Ishita Hardasmalani

C14-2103058

EXPERIMENT NO.5

Aim : Write a program to implement Three Address Code

Code:

```
def get_precedence(c):
   if c == '^':
      return 3
   elif c == '*' or c == '/':
      return 2
   return 1
def infix_to_postfix(ip):
   res = ""
   stack = []
   n = len(ip)
   for i in range(n):
      if ('A' <= ip[i] <= 'Z') or ('a' <= ip[i] <= 'z'):
         res += ip[i]
      elif ip[i] == '(':
         stack.append('(')
      elif ip[i] == ')':
         while stack[-1] != '(':
            res += stack.pop()
         stack.pop()
      else:
         while stack and get_precedence(ip[i]) <= get_precedence(stack[-1]):
            res += stack.pop()
         stack.append(ip[i])
```

```
  while stack:
      res += stack.pop()
return res
def is_operator(c):
    return c in ('+', '-', '*', '/', '^')

def tac(postfix, input_str):
    stack = []
    print("\nTAC statements :: \n")

    ct = 1
    for i in range(len(postfix)):
        if not is_operator(postfix[i]):
            stack.append(postfix[i])
        else:
            op2 = stack.pop()
            op1 = stack.pop()

            intr_rhs = op1 + postfix[i] + op2
            intr_lhs = 't' + str(ct)

            print(f"{ct}. {intr_lhs} = {intr_rhs}\n")
            ct += 1

            stack.append(intr_lhs)

    print(f"{ct}. {input_str[0]} = {stack.pop()}\n")
if __name__ == "__main__":
    ip = "a=b*c+b*c"
    infix = ip[2:]
    postfix = infix_to_postfix(infix)
    tac(postfix, ip)
```

<u>OUTPUT:</u>

```
TAC statements ::

1. t1 = b*c

2. t2 = b*c

3. t3 = t1+t2

4. a = t3
```