```cpp
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtdbg.h>

#ifdef _DEBUG
#ifndef DBG_NEW
#define DBG_NEW new ( _NORMAL_BLOCK , __FILE__ , __LINE__ )
#define new DBG_NEW
#endif
#endif // _DEBUG
//-----------------------------------------------------------
#include <iostream>
using namespace std;
#include "BigInt.h"
#include <string.h>
//-----------------------------------------------------------

BigInt BigInt_create(const char* s)
{
    BigInt a;
    //checking how much sfarot realy in the number (00042 -> number of sfarot is 2!)
    a.n_digits = no_leading_0_len(s);
    if (a.n_digits == 0)
        a.digits = nullptr;
    else
    {
        a.digits = new char[a.n_digits];
        int index = strlen(s) - a.n_digits;
        // assigning the number to array with the sfarot
        for (int i = 0; i < a.n_digits; i++)
        {
            a.digits[i] = s[index];
            index++;
        }
    }
    return a;
}

//checking how much sfarot realy in the number (00042 -> number of sfarot is 2!)
int no_leading_0_len(const char* s)
{
    int counter = 0;
    while (s[counter] == '0')
    {
        counter++;
    }
    int eff_len = strlen(s) - counter;
    return eff_len;
}

void BigInt_print(const BigInt& bi)
```

```cpp
{
    for (int i = 0; i < bi.n_digits; i++)
    {
        cout << bi.digits[i];
    }
}

BigInt BigInt_create(int num)
{
    BigInt a;
    int temp = num;
    a.n_digits = 0;
    while (temp != 0)
    {
        temp = temp / 10;
        a.n_digits++;
    }

    a.digits = new char[a.n_digits];
    for (int i = a.n_digits - 1; i >= 0; i--)
    {
        a.digits[i] = (num % 10) + '0';
        num = num / 10;
    }

    return a;
}

int BigInt_compare(const BigInt& a, const BigInt& b)
{
    //first comparing the "size" of the number because if number size is bigger so it is must be bigger!!!
    if (!(a.n_digits == b.n_digits))
    {
        if (a.n_digits > b.n_digits)
            return 1;
        else
            return -1;
    }

    for (int i = 0; i < a.n_digits; i++)
    {
        if (!(a.digits[i] == b.digits[i]))
        {
            if (a.digits[i] > b.digits[i])
                return 1;
            else
                return -1;
        }
    }
    return 0;
}

```

```cpp
void BigInt_assign(BigInt& a, const BigInt& b)
{
    BigInt_destroy(a);
    a.digits = new char[b.n_digits];
    a.n_digits = b.n_digits;
    for (int i = 0; i < b.n_digits; i++)
        a.digits[i] = b.digits[i];
}

void BigInt_destroy(BigInt& bi)
{
    delete(bi.digits);
}


```