

# CS2124 Data Structures

## Assignment 2: Stacks

Due 2/18/22 by 11:59pm

### Evaluate Postfix (10 points)

Sketch of algorithm for evaluating a postfix string:

- (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 If you do not have at least one operand in  $s$ , you should return an error in the boolean (remember to free your data).
    - ii pop an operand,  $op1$ , from  $s$
    - iii compute  $x\ op1$  (see unary table)
    - iv push the result into  $s$
    - v free  $op1$  and  $x$
  - 3 Else if  $x$  is a binary operator
    - i If you do not have at least two operands in  $s$ , you should return an error in the boolean (remember to free your data).
    - ii pop an operand,  $op2$ , from  $s$
    - iii pop an operand,  $op1$ , from  $s$
    - iv compute  $op1\ op2\ x$  (see binary table)
    - v push the result into  $s$
    - vi free  $op1$ ,  $op2$ , and  $x$ .
- (3) If  $s$  contains more than one operand after all of the tokens are evaluated you should return an error in the boolean.
- (4) Otherwise pop and return the only value in  $s$ .
- (5) Be sure to free all remaining malloced data before leaving the function (including when returning an error). Do not free the returned string.

Operator Type	Usage	Calculation
unary operator	$op1$ NOT	$!op1$
binary operator	$op1\ op2$ AND	$op1 \ \&\& \ op2$
binary operator	$op1\ op2$ NAND	$!(op1 \ \&\& \ op2)$
binary operator	$op1\ op2$ OR	$op1 \    \ op2$
binary operator	$op1\ op2$ NOR	$!(op1 \    \ op2)$
binary operator	$op1\ op2$ XOR	$op1 \ != \ op2$
binary operator	$op1\ op2$ COND	$!op1 \    \ op2$
binary operator	$op1\ op2$ BICOND	$op1 \ == \ op2$

## Convert Postfix to Infix (10 points)

Sketch of algorithm for converting a postfix strings to an 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 “( $x$   $op1$ )” into  $s$
    - iii free  $op1$  and  $x$
  - 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  $s$
    - iv free  $op1$ ,  $op2$ , and  $x$
- (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  $s$ .
- (5) Be sure to free all remaining malloced data before leaving the function (including when returning an error). Do not free the returned string.

### Hints for memory management:

- Every string that you push into your stack should be malloced. Using the provided function “booleanToString” from the file “booleanWithError.c” will make this easier. It also a couple other handy functions that you should consider using.
- I would recommend not using “strtok” to tokenize your string (it will require significantly more work on your part). Instead you should use the “tokenizeString” provided in the assignment.
- You should free strings after popping them from your stack (be sure to use them before freeing them).
- Operator tokens aren’t pushed in the stack and can be freed immediately after usage.
- Be sure to free all of your malloced data before returning. This included freeing your data in the case of an error.
- Remember to also free any remaining tokens in your stack and in the token array before returning. This should only be necessary in the case of an error.

### 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 files 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 and one other partner. Cheating will be reported to SCCS. Both the copier and copiee will be held responsible.**