|  |
| --- |
| Importjava.util.Stack; |
| Import java.util.\*; |
|  | public class inftopre { |
|  | static int precedence(char c){ |
|  | switch (c){ |
|  | case '+': |
|  | case '-': |
|  | return 1; |
|  | case '\*': |
|  | case '/': |
|  | return 2; |
|  | case '^': |
|  | return 3; |
|  | } |
|  | return -1; |
|  | } |
|  |  |
|  | static StringBuilder inftopre(String expression){ |
|  |  |
|  | StringBuilder result = new StringBuilder(); |
|  | StringBuilder input = new StringBuilder(expression); |
|  | input.reverse(); |
|  | Stack<Character> stack = new Stack<Character>(); |
|  |  |
|  | char [] charsExp = new String(input).toCharArray(); |
|  | for (int i = 0; i < charsExp.length; i++) { |
|  |  |
|  | if (charsExp[i] == '(') { |
|  | charsExp[i] = ')'; |
|  | i++; |
|  | } |
|  | else if (charsExp[i] == ')') { |
|  | charsExp[i] = '('; |
|  | i++; |
|  | } |
|  | } |
|  | for (int i = 0; i <charsExp.length ; i++) { |
|  | char c = charsExp[i]; |
|  |  |
|  | //check if char is operator or operand |
|  | if(precedence(c)>0){ |
|  | while(stack.isEmpty()==false && precedence(stack.peek())>=precedence(c)){ |
|  | result.append(stack.pop()); |
|  | } |
|  | stack.push(c); |
|  | }else if(c==')'){ |
|  | char x = stack.pop(); |
|  | while(x!='('){ |
|  | result.append(x); |
|  | x = stack.pop(); |
|  | } |
|  | }else if(c=='('){ |
|  | stack.push(c); |
|  | }else{ |
|  | //character is neither operator nor "(" |
|  | result.append(c); |
|  | } |
|  | } |
|  |  |
|  | for (int i = 0; i <=stack.size() ; i++) { |
|  | result.append(stack.pop()); |
|  | } |
|  | return result.reverse(); |
|  | } |
|  |  |
|  | public static void main(String[] args) { |
|  | String exp = "A+B\*(C^D-E)"; |
|  | System.out.println("Infix Expression: " + exp); |
|  | System.out.println("Prefix Expression: " + inftopre(exp)); |
|  | } |
|  | } |