ITU Computer Engineering Department
BLG 223E Data Structures, Fall 2021
Recitation #3

Problem Definition

You are asked to accept an infix string, representing a simple arithmetic expression, from the user, transform the string to postfix form and evaluate postfix expression.

Input: The input is a string containing only **single digit numbers**, arithmetic operators represented by + - * / by their conventional meanings. Parentheses **will not be used** in the input. You can assume that the input contains no semantical or syntactical errors.

Given: main() function with commented lines that may be used to test after the necessary functions are implemented. Also three helper functions, are provided: higher_priority() function calculates and returns the operator with higher priority or not. evaluate() function calculates the result of applying its third parameter as an arithmetic operation over its first and second parameters.

Wanted: You will complete two functions in the code as outlined below. Infix expression is taken from user in main() function. After that, 'infix_to_postfix()' function is called. You should do following transformation in these functions.

- void infix_to_postfix(string infix,string& postfix):
  - Scan the infix expression from left to right
  - If the scanned character is the number then this number is concatanated to postfix string
  - If the scanned character is an operator and the stack is not empty:
    - If the priority of the scanned operator is greater than the top most operator of the stack, push this operator into the stack
    - If the priority of the scanned operator is less than or equal to the top most operator of the stack, pop the operators from the stack and concatenate this operand to postfix string until find a low priority operator
  - If the scanned character is an operator and the stack is empty then push this operator into the stack
  - When the infix string ends pop all the operators remaining in the stack and concatenate to postfix string
- void eval_postfix(string expr):
  - Scan the prefix expression from left to right
  - Push number into stack
  - For an operator pop two numbers from stack, apply the operator and push the number back to stack
  - When the postfix string ends, pop the single number in the stack as the result

1 Example

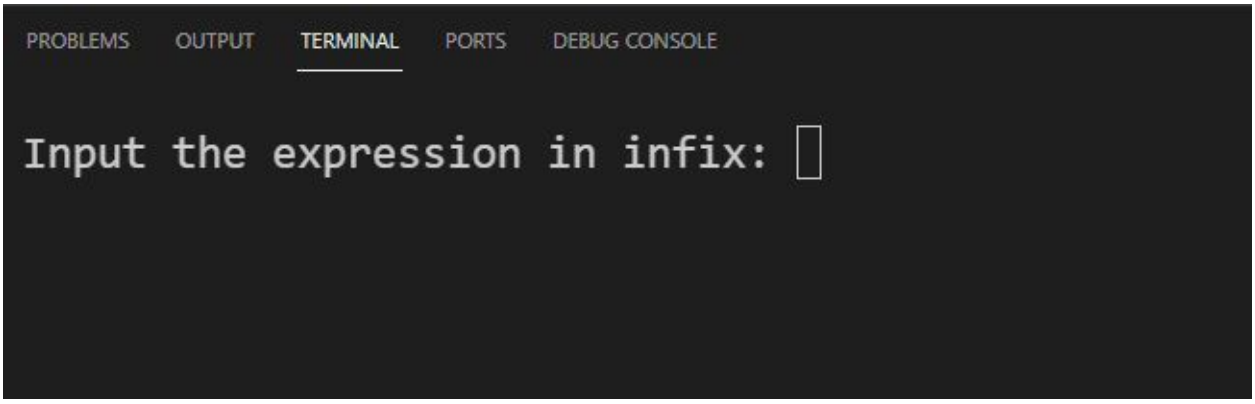

Figure 1: Terminal screen when the program is run.

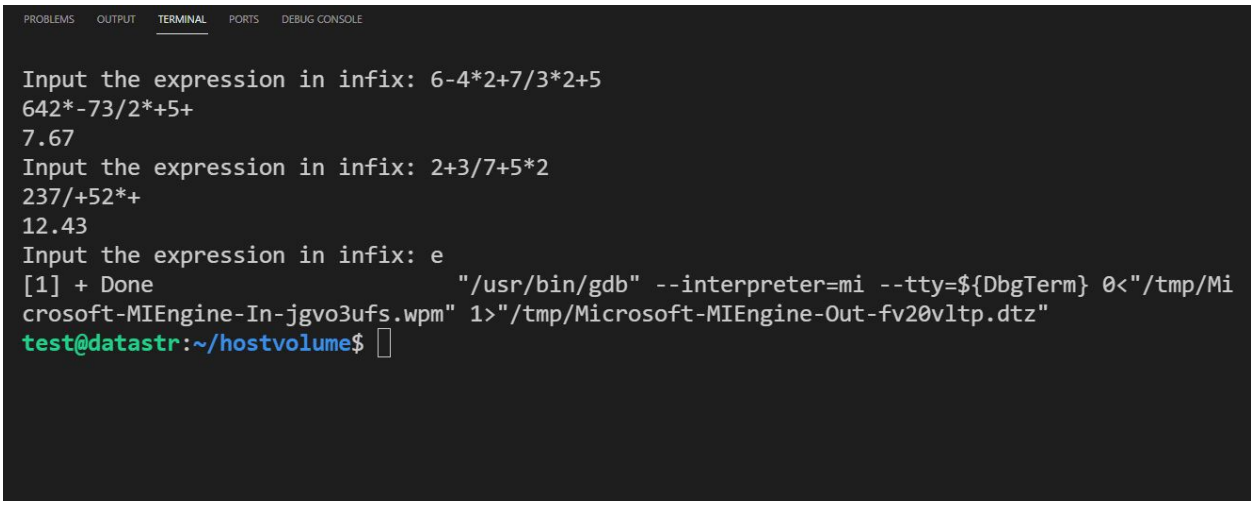

Figure 2: Terminal screen when the input is entered.

Submission Rules

- Make sure you write your name and number in all of the files of your project, in the following format:
  /* @Author
  Student Name: <student_name>
  Student ID : <student_id>
  Date: <date> */
- Use comments wherever necessary in your code to explain what you did.
- Your program will be checked by using **Calico(https://bitbucket.org/uyar/calico)** automatic checker.
- Do not share any code or text that can be submitted as a part of an assignment (discussing ideas is okay).
- Only electronic submissions through Ninova will be accepted no later than deadline.
- You may discuss the problems at an abstract level with your classmates, but you should not **share or copy code** from your classmates or from the Internet. You should submit your **own, individual** homework.

- Academic dishonesty, including cheating, plagiarism, and direct copying, is unacceptable.
- If you have any question about the recitation, you cand send e-mail to Yunus Emre Cebeci(cebeci16@itu.edu.tr).
- Note that **YOUR CODES WILL BE CHECKED WITH THE PLAGIARISM TOOLS!**