3D1 Microprocessor Systems
Balanced parentheses
Assignment 6
Due by: Week 12

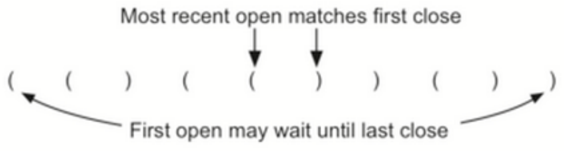
Design and write an ARM Assembly Language program that will determine if the parentheses in a string are correctly balanced. Balanced parentheses means that each opening symbol has a corresponding closing symbol and the pairs of parentheses are properly nested. Consider the following correctly balanced strings of parentheses:

- (()()()())
- (((())))
- (()((())()))

Compared those with the following, which are not balanced:

- ((((((())
- ()))
- (()()(()

The ability to differentiate between parentheses that are correctly balanced and those that are unbalanced is an important part of recognizing many programming language structures. The challenge then is to write an algorithm that will read a string of parentheses from left to right and decide whether the symbols are balanced. To solve this problem we need to make an important observation. As you process symbols from left to right, the most recent opening parenthesis must match the next closing symbol. Also, the first opening symbol processed may have to wait until the very last symbol for its match. Closing symbols match opening symbols in the reverse order of their appearance; they match from the inside out.


Most recent open matches first close
(( ) (( )) ( ))
First open may wait until last close

This is a clue that stacks can be used to solve the problem. Once you agree that a stack is the appropriate data structure for keeping the parentheses, the statement of the algorithm is straightforward. Starting with an empty stack, process the parenthesis strings from left to right. If a symbol is an opening parenthesis, push it on the stack as a signal that a corresponding closing symbol needs to appear later. If, on the other hand, a symbol is a closing parenthesis, pop the stack. As long as it is possible to pop the stack to match every closing symbol, the parentheses remain balanced. If at any time there is no opening symbol on the stack to match a closing symbol, the string is not balanced properly. At the end of the string, when all symbols have been processed, the stack should be empty. Use memory locations to indicate whether the test case strings are balanced or unbalanced.

Your programming assignments will be assessed based on two components:

- Written report in lab book.
- Demonstation of working code on computer.

You must show both of these to the Demonstator by the deadline for each assignment. You must buy and maintain a hardcover lab book and write a report covering each programming assignment. In your report you should:

- Include title and date on each page.
- Describe what you are attempting to do in English.
- Outline your algorithm using diagrams and pseudo code as appropriate.
- Write down the actual ARM assembly code used, including comments.
- Explain how you tested it and the results.
- Size of report should be between one and two A4 pages in length.

Assignments are only accepted in a lab book, it is not feasible to accept material on individual pieces of paper or electonic devices. Throughout this course marks are awarded for punctuality, neatness, organisation, spelling, ability to communicate technical information and results as well as assembly programming skill. Of particular importance in assembly language programming are indentation, comments and quality of label names. The lab book must be kept up to date and will be examined by the Lecturer or Demonstrator during the term. You must hand in your lab book no later than 4pm Wednesday, 16th December 2015, to the School of Computer Science and Statistics Reception, Ground Floor, The O'Reilly Institute. The office does not accept submissions after 4pm. Do not forget to write your name, student number and course clearly on the cover.