EXPERIMENT NO 10

|  |
| --- |
| OPERATORS = set(['+', '-', '\*', '/', '(', ')'])  PRI = {'+':1, '-':1, '\*':2, '/':2}  def infix\_to\_postfix(formula):  stack = [] output = '' for ch in formula: if ch not in OPERATORS:  output += ch elif ch == '(':  stack.append('(') elif ch == ')': while stack and stack[-1] != '(':  output += stack.pop() stack.pop() # pop '(' else: while stack and stack[-1] != '(' and PRI[ch] <= PRI[stack[-1]]:  output += stack.pop() stack.append(ch)  while stack:  output += stack.pop() print(f'POSTFIX: {output}') return output  def infix\_to\_prefix(formula):  op\_stack = [] exp\_stack = [] for ch in formula:  if not ch in OPERATORS: exp\_stack.append(ch) elif ch == '(':  op\_stack.append(ch) elif ch == ')': while op\_stack[-1] != '(': op = op\_stack.pop() a = exp\_stack.pop() b = exp\_stack.pop() exp\_stack.append( op+b+a ) op\_stack.pop() # pop '(' |

|  |
| --- |
| else:  while op\_stack and op\_stack[1] != '(' and PRI[ch] <= PRI[op\_stack[-1]]:  op = op\_stack.pop() a = exp\_stack.pop() b = exp\_stack.pop() exp\_stack.append( op+b+a ) op\_stack.append(ch)  while op\_stack:  op = op\_stack.pop() a = exp\_stack.pop() b = exp\_stack.pop() exp\_stack.append( op+b+a ) print(f'PREFIX: {exp\_stack[-1]}') return exp\_stack[-1]    def generate3AC(pos): exp\_stack = [] t = 1 for i in pos: if i not in OPERATORS: exp\_stack.append(i) else:  print(f't{t} := {exp\_stack[-2]} {i} {exp\_stack[-1]}') exp\_stack=exp\_stack[:-2] exp\_stack.append(f't{t}') t+=1 def generate3ACTable(pos):  exp\_stack = [] t = 1 for i in pos: if i not in OPERATORS: exp\_stack.append(i) else:  print(f' {i}\t|\t{exp\_stack[-2]}\t|\t{exp\_stack[-1]}\t|\tt{t} ') exp\_stack=exp\_stack[:-2] exp\_stack.append(f't{t}') t+=1  expres = input("INPUT THE EXPRESSION: ") pre = infix\_to\_prefix(expres) print("") pos = infix\_to\_postfix(expres) generate3AC(pos) |

print("\n-----------Quadruple Table----------------\n") print("op\t|\targ1\t|\targ2\t|\tResult\n") generate3ACTable(pos)

**OUTPUT:**



