MMCC '15 P3 - Momoka

Time limit: 1.0s **Memory limit:** 768M

Momoka Oginome is fond of her diary. Her diary can create miracles! All she has to do is chant the magic spell written in the diary.

Unfortunately, such magic comes at a price. After changing fate, the user of the spell will get injured (and possibly burst into flames). Momoka decides to look for a way to remove the curse. She thinks she's onto something! To investigate further, Momoka will type the contents of her diary into a word processor, and after each keystroke she wants to know the number of palindromes in the string she has typed so far. Sometimes, she makes mistakes and presses backspace, which will remove the last letter that was added to the string.

Implementation Details

You should write a program which implements the advanced word processing software with palindrome detection capabilities. Locally, your program should include the header file `momoka.h` by `#include "momoka.h"`. When grading your submission online, the judge will automatically prepend this line to your file.

Your program should implement the following procedures.

- `void Init()` This procedure is called only once in the beginning. Use it to perform any initialization your program might need.
- `long long Type(char letter)` This procedure is called when Momoka types a letter. It is guaranteed that `letter` will be a lowercase English letter between `a` and `z`. You should return the number of palindromes in the string so far.
- `void Backspace()` This procedure is called when Momoka wants to erase the last character added by `Type`. It is guaranteed that there is at least one letter in Momoka's string when this procedure is called.

You can find a sample grader in the C language here.

You can find the header `momoka.h` here.

You can find sample input for the sample grader with Windows-style line endings here.

Constraints

All input data satisfy the following conditions.

- The total number of calls to `Type` and `Backspace` is less than or equal to $1\,000\,000$.

Note

The grader will generate correct input and output dynamically. This means that the time and memory on the submission results page may not necessarily be solely from your program's execution.