ALERT!

1. The objective of this lab is understanding and finding LIFO behavior in different problems.
2. This is an individual lab, you are strictly **NOT** allowed to discuss your solution with fellow colleagues, even not allowed asking how is he/she is doing, it may result in negative marking. You can **ONLY** discuss with your TAs or with me.
3. Beware of memory leaks and dangling pointers.
4. Pay attention to **GOOD coding conventions** e.g.
   - Proper indentation.
   - Meaning variable and function names.
   - Use camelCase naming convention
   - Use meaningful prompt lines/labels for all input/output
5. **Anyone caught in act of plagiarism would be awarded an "F" grade in this Lab.**

Task 01: [5+5+5 Marks]

Write a program that reads from the user a mathematical expression where each operand is space separated.
E.g. 13 + (24 – 19) × (212 / 2)

We have already discussed Infix to postfix conversion in the class. Now, you are required to implement that logic. After converting the expressions also evaluate the expression and return result.
Implement following four functions:

1. **validate (expression):** This function should check whether the given infix expression is correct and balanced or not. We will consider an expression correct if the parenthesis are correctly added, also the operands and operators are balanced. For example: $(a \times \{b + c\} - 4/x\ + [e - 5])$ is a correct expressions. While, $(5 + \{6 \times) - 2\}$ and 3 + – 2 are examples of incorrect expressions.
2. **infixToPostfix (expression):** This function should take an infix expression as input and returns equivalent postfix expression.
3. **evaluate (expression):** This function should take postfix expression as argument and returns the value of the expression after evaluation.

Also write a proper main program providing menu to make it easy to test your functions. No marks shall be given without this driver program. Remember! conversion, validation and evaluation all three tasks should be performed using stack. You may use string tokenizer to tokenize the string and use *atoi* function for string to integer casting.

Task 02: [10 Marks]

A palindrome is a word, phrase, number or other sequence of units that has the property of reading the same in either direction. Write a program that could determine whether the given string is a palindrome (using stack).
Examples:
Palindrome Words:

- Dad
- 1221
- Racecar
- Rotator
- Level
- Civic

Palindrome phrases:

- Too bad--I hid a boot.
- Do geese see God?
- "Go Hang a Salami! I'm a Lasagna Hog!"
  (title of a book on palindromes by Jon Agee, 1991)
  Note: Handle spaces and punctuation marks carefully.

Task 03: [10 Marks]

A common problem in text processing is to find the frequency of a particular word in any given text file. Your task is to write a program that takes a filename and a word from the user as strings and displays the number of occurrences of that word in that particular file. For example, the word "the" occurs 7 times in the following text file. Assume the search is *case sensitive*.

sample.txt

```
The Wheel of Time turns, and Ages come and pass, leaving memories that
become legend. Legend fades to myth, and even myth is long forgotten
when the Age that gave it birth comes again. In one Age, called the
Third Age by some, an Age yet to come, an Age long past, a wind rose on
the great plain called the Caralain Grass. The wind was not the
beginning. There are neither beginnings nor endings to the turning of
the Wheel of Time. But it was a beginning.
```