Programowanie Objektywne (angielsku)

Due: May 5th 2009

The purpose of this exercise is that you understand how to build data structures that use internal allocation and deallocation. Consider the following declaration:

```c
struct digit
{
    char n;
    digit* next;
};
```

```c
class bigunsigned
{
    digit* d;

    bigunsigned( );
        // Default constructor constructs number 0.

    bigunsigned( unsigned x )
        // Constructs x as big number.

    bigunsigned( const bigunsigned& x );
        // Copy constructor.

    "bigunsigned( );

    void operator = ( const bigunsigned& x );
};
```

The exercises must be made in two files `bignum.h` and `bignum.cpp`. Declarations must be in `bignum.h`, and implementations must be in `bignum.cpp`.

1. Write the operators listed above (constructors, copy constructor, assignment, destructor)

2. Test the standard operations of the previous task for memory leaks. Check that assignments of form `i = i` are unproblematic.
3. Implement the operators +, -, *. You can use the solutions from Task list 3 as starting point.

4. Implement the operators ++, --, both as postfix and as prefix operator. You may use the operators +,- of the previous task.

5. Implement the operators <,>,<=,>=,!=, ==. (Do it in the same way as with date. Implement a single function
   int compare( const bignum& i1, const bignum& i2 ), which is used by all the other comparison operators)

6. Implement

   std::ostream& operator << ( std::ostream&, const bigunsigned& );

7. Use the previous to compute some big number of your choice, for example $2^{100}$, or $69!$. 