Assignment 3
Stacks

Deadline: 4 March 2022 at 23:59 on Submitty

Working individually, complete the assignment below. Submit your solution to Submitty (`https://submit.scss.tcd.ie`). By submitting your solution, you are confirming that you have familiarised yourself with College's policy on plagiarism (`https://libguides.tcd.ie/plagiarism`).

Your mark will be the auto-graded mark assigned by Submitty (10 marks) plus a manually assigned mark (5 marks) for programs that demonstrate (i) excellent presentation, (ii) helpful, concise pseudocode comments, (iii) a well-structured approach to solving the problem and (iv) make good use of the stack and the available ARM Assembly Language instructions and addressing modes for manipulating the stack.

You are allowed to submit five attempts for the assignment without penalty. Subsequent attempts will attract a 1 mark penalty each, up to a maximum penalty of 5 marks.

Submitty will allow you eight "late days" over the full semester. This means, for example, you can submit one assignment late by eight days or eight assignments late by one day (or part thereof) each, without penalty. Once your "late days" are used up, you will receive zero marks for any late submissions.

Instructions

"Reverse Polish Notation" or "postfix notation" is an alternative way of writing mathematical expressions that has the advantage of avoiding the need to use parenthesis (brackets). For example, the "infix" expression

$$(5 + 3) * (4 + 2)$$

can be rewritten using Reverse Polish Notation notation as

$$5\ 3 + 4\ 2 + *$$

Expressions using Reverse Polish Notation can also be conveniently evaluated using a stack.

Write an ARM Assembly Language program that, given a NULL-terminated ASCII string representing a mathematical expression using Reverse Polish notation, will use a stack to evaluate the expression and store the result in R0. Assume the original string is stored in memory at the address in R1.

Assume that the original string contains only digits from '0' to '9', the operators '+', '-' and '*' and spaces. Spaces are used to separate consecutive values.

Programs that work for expressions containing only single digit values, like the example above,

will attract marks up to 70%. Programs that can handle expressions containing multi-digit values, like the expression below, will attract up to 100%.

$$20\ 10 - 4\ 2 + *$$

Your program may use either the system stack or your own stack. If you use the system stack be aware that if your program does not pop off everything that it pushes on to the stack, your program will fail all Submitty tests. You may assume that your program will only be given valid expressions as input.

Hints

Process the original string one character at a time.

If you encounter a value in the string, push the value on the stack.

If you encounter an operator in the string, pop the top two values off the stack and apply the operator (add, subtract, multiply) to the two values. Push the result back on to the stack.

When you get to the end of the string, you should have just a single value on the stack. This is your result. Pop if off the stack into register R0.