger::isGreaterThan(){
    if (largerInteger == 1){
        cout << "True" << endl;
    }
    else{
        cout << "False" << endl;
    }

}

void HugeInteger::isLessThan(){
    if (largerInteger == 1 or largerInteger == 0){
        cout << "False" << endl;
    }
    else{
        cout << "True" << endl;
    }

}

void HugeInteger::isGreaterThanOrEqualTo(){
    if (largerInteger == 1 or largerInteger == 0){
        cout << "True" << endl;
    }
    else{
        cout << "False" << endl;
    }

}

void HugeInteger::isLessThanOrEqualTo(){
    if (largerInteger == 2 or largerInteger == 0){
        cout << "True" << endl;
    }
    else{
        cout << "False" << endl;
    }

}
```

```
271 void HugeInteger::isZero(){
272     for (int i=0; i<40; i++){
273         if (digits[i] != 0){
274             cout << "False" << endl;
275             break;
276         }
277         zeroCounter++;
278     }
279     if (zeroCounter == 40){
280         cout << "True" << endl;
281     }
282 
283 }
284 
```

```
num1 + num2: 67968

The difference between two integers: 58900

num1 * num2: 287609756

num1 is equal to num2: False
num1 is not equal to num2: True
num1 is greater than num2: True
num1 is less than num2: False
num1 is greater than or equal to num2: True
num1 is less than or equal to num2: False
num1 is zero: False
```