CS2123 Data Structures - Spring 2020

Assignment 2: Stacks

Due 2/22/20 by 11:59pm

Evaluate Postfix (10 points)

Sketch of algorithm for evaluating postfix strings:

(1) Create stack s.
(2) For each token, x, in the postfix expression:
    1 If x is T or F push it into the stack s.
    2 Else if x is a unary operator
        i pop an operand, $op1$, from s
        ii compute x $op1$ (see unary table below)
        iii push the result into s
    3 Else if x is a binary operator
        i pop an operand, $op2$, from s
        ii pop an operand, $op1$, from s
        iii compute $op1$ $op2$ x
        iv push the result into s
(3) If you do not have enough operands in s to perform an operation you should return an error in the boolean.
(4) Likewise, if s contains more than one operand after all of the tokens are evaluated you should return an error in the boolean.
(5) Otherwise pop and return the only value in s.

Unary operations:

$op1$ NOT	!$op1$

Binary operations:

$op1$ $op2$ AND	$op1$ && $op2$
$op1$ $op2$ NAND	!($op1$ && $op2$)
$op1$ $op2$ OR	$op1$ || $op2$
$op1$ $op2$ NOR	!($op1$ || $op2$)
$op1$ $op2$ XOR	$op1$!= $op2$
$op1$ $op2$ COND	!$op1$ || $op2$
$op1$ $op2$ BICOND	$op1$ == $op2$

Convert Infix to Postfix (10 points)

Sketch of algorithm for converting postfix strings to infix strings:

(1) Create stack s.
(2) For each token, x, in the postfix expression:
    1 If x is T or F push it into the stack s.
    2 Else if x is a unary operator
        i pop an operand, $op1$, from s
        ii push the string "($op1$ x)" into s
    3 Else if x is a binary operator
        i pop an operand, $op2$, from s
        ii pop an operand, $op1$, from s
        iii push the string "($op1$ x $op2$)" into s
(3) You assume that the postfix string is well formatted (feel free to implement error checking if you would like).
(4) pop and return the value in s.

Hints for memory management:

- Every string that you push into your stack should be malloc-ed.
- You should free strings after popping them (be sure to use them before freeing them).
- Operator tokens will be freed after usage since they are not put into the stack.

Deliverables:

Your solution should be submitted as "booleanEvaluation.c", "booleanEvaluation.h", and "makefile". Also attach any additional files you create to solve this problem.

Upload these file to Blackboard under Assignment 2. **Do not zip your files**.

To receive full credit, your code must compile and execute. You should use valgrind to ensure that you do not have any memory leaks.

Remember:

The program you submit should be the work of only you. Cheating will be reported to UTSA's office of Student Conduct and Community Standards. Both the copier and copiee will be held responsible.