**Project Documentation**

**Name:** Attiqa Sheikh

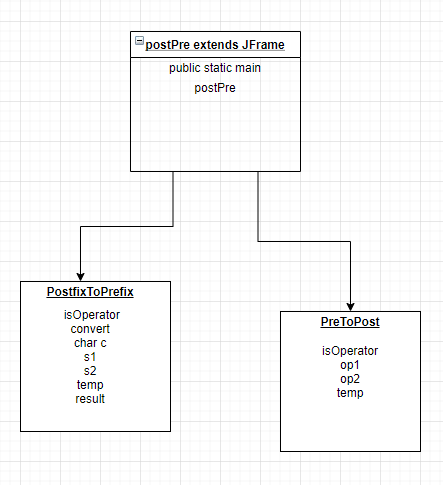
**Assignment:** Project 1

**Date:** May 27, 2020

**Problem Statement**: Write a program that converts prefix expressions to postfix and postfix expressions to prefix.

**Design:**

**UML Diagram:**



|  |
| --- |
| **Code:**  **1 import java.awt.event.ActionEvent;  2 import java.awt.event.ActionListener;  3 import java.util.\*;  4 import javax.swing.JButton;  5 import javax.swing.JFrame;  6 import javax.swing.JLabel;  7 import javax.swing.JTextField;  8   9 //class to convert postfix to prefix  10 class PostFixToPreFix {  11 //we store operatores in cases to check and access...  12 boolean isOperator(char x){  13 switch (x){  14 case '-':  15 case '+':  16 case '/':  17 case '\*':  18 case '^':  19 return true;  20 }  21 return false;  22 }  23 //method to convert user expression to prefix...  24 public String convert(String expression){  25 Stack<String> stack = new Stack<>();  26 for (int i = 0; i <expression.length() ; i++) {  27 char c = expression.charAt(i);  28 if(isOperator(c)){  29 String s1 = stack.pop();  30 String s2 = stack.pop();  31 String temp = c + s2 + s1;  32 stack.push(temp);  33 }else{  34 stack.push(c+"");  35 }  36 }  37 String result = stack.pop();  38 return result;  39 }  40 }  41   42 //class to convert prefix to postfix  43 class PretoPost{  44 // funtion to check if character is an operand  45 boolean isOperator(char x)  46 {  47 switch (x)  48 {  49 case '+':  50 case '-':  51 case '/':  52 case '\*':  53 return true;  54 }  55 return false;  56 }  57 public  58 // Convert prefix to Postfix expression  59 String preToPost(String pre\_exp)  60 {  61 Stack<String> s= new Stack<String>();   62 // length of expression  63 int length = pre\_exp.length();  64 // reading from right to left  65 for (int i = length - 1; i >= 0; i--)  66 {  67 // check if symbol is operator  68 if (isOperator(pre\_exp.charAt(i)))  69 {  70 // pop two operands from stack  71 String op1 = s.peek(); s.pop();  72 String op2 = s.peek(); s.pop();  73 // concat the operands and operator  74 String temp = op1 + op2 + pre\_exp.charAt(i);  75 // Push String temp back to stack  76 s.push(temp);  77 }  78 // if symbol is an operand  79 else  80 {  81 // push the operand to the stack  82 s.push( pre\_exp.charAt(i)+"");  83 }}  84 // stack contains only the Postfix expression  85 return s.peek();  86 }};  87 //Main class which extendsd GUI form to JFrame  88 public class postPre extends JFrame  89 {  90 JTextField txt1;  91 JTextField txt2;  92   93 JButton button1;  94 JButton button2;  95   96 JLabel jl1;  97 JLabel jl2;  98   99 postPre() 100 { 101 this.setTitle("Expression Converter"); 102 this.setLayout(null); 103  104 jl1 = new JLabel("Enter Expression"); 105 jl1.setBounds(100, 50, 100, 50); 106 add(jl1); 107  108 jl2 = new JLabel("Result"); 109 jl2.setBounds(150, 200, 80, 50); 110 add(jl2); 111  112 txt1 = new JTextField(""); 113 txt1.setBounds(210, 60, 210, 30); 114 add(txt1); 115  116 txt2 = new JTextField(""); 117 txt2.setEditable(false); 118 txt2.setBounds(200, 210, 210, 30); 119 add(txt2); 120  121 button1 = new JButton("Prefix to Postfix"); 122 button1.setBounds(100, 130, 150, 30); 123  124 button1.addActionListener( 125  126 new ActionListener(){ 127 @Override 128 public void actionPerformed(ActionEvent ae) { 129 String pre\_exp = txt1.getText(); 130 PretoPost ptp = new PretoPost(); 131 txt2.setText(ptp.preToPost(pre\_exp)); 132 }}); 133  134 add(button1); 135 button2 = new JButton("Postfix to Prefix"); 136 button2.setBounds(270, 130, 150, 30); 137 button2.addActionListener( 138 new ActionListener(){ 139  140 @Override 141 public void actionPerformed(ActionEvent ae) { 142 String exp = txt1.getText(); 143 PostFixToPreFix ptp = new PostFixToPreFix(); 144 txt2.setText(ptp.convert(exp)); 145 }}); 146  147 add(button2); 148 this.setVisible(true); 149 this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); 150 this.setBounds(300, 200, 500, 300); 151 }  152 public static void main(String[] args){ 153 postPre convertor = new postPre(); 154 }}** |

