Práctica: Balanceo de paréntesis.

Objetivos.

- Aprender a utilizar el TAD Stack para resolver un problema.

Descripción.

Desarrollar un algoritmo que compruebe que los paréntesis están balanceados en un texto (asume el juego de caracteres ASCII de 8 bits).
Los tipos de paréntesis a comprobar son ‘(‘, ‘[‘, ‘{‘, los cuales concuerdan con los paréntesis cerrados correspondientes ‘)’, ‘]‘ y ‘}’.
El programa mostrará “Correcto” si el texto está balanceado o la posición del carácter correspondiente al primer paréntesis que está mal balanceado. Se dará prioridad a mostrar el primer paréntesis cerrado no balanceado, por ejemplo 3 para el caso “()}” o sino el primer paréntesis abierto no balanceado, por ejemplo 1 para el caso “({}”.
Para resolver este problema se utilizará una pila en la que se apilarán los paréntesis abiertos encontrados y serán desapilados conforme encontremos paréntesis cerrados que concuerden con la cima de la pila. Cuando se apila un paréntesis abierto, se debe añadir el posición del carácter correspondiente para tener esa información si al final ese paréntesis no está balanceado.
La pila será implementada usando una lista simple. La implementación dinámica o acotada (la profundidad máxima será 100) es de libre elección por el alumno.
Algunos ejemplos de ejecución serían:

Caso 1:
Entrada: “”
Salida: “Correcto”

Caso 2:
Entrada: “()”
Salida: “Correcto”

Caso 3:
Entrada: “[]()”
Salida: “Correcto”

Caso 4:
Entrada: “{[]}”
Salida: “Correcto”

Caso 5:

Entrada: “(}”
Salida: 2
Se da prioridad a los paréntesis cerrados.

Caso 6:
Entrada: “({}”
Salida: 1

Caso 7:
Entrada: “{(})”
Salida: 3

Caso 8:
Entrada: “{}[“
Salida: 3