Practice Arena

Practice problems aimed to improve your coding skills.

- PRACTICE-02_SCAN-PRINT
- PRACTICE-03_TYPES
- LAB-PRAC-02_SCAN-PRINT
- LAB-PRAC-01
- PRACTICE-04_COND
- BONUS-PRAC-02
- LAB-PRAC-03_TYPES
- PRACTICE-05_COND-LOOPS
- LAB-PRAC-04_COND
- LAB-PRAC-05_CONDLOOPS
- PRACTICE-07_LOOPS-ARR
- LAB-PRAC-06_LOOPS
- LAB-PRAC-07_LOOPS-ARR
- LABEXAM-PRAC-01_MIDSEM
- PRACTICE-09_PTR-MAT
- LAB-PRAC-08_ARR-STR
- PRACTICE-10_MAT-FUN
- LAB-PRAC-09_PTR-MAT
- LAB-PRAC-10_MAT-FUN
  - Stack
  - The Prutor Editor
  - Finding your identity
  - Queue
  - The Prutor Editor Part II
  - Only Ones
  - Graphs
  - How Mr C actually does Math
  - The Hidden Positives and Negatives
  - How Prutor Manages Memory
  - Message in the Matrix
  - The Hidden Key
- PRACTICE-11_FUN-PTR
- LAB-PRAC-11_FUN-PTR
- LAB-PRAC-12_FUN-STRUC
- LABEXAM-PRAC-02_ENDSEM
- LAB-PRAC-13_STRUC-NUM
- LAB-PRAC-14_SORT-MISC

The Prutor Editor Part II

LAB-PRAC-10_MAT-FUN

The Prutor Editor Part II [20 marks]

Problem Statement
We have been using Prutor for a long time now. The Prutor has an excellent editor where we enter our code and edit it. We can enter visible or whitespace characters, move left and right using arrow keys within and across different lines of our code, as well as use delete and backspace to delete characters. In this problem, we will implement a very very simple text editor.

The first line of your input will give you a string with at most 99 characters. The string will be presented in a single line and may contain alphabet characters (upper and lower case), digits 0-9 and spaces. The next lines will each contain an instruction to either move the cursor left or right or delete a character in one of two ways described below.

Try to use the gets function to read in the string.

1. Initially the string will be what is give in the input and the cursor will be at the very end of the string i.e. if the string has n characters (n < 100), then the cursor will be at index n.
2. If the instruction is the character '<' (without quotes), move the cursor one index to the left. If the cursor is already at index 0, it stays there.
3. If the instruction is the character '>' (without quotes), move the cursor one index to the right. If the cursor is already at the end of the current input, it does not move.
4. If the instruction is the character ')' (without quotes), perform the "delete" operation i.e. delete the character at the cursor position. The cursor position stays the same. The length of the string decreases by one after this operation. If the string is already empty or if the cursor is at the end of the input, the delete operation does nothing.
5. If the instruction is the character '(' (without quotes), perform the "backspace" operation i.e. delete the character just before the cursor position. The cursor index decreases by one. The length of the string decreases by one after this operation. If the string is already empty or if the cursor is at index 0, the backspace operation does nothing.
6. If the instruction is the character '#', no more operations to perform. Do not print the string and simply exit the program.

After every delete or backspace operation i.e. when the character is not < or >, print the current state of the string on a separate line after performing the said operation.

Caution

1. We assure you that the total length of the string will never exceed 99 characters.
2. We will not penalize you for extra newlines at the end of your output. However, do not have extra newlines in the middle of your output or else have trailing spaces in any line of the output.
3. Be careful about extra/missing lines and extra/missing spaces in your output.

EXAMPLE:

INPUT
ABCD
)
(
<
(
#

OUTPUT:
ABCD
ABC
AC

Explanation: The following explains the location of the cursor and the reasoning after every instruction. The location of the cursor is denoted using the pipe symbol.

1. Initially the string is ABCD|
2. Instruction ')': No effect since cursor at the end already. String is still ABCD|
3. Instruction '(': Backspace. String is ABC|
4. Instruction '<': Move cursor left. String is AB|C
5. Instruction '(': Backspace. String is A|C
6. Instruction '#': Terminate

Grading Scheme:
Total marks: **[20 Points]**

There will be partial grading in this question. There are several lines in your output. Printing each line correctly, in the correct order, carries equal weightage. Each visible test case is worth 2 points and each hidden test case is worth 4 points. There are 2 visible and 4 hidden test cases.

Please remember, however, that when you press Submit/Evaluate, you will get a green bar only if all parts of your answer are correct. Thus, if your answer is only partly correct, Prutor will say that you have not passed that test case completely, but when we do autograding afterwards, you will get partial marks.

Start Solving! (/editor/practice/6201)