(5) [15 points]

Imagine that you are writing a simple game for young children to learn to recognize and match shapes. For our purposes, shapes are restricted to squares and circles. Shapes arrive over time and fall on a pile of items. See Figure 1.

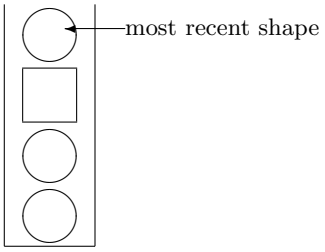

Figure 1: An example of a game state.

At each point, the child may press one of two buttons, 'L' and 'R'. 'L' should be the button to press if there is a square on top, and 'R' if there is a circle on top. More precisely, the rules are as follows:

- If the child presses the correct button ('L' when the top shape is a square, 'R' when it is a circle), the child gets a point, and the top shape disappears.
- If the child presses the wrong button ('L' when the top shape is a circle, 'R' when it is a square), the child loses a point (the points could become negative), and no shape disappears.
- If the child presses a button when the pile is empty, nothing happens.
- The game starts at 0 points.
- Items appear at a regular pace, and their appearance has nothing to do with what buttons are pressed.
- The game may end even when there are still items left.

You will be given a sequence consisting of the letters 'L', 'R', 'S', 'C' (representing the pressing of the two buttons, and the arrival of the two shapes, respectively), and are to determine from it the final score of the child. We promise that the sequence will contain only those four letters, and nothing else.

As an example, for the sequence "SCLLR", the final score will be -1, and there will be a square left in the end. The two presses of the 'L' button are wrong, so each incurs a penalty of 1 point and changes nothing. The press of 'R' is correct, so it earns a point and makes the circle disappear, but the square remains.

Insert your code in the following:

```
#include <iostream>
using namespace std;

int main ()
{
   string s;   // the sequence of characters describing the game's events
   int score = 0;
   cin >> s;

   // your code goes here.









   cout << "The score is " << score << endl;
   return 0;
}
```