MIPS Assembly Program for Evaluating an Expression in Postfix format

Aditya Goel — 2019CS10671

March 1, 2021

Program Design Specifications: Input & Output Format

- The Program lets you enter a postfix expression with constant integer operands in the range 0-9 and operators +, -, and *
- The input of the postfix expression is taken as a string which is terminated by a newline character, and any character other than what is specified above prints an error "Invalid Character Found".
- An "Invalid Postfix Expression" error is printed if the entered postfix expression is not valid, i.e., if the sequence of operators and operands is not specified correctly.
- A valid postfix expression with the result at each step within the memory bounds of a 32 bit signed integer prints the final result as "The Result is : $< answer >$".
- A postfix expression with its intermediate or final results that overflow from the memory bounds of a 32 bit signed integer, raises an Arithmetic Overflow error.

Strategy & Approach Used

- I have implemeted a stack, in order to evaluate the postfix expression. As we traverse through the given string of postfix expression, character by character :
  - If the character is an operand with integer value between 0-9 , we first convert the ASCII character to the corresponding integer and then push it into our stack.
  - If the character is an operator +, - or *, we pop two operands from the stack and evaluate the operator. We push this result into our stack.
  - If the character is neither an operand with integer value between 0-9 and nor an operator +, - or *, we print an "Invalid Character Found" error.
- Finally the operation ends on detection of a newline character and if the postfix expression was valid, it evaluates to an integer value, which is then printed as a result of the given postfix expression.
- In case the given input expression is not correctly specified, then an "Invalid Postfix Expression" error is printed.

Test Cases & Corner Cases Handled

1. Entering an Empty string as the postfix expression, prints an error "Invalid Postfix Expression".
   - A test on input "" resulted in an "Invalid Postfix Expression" error.
2. Entering any invalid character, other than those specified in the program design specifications prints "Invalid Character Found" error.
   - A test on inputs such as "a3+", "b 2+", "{}35+" etc. resulted in an "Invalid Character Found" error.
3. If the sequence of operators and operands is not specified correctly, then an "Invalid Postfix Expression" error is printed.

- A test on inputs such as "3+5", "+25", "25++3" etc. resulted in an "Invalid Postfix Expression" error.

4. If for n operands in the postfix expression we do not have n-1 operators or if there remain more than 1 elements in the stack after the entire evaluation of the postfix expression, then an "Invalid Postfix Expression" error is printed.

   - A test on inputs such as "35+2", "35+2+*", "35+2-3454" etc. resulted in an "Invalid Postfix Expression" error.

5. A postfix expression with its intermediate or final results that overflow from the memory bounds of a 32 bit signed integer, raises an Arithmetic Overflow error.

   - A test on inputs such as "99*9*9*9*9*9*9*9*9*", "11+1+1+1+...2147483647times" etc. resulted in an Arithmetic Overflow error.