

Distributed Round 2 2016

[A. Testrun](#)[B. again](#)**C. lisp_plus_plus**[D. asteroids](#)[E. gas_stations](#)[Contest Analysis](#)[Questions asked](#) 3

Submissions

Testrun

0pt	Not attempted 0/74 users correct (0%)
-----	--

again

1pt	Not attempted 401/409 users correct (98%)
14pt	Not attempted 368/399 users correct (92%)

lisp_plus_plus

3pt	Not attempted 390/399 users correct (98%)
17pt	Not attempted 355/385 users correct (92%)

asteroids

5pt	Not attempted 283/305 users correct (93%)
25pt	Not attempted 91/170 users correct (54%)

gas_stations

8pt	Not attempted 191/233 users correct (82%)
27pt	Not attempted 28/95 users correct (29%)

Top Scores

eatmore	100
Marcin.Smulewicz	100
tozangezan	100
Errichto.rekt	100
mnbvmar	100
qwerty787788	100
sevenkplus	100
tczajka	100
fhlasek	100
wata	100

Problem C. lisp_plus_plus

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the [Quick-Start Guide](#) to get started.

small 3 points 2 minute timeout	The contest is finished.
---------------------------------------	--------------------------

large 17 points 10 minute timeout	The contest is finished.
---	--------------------------

Problem

Lisp++

Alyssa is a huge fan of the Lisp programming language, but she wants it to have even more parentheses, so she is designing a new language, Lisp++. A valid Lisp++ program consists of one of the following. (In this specification, P stands for some valid program -- not necessarily the same program each time.)

- $()$ Literally, just an opening parenthesis and a closing parenthesis.
- (P) A program within a pair of enclosing parentheses.
- PP Two programs (not necessarily the same), back to back.

Alyssa is working on a compiler for Lisp++. The compiler must be able to evaluate a string consisting of $($ characters and/or $)$ characters to determine whether it is a valid Lisp++ program, and provide the user with some helpful information if it is not. If the program is valid, the compiler should print -1. Otherwise, it should print the length of the longest prefix that could be extended into a valid program by adding zero or more additional characters to the end. If that prefix is the empty prefix, the compiler should print 0. In particular, if the input string is not a valid program, but can be extended to a valid program, the compiler should print the length of the input string.

For example:

- $((()()))$ is a valid program, so the compiler should print -1.
- $((()))$ is not a valid program. The prefix $((()))$ is the longest prefix that is or could be extended into a valid program (in this case, it already is one), so the compiler should print 4. The only longer prefix is $((()))$ (i.e., the entire string), but there is no way to add (any number of) characters to the end of that string to make it into a valid program.
- $)$ is not a valid program. The prefix $)$ cannot be extended into a valid program. The empty prefix is not a valid program, but it can easily be extended into one (by adding $()$, for example). So the compiler should print 0.

Given a string, what should Alyssa's compiler print?

Input

The input library is called "lisp_plus_plus"; see the sample inputs below for examples in your language. It defines two methods:

- **GetLength():**
 - Takes no argument.
 - Returns a 64-bit integer: the length of the string.
 - Expect each call to take 0.07 microseconds.
- **GetCharacter(i):**
 - Takes a 64-bit integer in the range $0 \leq i < \text{GetLength}()$.
 - Returns a character: either $($ or $)$, the character at position i .
 - Expect each call to take 0.07 microseconds.

Output

Output a single line with a single integer: what the compiler should print, as described above.

Limits

Time limit: 3 seconds.
Memory limit per node: 128 MB.
Maximum number of messages a single node can send: 1000.
Maximum total size of messages a single node can send: 8 MB.
GetCharacter(i) is always $($ or $)$.

Small input

Number of nodes: 10.
 $1 \leq \text{GetLength}() \leq 10^6$.

Large input

Number of nodes: 100.
 $1 \leq \text{GetLength}() \leq 10^9$.

Sample

Input	Output
See input files below.	For sample input 1: -1 For sample input 2: 4 For sample input 3: 0

The sample input files are the same examples from the problem statement, in the same order.

Sample input libraries:

Sample input for test 1: [lisp_plus_plus.h](#) [CPP] [lisp_plus_plus.java](#) [Java]

Sample input for test 2: [lisp_plus_plus.h](#) [CPP] [lisp_plus_plus.java](#) [Java]

Sample input for test 3: [lisp_plus_plus.h](#) [CPP] [lisp_plus_plus.java](#) [Java]

All problem statements, input data and contest analyses are licensed under the [Creative Commons Attribution License](#).

© 2008-2017 Google [Google Home](#) - [Terms and Conditions](#) - [Privacy Policies and Principles](#)

Powered by



Google Cloud Platform