Fourth programmin exercise: Really large numbers

The exercise

Your goal is to make a C++ program that lets a user calculate things with very large numbers through a simple menu. The large numbers are represented by a doubly-connected pointer-list: So we are finally going to use pointers! The list consists of cells that contain two pointers to the previous and next cell. It also has a integer in which *k* digits of an integer are stored. *k* is a constant in this case, with a value between 1 and 9. We are basically working in a 10k-number system. For example, if k=2 the number 23056007008 is stored as 2 30 56 0 70 8.
In the menu the user can manipulate three large numbers, say A, B and C. (hint: use an array with the three large numbers!). The user can enter each of the three numbers, print them, add two together or multiply two to the third, calculate a fibonacci-number in any of them, or even sort the cells of a single large number. For example: put 123056007008 in B, the 567th Fibonacci number in C and their product in A.
The menu is very comparable to that of the third programming exercise. You can even reuse a lot of your code!

The idea is to make a class that represents the largeNumber, with member-variables:

1. **A pointer to the beginning of the sequence of cells**
2. **A pointer to the end of the sequence of cells**
3. **An int with the number of cells that are used.**

The cells (a struct with two pointers and an int) are doubly-connected: every cell is connected with a pointer to the previous, and another pointer to the next cell.

The class largeNumber should have at least the following functions:

1. **print: Print the largeNumber on the screen**
2. **read: read a large number from the keyboard (what happens when someone doesn't enter a k-fold number you should decide for yourself.**
3. **add(largeNumber1, largeNumber2): this large number should become the sum of the two large numbers largeNumber1 and largeNumber2.**
4. **fibonacci(n): The largeNumber becomes the n-th fibonacci number (with n < 10000). The first two fibonacci numbers are 1, and the sum of the i-th and i+1th is the i+2nd.**
5. **bubbleSort: Sort the cells with bubblesort (search for this online). Stop when in one iterations nothing has changed! Make sure that you actually swap the cells around, not only the numbers in them!!!! The example number should become 2 8 30 56 70, printed as 208305670.**
6. **multiply(largeNumber1,largeNumber2): the large number becomes the product of these**

two large numbers.

7. addBehind: Add a cell at the end of the existing large number cell sequence (pay attention to what should happen if this is the first cell!!)
8. makeOne: the largeNumber becomes the number 1, represented with only 1 cell.
9. makeZeros(m): make a 'largeNumber' existing of m cells with a 0 in them.
10. destroy: Delete all the cells that are used by this largeNumber.

The first six functions are listed in terms of difficulty. Also make the last 4 helper functions to make your programming easier. Make as many other helping functions you think you need, like copying etc. Don't forget to make a constructor for your class!

If k > 4, it is possible that when performing multiplication the resulting number becomes larger than INT_MAX. Temporarily use a long long int for multiplication, that should surely solve the problem.

Put all the code, classes, functions and structs in one file.

For those well-versed in class-brewing, define your own *, + and = operators for the largeNumbers (read the designated part of chapter 8). This means that you can calculate with largeNumbers as if they were simple ints or doubles! c += a*b etc.

Opmerkingen

Use proper member-functions. For this exercise as well, none of your functions should be longer than 20 lines! Every functions needs to have proper comments, explaining their purpose and use. You only need to use the iostream header file. Rough indication of the C++ program: 400 regels. Also include an information block at the start.