2002 Sun Microsystems and TopCoder Collegiate Challenge – Problem Statement

Nested PROBLEM STATEMENT

Programming environments commonly have features to indicate parenthesis and bracket matching when editing a program. Even with this functionality, however, finding bugs caused by a missing or an extra symbol are often difficult to find. To assist your work you decide to write a preprocessor that analyzes the parentheses and brackets of a program to determine the most likely cause of symbol mismatch. In a correct program, the sequence of parentheses and brackets should form a balanced string, nested properly. In other words there is a one-to-one relationship between open parentheses and closed parentheses, with a similar relationship for brackets. For each pair of parentheses and brackets, the open symbol should come before the close symbol and the substring between the open and closed symbol should also be balanced. In other words,

```
Balanced_String = Balanced_String Balanced_String
                | '(' Balanced_String ')'
                | '[' Balanced_String ']'
                | ""
```

Examples of balanced strings: "", "[]", "()[]", "([])()"
Examples of non-balanced strings: "())", "(]", "([)]"

DEFINITION
Class Name: Nested
Method Name: makeBalanced
Parameters: String, int, int
Returns: int
Method signature (be sure your method is public): int makeBalanced(String data, int insCost, int delCost);

If you are allowed to insert parentheses and brackets anywhere in the string at cost insCost or delete parentheses and brackets at cost delCost, determine the minimum cost of operations that will yield a balanced string.

TopCoder will ensure the validity of the inputs. Inputs are valid if:
- 0 <= insCost <= 1000
- 0 <= delCost <= 1000
- length(data) <= 50
- data is composed entirely of '(', ')', '[', and ']'

EXAMPLES

"()[]", 3, 2: makeBalanced should return 0 (the String is balanced)
"([)]", 1, 2: makeBalanced should return 2 (insert two parentheses to get "()[()]")
"(((]))", 1, 1: makeBalanced should return 2 (delete the ']' and add a ')' to get "((()))")
"[)][)()(])", 2, 1: makeBalanced should return 4
"[()]]))((]])(((]][])", 10, 1: makeBalanced should return 10
"(((((((((((((((((((((((((]]]]]]]]]]]]]]]]]]]]]]]]]", 5, 6: makeBalanced should return 250
"(]][))(]][))(]][))(]][))(]][))(]][))(]][))(]][))(]", 14, 99: makeBalanced should return 476