Evaluative Lab (4 Marks)

Birla Institute of Technology & Science, Pilani
Data Structures & Algorithms (CS F211)
Lab Assignment – 2 (Designing a compiler)

Problem

Designing a very simplified compiler.

Description

A compiler has two parts:
1. Front End &
2. Back End.

Front End of the compiler takes as input the source code (for example the C program) and converts it into an intermediate code. Back End of the compiler takes as input the intermediate code and converts it into machine code. Front End of the compiler also does the error checking of the source code and it reports the errors if it is able to find them.

Your "Source Code" will be a valid infix expression (there is no need for error checking) which will be read from stdin as a string. The variables will be alphabets (small as well as capital). The operators will be +, -, * and /. The expression can have the parenthesis '(' and ')'. Your "intermediate Code" will be the postfix expression equivalent to the infix expression. You will have to use the infix to postfix conversion algorithm and write the result on stdout. You have to use algorithm based on Stack data structure discussed in lecture. You may implement Stack either using linked list or using array.

For "Machine Code" we will consider a very simplified machine having only a single register and six instructions. The six instruction are as follows:

1. **LD A :** Places the operand A into the register.
2. **ST A :** Places the contents of the register into the variable A.
3. **AD A :** Adds the contents of the variable A to the register.
4. **SB A :** Subtracts the contents of the variable A form the register.
5. **ML A :** Multiplies the contents of the register by the variable A.
6. **DV A :** Divides the contents of the register by the variable A.

For converting the postfix expression into machine code you can use the above six instructions by replacing with suitable variables. You can also use temporary variables of the form TEMPn (TEMP1, TEMP2, …). For generating the machine code you will have to use the postfix evaluation algorithm with some modifications: in place of evaluating a subexpression (by popping two top most elements) you will have to write the equivalent machine code on the stdout (Starting from the second line you will have to write the machine code, one instruction per line), possibly using some temporary variable of the form TEMPn. In place of pushing the value of the subexpression, you will have to push the temporary variable TEMPn which stores the value of the subexpression.

Example:

Input: (A+B*C) / (D-E)

Output: Postfix Expression = ABC * + DE - / … line 1

Machine code will be generated as follows:

<u>* + D E - /</u> Stack 


```
LD    B          … line 2
ML    C          … line 3
ST    TEMP1      … line 4
```

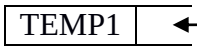
TEMP1 = B*C is pushed on the stack (after popping B and C)

<u>+ D E - /</u>

```
LD    A          …line 5
AD    TEMP1      … line 6
ST    TEMP2      … line 7
```

Stack 

TEMP2 = A+TEMP1 is pushed on the stack (after popping B and C)

<u>DE-/</u> Stack 

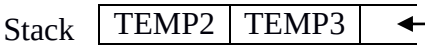
<u>-/</u> Stack 


```
LD    D          … line 8
SB    E          … line 9
ST    TEMP3      … line 10
```

TEMP3 = D-E is pushed on the stack (after popping D and E)

Evaluative Lab (4 Marks)

/
LD TEMP2 … line 11
DV TEMP3 … line 12
ST TEMP4 … line 13



TEMP4 = TEMP2/TEMP3 is pushed on the stack
(after popping TEMP2 and TEMP3)

Sample test cases:

Input:
(A+B*C) / (D-E)

Output:
ABC * + DE - /
LD B
ML C
ST TEMP1
LD A
AD TEMP1
ST TEMP2
LD D
SB E
ST TEMP3
LD TEMP2
DV TEMP3
ST TEMP4

Instructions:

- Whole input expression should be read as a string from stdin.
- Resultant postfix expression should be printed on stdout without any extra spaces.
- Machine code should be printed from second line of output and each instruction should be on a new line. No extra/empty new lines should be printed.
- For 1 hour 45 min, only a subset of test cases will be visible to students after submitting the code on the portal. After 1hour 45 min, all test cases will be made visible and they will have last 15 min to correct their code and resubmit.
- At the end of 2 hour period, the online system will stop accepting the submissions.

Evaluative Lab (4 Marks)

- Only the last submission by the student before end of lab will be considered for evaluation.
- Following messages by online portal will **tentatively** fetch these marks:
  - Correct ➔ 4 marks
  - Wrong-answer (correct only for some test cases) ➔ 3 marks
  - Run-error/Compiler-error/Timelimit-error ➔ 2 marks
- All submitted source code will be later checked manually by the instructor and final marks will be awarded, which will be posted on Nalanda after the lab assignment has been done by all lab sections.