Balance Parentheses

Stephen Gerkin
November 12, 2019
CIST 2362 CRN 22282

Program Description

Implement an algorithm that tests a string for the balance of parentheses in a string. A string is considered balanced when each left opening parenthesis is matched with a closing right parenthesis. A string containing no parentheses is considered balanced.

Programming Strategy

A string input will be received from the user to check for balance. The string will then be iterated over. At each character, if it is an opening parenthesis, this will be added to a stack. At each closing parenthesis, the stack will be checked to see if there is an opening parenthesis to match. If there is, the last entry to the stack will be removed. If the character in the string is not an opening or closing parenthesis, it will be ignored. Once the string iteration has been complete, it will be determined if the stack is empty or not. An empty stack signifies that the string is balanced, whereas a non-empty stack indicates that the string is not balanced.

The LinkedList implementation created for the previous lab will be used as it contains methods for using the List class as a stack or a queue.

StringChecker Class Documentation

Main class that contains the method for checking if a string of is balanced with regards to parentheses.

This is a utility class and not meant to be instantiated. Additionally, contains rudimentary unit testing that verifies that the algorithm used to verify a string is working as intended.

Member Data Documentation

Access	Modifier(s)	Data type	Name	Description
private	static const	char	LEFT	For holding the character value for a left parenthesis.
private	static const	char	RIGHT	For holding the character value for a right parenthesis.

Constructor and Destructor

Default constructor is deleted as this is a utility class and not meant to be instantiated. Destructor is default.

Member Functions

static bool StringChecker::isBalanced (const std::string & *str*) [inline], [static], [noexcept]

Iterates over a string to determine if the parentheses are balanced.

At each character, if the character is an opening parenthesis, it is added to a stack. If the character is a closing parenthesis, we attempt to get the most recent character added to the stack. If the stack is empty, the string is not balanced and we exit early. If the character on top of the stack is an opening parenthesis, we continue to iterate. If the character in the string is not an opening or closing parenthesis, we continue iterating and do nothing. Finally, once we are done iterating through the string, if we have not already exited early, we return whether or not the stack is empty. If it is empty, the string is balanced. Otherwise, it is not.

Parameters

str	The string we want to check.

Returns

True if the string has balanced parentheses (or does not contain any).

static bool StringChecker::performTest (const std::string & *str*, bool *expectedResult*) [inline], [static], [private]

Helper for performing a test.

Prints to screen status of test.

Parameters

str	The string we want to test against.
expectedResult	The result we expect from the string.

Returns

Whether or not the test passed.

static void StringChecker::test_isBalanced () [inline], [static], [noexcept]

Rudimentary unit test for **isBalanced()**.

Tests several good test cases and several bad test cases.

Input and Output Demonstration

Unit tests

```
"D:\School\2019 - IV - Fall\CIST2362 - C++ II\Programs\10\BalanceParentheses\cmake-build-debug\BalanceParentheses.exe"
Test on: "()()()()()()()" ...
Pass.
Test on: "(((((((())))))))" ...
Pass.
Test on: "((()(()))" ...
Pass.
Test on: "a" ...
Pass.
Test on: "(a)" ...
Pass.
Test on: "(a((a)a)a)" ...
Pass.
Test on: "(
)" ...
Pass.
Test on: "((((()))" ...
Pass.
Test on: "()()()(" ...
Pass.
Test on: ")(" ...
Pass.
Test on: "(a(" ...
Pass.

Process finished with exit code 0
```

Start message

```
This program will test for parentheses balance in a string
Balanced is achieved when every right, closing parenthesis ')'
is preceded by a left, opening parenthesis '('
and every opened parenthesis is closed.
For example, ()() or (()) is balanced but )( or (() is not.
Enter a string to test for balance (to quit type "quit"):
>>
```

Balanced input

```
>> ()()

Yes, that string is balanced.
Enter a string to test for balance (to quit type "quit"):
>>
```

Unbalanced input

```
>>  (()

No, that string is not balanced.
Enter a string to test for balance (to quit type "quit"):
>>
```

Empty input

```
>>

Yes, that string is balanced.
Enter a string to test for balance (to quit type "quit"):
>>
```

Quit entered

```
Enter a string to test for balance (to quit type "quit"):
>> quit


Process finished with exit code 0
```