CS 32

Machine Problem 1

Deadline: 27 February 2017, 11:59 PM

General Instructions

- For the machine problem, you are to write your implementation using the **C programming language**. You must use **gcc (version 4 or later)** as a reference compiler.
- The name assigned to the source code file that will contain the main method (assuming multiple source code files will be implemented) is *polish.c*.
- Place a comment at the very beginning of your source code stating your student number and your full name.
- In the event that multiple source code files were created, place them all in a compressed folder named *polish.zip*.
- If you have consulted references (books, journal articles, online materials, other people) *apart from the one used as reference for the specification*, cite them as references by adding a comment detailing the references after your name. Use the MLA format for listing references: http://www.sciencebuddies.org/science-fair-projects/project_mla_format_examples.shtml. (For lines requiring italics, just place them inside quotation marks.)
- Add a comment before each function/method in your code, save for the main method, detailing the following: 1) input (if any) 2) output (if any) and 3) what the method does.
- You are only to submit the source code of your implementation. Submission of the machine problem should be done via e-mail. Attach the source code file/s, and write as the subject header of the e-mail: [CS 32] < *Student Number* > − < *Last Name, First Name* > − Machine Problem 1. For example, [CS 32] 202099969 - Kreuβ, Johann - Machine Problem 1. Send your answers to janmichaelyap@gmail.com.
- **You should receive a confirmation e-mail from me stating receipt of your deliverable within 24 hours upon your submission of the MP**. If you have not received any, forward your previous submission using the same subject header once more.
- If your *program does not compile and/or does not run properly for at least one of the test inputs even after minor edits to the source code*, then your **submission is deemed invalid and will not be accepted**. You will be asked to submit a working deliverable. **This provision will also hold for subsequent (repeat) submissions**. **Date of submission is considered to be the date when an acceptable deliverable was submitted.**
- If you have any questions regarding an item (EXCEPT the answer and solution) in the machine problem, do not hesitate to e-mail me to ask them. **However, I will no longer entertain questions that were sent or asked after 12:01am of 24 February 2017**.
- Follow the instructions as mentioned above, especially those pertaining to actual submission of the deliverable. **Lack of or erroneous implementation of any part of the instructions may be considered a deficient submission, and thus may not be accepted**.

Conversion of a simple program expression from infix to postfix form

You are to implement an algorithm that converts a program expression from infix to postfix form, as mentioned in Chapter 4.3.2 of E.P. Quiwa's book. The additional conditions for this machine problem are the following:

- The operands include only small letters of the English alphabet.
- The following arithmetic, relational and logical operators (ordered according to decreasing precedence) instead are allowed in the expression to be converted:

Operator/s	*Associativity*
!	*Right*
ˆ	*Right*
* / %	*Left*
+ -	*Left*
< <= > >=	*Left*
== !=	*Left*
& |	*Left*

Parentheses are to be expected in some of the expressions.

Implementation Specfications

The input should be a text file, with the first row specifying the number of expressions (and succeeding rows) contained in the input file, followed by rows, each of which representing a single program expression. Each expression will contain any of the listed operations, any valid operand, and possibly parentheses, written in a manner such that operands and operators are separated by a white space. There are no white spaces separating parentheses (open and closed) and the operands. Here is an example of the content of an input file:

```
3
a + b - c / d + e ^ f ^ g
(a - (b ^ (c & d) | e))
(a >= b) < c != ! b
```

The same format for the input is expected in the output of the program (i.e., spacing between operands and operators).

Required features for the machine problem are as follows:

- Use the C code in Chapter 4.2.5 of the E.P. Quiwa text for the implementation of a linked stack.
- Follow exactly the EASY code of the Polish procedure as detailed in Chapter 4.3.2 when implementing the algorithm.

Sample Run

Assume that the sample file contents in the previous section were containied in a file named *input1.txt*.

```
Program started.

Enter file name> input1.txt

Expression 1:
Infix: a + b - c / d + e ^ f ^ g
Postfix: a b + c d / - e f g ^ ^ +

Expression 2:
Infix: (a - (b ^ (c & d) || e))
Postfix: a b c d & ^ e | -
```

```
Expression 3:
Infix: (a >= b) < c != ! b
Postfix:  a b >= c < b ! !=

Program terminated.
```

Criteria for Grading

The machine problem submission will be graded via the following criteria:

- Program effectivity (50%)
  The program should be able to perform the algorithm according to specification on a number of test cases.

- Adherence to specification (30%)
  The implementation should, to the fullest extent possible, adhere to the specifications set for the machine problem. Much focus is given to the required feature/s as mentioned.

- Program documentation and readability (20%)
  The program should have proper internal documentation (i.e. comments detailing functionalities), and that the documentation details the mechanisms of each function and some code blocks clearly. The code should also be readable in the sense that function and variable names are as intuitive as possible, and that proper indentations/blocking of statements is observed (e.g. statements on the same level/block should be properly aligned, statements inside a *for* loop should be indented, etc.).