CSCI-20 Assignment #1 – Linked Lists. (100 points, due Feb. 9, 2012 in lab)

Create a linked list to implement an arbitrary-precision unsigned integer class HUGE_INT. Create the linked list the simple way, without bothering with carrier elements in the list. Just create a struct that holds the digit and the next pointer, and manage this inside the HUGE_INT class. HUGE_INT will also need to keep the size (how many decimal digits are in the number), and a print width (defaulting to the size of the number).

For the HUGE_INT class, implement normal (unsigned long int parameter), copy and default (set to 0) constructors, a method to assign an arbitrary unsigned long int to the HUGE_INT (which the constructors may use), a method to add two HUGE_INTs returning a new HUGE_INT with the result, a method to multiply two HUGE_INTs returning a new HUGE_INT with the result, a method to return the number of decimal digits in the HUGE_INT, an assignment = operator, and a destructor. You may use operator functions for the addition and multiplication methods if you wish to, but it is not required. You MAY NOT use an array or a vector to hold the digits of your arbitrary-precision integer. You MUST use a singly-linked list with a head pointer only (no tail pointer).

Also, implement a SetPrintSize() method to specify the print width (pad to the left with spaces, to align the numbers in columns), and a Print() or operator <<() method to print the number using the previously specified width (or the size of the number if it is longer than the specified width). You may add other methods if needed for your design.

The Fibonacci series: 1, 1, 2, 3, 5, 8, 13, 21, 34… begins with the terms 1 and 1, and has the property that each successive term is the sum of the two preceding terms. For example, Fib(1) is 1, Fib(2) is 1and Fib(7) is 13. (Here, we are counting the values in the series from 1. Counting from 0, the series would start 0, 1, 1, 2, etc.)

Write two clients (DO NOT use a single client for both tests):

1) A program to calculate and print Fibonacci numbers 1 through 100 (the first 100 numbers in the series). This will (far) overflow the value that can be held in an unsigned long int.
2) A program to calculate and print the first 100 factorials (n!). The factorial (n!) is defined as the product n * (n-1) * … * 2 * 1. You can express this recursively as fact(1) = 1, fact(n, n > 1) = n * fact(n-1).

For both clients, keep your numbers in an array of 100 HUGE_INTs and use the size of the largest (last) number to print the values in a right-justified column down the page. Do not use recursive approaches to the factorial and Fibonacci functions. They will be considerably harder to implement (and slower running) recursively.

Email me your source for the library class file, the header and the two test programs, as well as the output from your test runs. I want to see your code in lab and discuss it with you before you turn it in, at least a few days before it's due.

Hints:
This is easier to do if the digits are kept right-to-left in the list, e.g., the least significant digit is at the head of the list.
To print, print any spaces you need to align first, and then print the digits in reverse order.
Work out the decimal algorithms you learned in elementary school for addition and multiplication by hand: the algorithms map directly to the code.

The 100^{th} Fibonacci number is big:
354224848179261915075
100! is really big:
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000000
You can check your output values easily on the Web.