INFSCI 0020 Program Design and Software Tools, Spring-05
Homework 4
Due before Midnight Feb 27, Sunday

Total Score 130 (100 + 30 Points Extra Credit):

This exercise is a modification of the exercise 6.15. For this exercise you will create a class HugeInteger that uses a 40-element array of digits to store integers as large as 40 digits each. To implement the class, do the following:

- Create a variable to indicate whether the number is *positive* or *negative*
- Overload the <<, >>, +, -, and * operators for this class. Note that you do not have to define the functions *input, output, add, substract,* and *multiplication* as per exercise 6.15 – these overloaded operators will replace them.
- Overload the *pre-increment* and *post-increment* operators, and *pre-decrement* and *post-decrement* operators;

  Overflow problem **-** Note that in all arithmetic operations you could get a result that is 41 digits. In such a case, give out an appropriate message instead of doing the operation.

- Overload the assignment operator (=);
- Include *encryption* functionality as follows:
  - Define a private member int Key
  - Define public functions void setKey(int) and int getKey() to modify the values.
  - Define public functions encrypt() and decryt(), which work as follows

    encrypt() - use the Key value to circular shift the digits to the right
    decrypt() - use the Key value to circular shift the digits to the left

- Provide the following constructors:
  - A *default constructor* that accepts an integer value (defaults to 0) to initialize the new object.
  - A constructor that accepts a string of digits that is used to initialize the new object.
  - a *copy constructor* that accepts an existing huge integer value to initialize the new object;

  Note that you need to appropriately handle the member variable Key in each of these constructors.

Provide appropriate user interface for testing your program.

For Extra Credit:

- Remove the 40 digit restriction.
- There should not be an *overflow problem*