**Testing: For each testing scenario you choose to test this program describe the following:**

**Test #1**

**Data**: \* 2 + 2 - + 12 9 2

**Test Cases**: Checking to compare the output of example to my own output.

**Expected Results**: 2\*

**Test #2**

**Data**: \* 2 + 2 - + 12 9 2

**Test Cases**: Checking to see if exceptions are thrown when clicking Postfix to Prefix rather than Prefix to Postfix

**Expected Results**: Exceptions thrown, blank result field

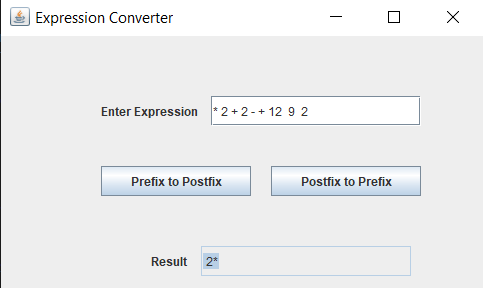
**Test #3**

**Data**: 4 5 7 2 + - \*

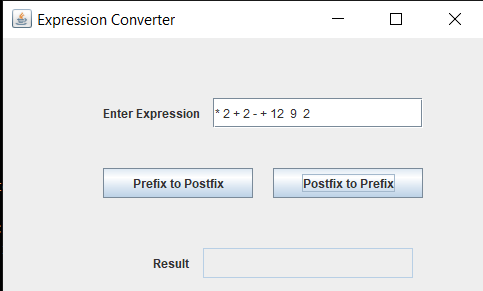
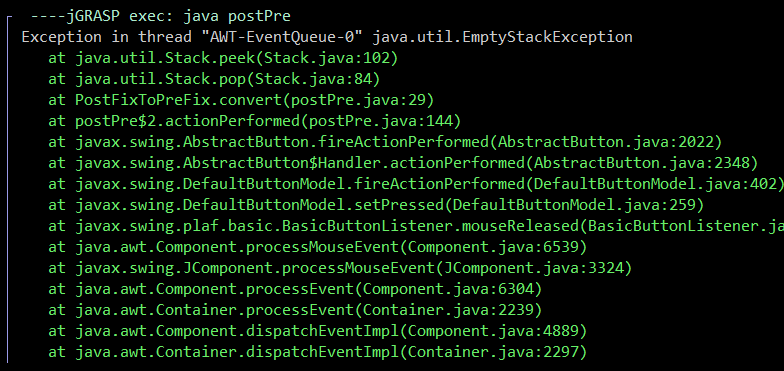
**Test Cases**: Checking to see functionality of Postfix to Prefix for expression.

**Expected Results**: \* - + 2

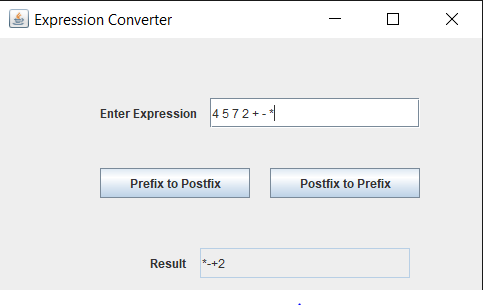
**Output:**

**Test #1**

**Test #2**



**Test #3**



**Reflection:** Some important concepts in this project that I learned was creating and managing a stack. At first it was difficult for me to remember how to use ActionListener and implement functions into my buttons but after a little review on the Java website I was able to figure it out. Initially I had begun designing my GUI using JPanel, but I had forgotten that it was meant to be on JFrame, which I quickly changed after an hour of working on it and not understanding what I was doing wrong. I will use this assignment as a tool for any future errors I might make, I definitely realize that there are a few errors in my project, but I have given my all to this project this entire week. I may not be an exceptional programmer but I am definitely improving as I have noticed this week through my effort and new knowledge.