STACKS AND QUEUES 1

Problem 1. Write a program to check for balancing brackets

A bracket is considered to be any one of the following characters: (,), {, }, [, or].

Two brackets are considered to be a matched pair if the an opening bracket (i.e., (, [, or {) occurs to the left of a closing bracket (i.e.,),], or }) of the exact same type. There are three types of matched pairs of brackets: [ ], { }, and ( ).

A matching pair of brackets is not balanced if the set of brackets it encloses are not matched. For example, {[(])} is not balanced because the contents in between { and } are not balanced. The pair of square brankets encloses a single, unbalanced opening bracket, (, and the pair of parentheses encloses a single, unbalanced closing square branket,].

By this logic, we say a sequence of brackets is balanced if the following conditions are met:

- It contains no unmatched branckets.
- The subset of brackets enclosed within the confines of a matched pair of brackets is also a matched pair of brackets.

Give a string of brackets, determine whether each sequence of brackets is balanced.

Example:

- The string {[()]} meets both criteria for being a balanced string
- The string {[(])} is not balanced because the brackets enclosed by the matched pair { and } are not balanced: [(])
- The string {{[[(())]]}} meets both criteria for being a balanced string

Problem 2. Write a program to convert an infix expression that includes ^, (,), +, -, *, and / to postfix.

Given an infix expression in the form of a string **str**. Convert this infix expression to postfix expression.

- **Infix expression:** The expression of the form A **op** B. When an operator is in-between every pair of operands.
- **Postfix expression**: The expression of the form A B **op**. When an operator is followed for every pair of operands.

Example:

- Input - Infix expression: `A+B*(C^D-E)^(F+G*H)-I`
  Output - Postfix expression: `ABCD^E-FGH*+^*+I-`
- Input - Infix expression: `A*(B+C)/D`
  Output - Postfix expression: `ABC+*D/`

Problem 3. Write a program to convert a postfix expression to infix.

Example:

- Input – Postfix expression: ABC++
- Output – Infix expression: `(A+(B+C))`
- Input – Postfix expression: AB*C+
- Output – Infix expression: `((A*B)+C)`

Problem 4. Write a program to evaluate a prefix expression.

- **Prefix expression**: The expression of the form **op** A B. When an operator is before for every pair of operands.

Give a prefix expression, the task is to evaluate the expression and print the final value. Operators will only include the basic arithmetic operators like `*`, `/`, `+` and `-`.

Example:

- `Input: - + 7 * 4 5 + 2 0`
- `Output: 25`