ACADEMY OF TECHNOLOGY

Lab Assignment (Assignment 10)

Paper name: Data Structure and Algorithm

Code: PCC-CS391 **Semester: 3rd**

Discipline: CSE **Time: 2 Hours**

Date: August 10, 2022

1. Assume that there are only five operators ($*$, $/$, $+$, $-$) in an expression and operand is single digit only. Write a C/C++ Program
   (a) To Convert Infix to Postfix Expression using Stack.
   (b) To evaluate a given postfix expression.

Algorithm:

Algorithm 1: Infix-To-Postfix

```
   Input: Infix Expression
   Output: Postfix Expression
1  Scan character at a time from left to right;
2  while there is symbol do
3      if symbol is '(' then Push into the operator stack;
4      if symbol is an operand then Put it into output array;
5      if symbol is an operator then
6          if operator stack is empty then
7              Push into the stack;
8          end
9          if stack top is '(' then
10             Push into the stack;
11         end
12         if precedence(symbol) > precedence(stack top) then
13             Push the symbol into the operator stack;
14         end
15         else
16             while precedence(symbol) ≤ precedence(stack top) do
17                 pop element from operator stack;
18                 Put popped element into output array;
19             end
20             push the symbol into the operator stack;
21         end
22     end
23     if symbol is ')' then pop operator stack and put to into output array
        until the stack top is '(';
24     pop and ignore '(';
25 end
26 Now pop out all the remaining operators from the operator's stack and push
    into output array;
27 Display output array;
```

Evaluation of Postfix Expression:

Algorithm:

***Algorithm 2:* Evaluation of Postfix Expression**

```
    Input: Postfix Expression
    Output: Evaluated value of Postfix Expression
 1  Scan one character at a time from left to right;
 2  while there is symbol do
 3      if symbol is an operand then Push into the stack;
 4      if symbol is an operator X then
 5          $operand_2 := pop()$;
 6          $operand_1 := pop()$;
 7          $result := operand_1\ X\ operand_2$;
 8          Push result into the stack;
 9      end
10  end
11  pop stack to get the required value;
```

How to check if the given symbol is an operand

```c
int isOperand(char ch){
    return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z')
       || (ch >= '0' && ch <= '9');
}
```

How to find precedence of a given operator, higher value means higher precedence

```c
int precedence(char x){
  if(x == '^')
        return 3;
    if(x == '*' || x == '/')
        return 2;
    if(x == '+' || x == '-')
        return 1;
    return -1;
}
```

When an operator is encountered

```
    .................\;
    .................\;
    while (!s.isEmpty()
                  && precedence(symbol) <= precedence(s.peek()))
                      {
                if (symbol == '^' && s.peek() != '^')
                    break;
                else {
                    expression += s.peek();
                    s.pop();
                }
            }
            s.push(symbol);
        }
        .................\;
        .................\;
```