2025.2 Object Oriented Programming, Project #2

(Due : November 2 (Sunday), 11:59pm)

Submission Rule

1. Create a directory "proj2" and its subdirectories "prob1" and "prob2" in "proj2"
2. Insert compilable source code package(inf_int.h, inf_int.cpp, main.cpp, ...), readme.txt, and demo video file (.mp4 format, 1 or 2 minutes) showing execution of the test program for Problem1 into "prob1". Insert source code package(inf_int.h, inf_int.cpp, main.cpp, ...), readme.txt and demo video file (.mp4 format, 1 or 2 minutes) showing execution of the program for Problem2 into "prob2". In readme.txt file, you should briefly explain how to compile and execute your code.
4. zip the directory "proj2" into proj2.zip and submit the zip file into eClass homework board.

Problem1. Complete implementing `inf_int` class that represents infinite precision integer and its operations. The specification of `inf_int` is provided to you in "`inf_int.h`". You should add its implementation in "`inf_int.cpp`". Test the correctness of your implementation by writing "`main.cpp`". "`inf_int.h`" and an example of "`main.cpp`" are downloadable from our class website. After compilation, your code should generate correct result. The result of this problem should be submitting compilable source code package (`inf_int.h`, `inf_int.cpp`, `main.cpp`, `.sln` files (files for opening with visual studio 2010 or 20xx)). FYI, I will test your code by replacing the "`main.cpp`" with my own `main.cpp` files that contain various test cases and checking whether your code generates correct result according to my own `main.cpp`. You have to effectively use dynamic memory allocation(`new`) and deallocation(`delete`) for handling the variable sizes of infinite precision integer numbers.

Problem2. Write a calculator program using the C++ files (`inf_int.h` and `inf_int.cpp`) you made for problem1. Your calculator program should support addition(+), subtraction(-), and multiplication(*) of infinite precision integers. Your calculator should follow input and output formats as shown below.

Keyboard Input Format : After executing your program on command line, your program prints "Input:" and wait for a user keyboard input. The user keyboard input should be an expression that has a format like :

(positive integer)(space)(operator)(space)(positive integer)

After a user finishes the keyboard input and presses an enter key, your program should print the calculation result of the input expression. The program repeat this input and output processes. The program is terminated when 0 is given as an input.

Execution Example:

```
> prob2.exe
Input: 345 - 11111111111111111111111111111111111111111111111111111
Output: -11111111111111111111111111111111111111111111111110766
Input: 333333333333333333333 * 73755
Output: 24584999999999999999999975415
Input: 25555555555551111 + 2111111111
Output: 25555557666662222
Input: 0

>
```