```
1 #define _CRTDBG_MAP_ALLOC
2 #include <stdlib.h>
3 #include <crtdbg.h>
4
5 #ifdef DEBUG
6 #ifndef DBG_NEW
7 #define DBG_NEW new ( _NORMAL_BLOCK , __FILE__ , __LINE__ )
8 #define new DBG NEW
9 #endif
10 #endif // _DEBUG
11 //-----
12 #include <iostream>
13 using namespace std;
14 #include "BigInt.h"
15 #include <string.h>
16 //-----
17
18 BigInt BigInt_create(const char* s)
19 {
20
       BigInt a;
       //checking how much sfarot realy in the number (00042 -> number of
21
         sfarot is 2!)
22
       a.n_digits = no_leading_0_len(s);
23
       if (a.n_digits == 0)
24
           a.digits = nullptr;
25
       else
26
       {
           a.digits = new char[a.n_digits];
27
28
           int index = strlen(s) - a.n_digits;
           // assigning the number to array with the sfarot
29
30
           for (int i = 0; i < a.n_digits; i++)</pre>
31
              a.digits[i] = s[index];
32
              index++;
33
34
           }
35
       }
36
       return a;
37 }
38
39 //checking how much sfarot realy in the number (00042 -> number of sfarot →
     is 2!)
40 int no_leading_0_len(const char* s)
41 {
42
       int counter = 0;
43
       while (s[counter] == '0')
44
       {
45
           counter++;
46
       int eff len = strlen(s) - counter;
47
48
       return eff_len;
49 }
50
51 void BigInt_print(const BigInt& bi)
```

```
...pos\00P_LAB03_14.11.19\00P_LAB03_14.11.19\BigInt_1.cpp
```

```
2
```

```
52 {
 53
         for (int i = 0; i < bi.n_digits; i++)</pre>
 54
 55
             cout << bi.digits[i];</pre>
 56
 57 }
 58
 59 BigInt BigInt_create(int num)
 60 {
 61
         BigInt a;
 62
         int temp = num;
 63
         a.n_digits = 0;
 64
         while (temp != 0)
 65
 66
             temp = temp / 10;
 67
             a.n_digits++;
 68
         }
 69
 70
         a.digits = new char[a.n_digits];
 71
         for (int i = a.n_digits - 1; i >= 0; i--)
 72
 73
             a.digits[i] = (num % 10) + '0';
 74
             num = num / 10;
 75
         }
 76
 77
         return a;
 78 }
 79
 80 int BigInt_compare(const BigInt& a, const BigInt& b)
 81 {
         //first comparing the "size" of the number because if number size is
 82
           bigger so it is must be bigger!!!
 83
         if (!(a.n_digits == b.n_digits))
 84
 85
             if (a.n_digits > b.n_digits)
                 return 1;
 86
 87
             else
 88
                 return -1;
 89
         }
 90
 91
         for (int i = 0; i < a.n_digits; i++)</pre>
 92
 93
             if (!(a.digits[i] == b.digits[i]))
 94
             {
95
                 if (a.digits[i] > b.digits[i])
 96
                      return 1;
 97
                 else
 98
                     return -1;
 99
             }
100
         }
         return 0;
101
102 }
103
```

```
...pos\00P_LAB03_14.11.19\00P_LAB03_14.11.19\BigInt_1.cpp
```

```
104 void BigInt_assign(BigInt& a, const BigInt& b)
105 {
106
        BigInt_destroy(a);
107
        a.digits = new char[b.n_digits];
        a.n_digits = b.n_digits;
108
109
        for (int i = 0; i < b.n_digits; i++)</pre>
            a.digits[i] = b.digits[i];
110
111 }
112
113 void BigInt_destroy(BigInt& bi)
114 {
115
        delete(bi.digits);
116 }
117
118
```