University of Asia Pacific
Department of Computer Science & Engineering

Course Title: Compiler Design Lab

Course Code: CSE 430

Lab Exercise-05 Solution

Submitted by:-

Abid Al Mohtasin

20101089

B1

Description of the exercise:

Write a program to generate three address code from a given expression. Three-address code (often abbreviated to TAC or 3AC) is an intermediate code used by optimizing compilers to aid in the implementation of code-improving transformations. Three address code is easy to generate and can be easily converted to machine code. It makes use of at most three addresses and one operator to represent an expression and the value computed at each instruction is stored in a temporary variable generated by the compiler. The compiler decides the order of operation given by three address code.

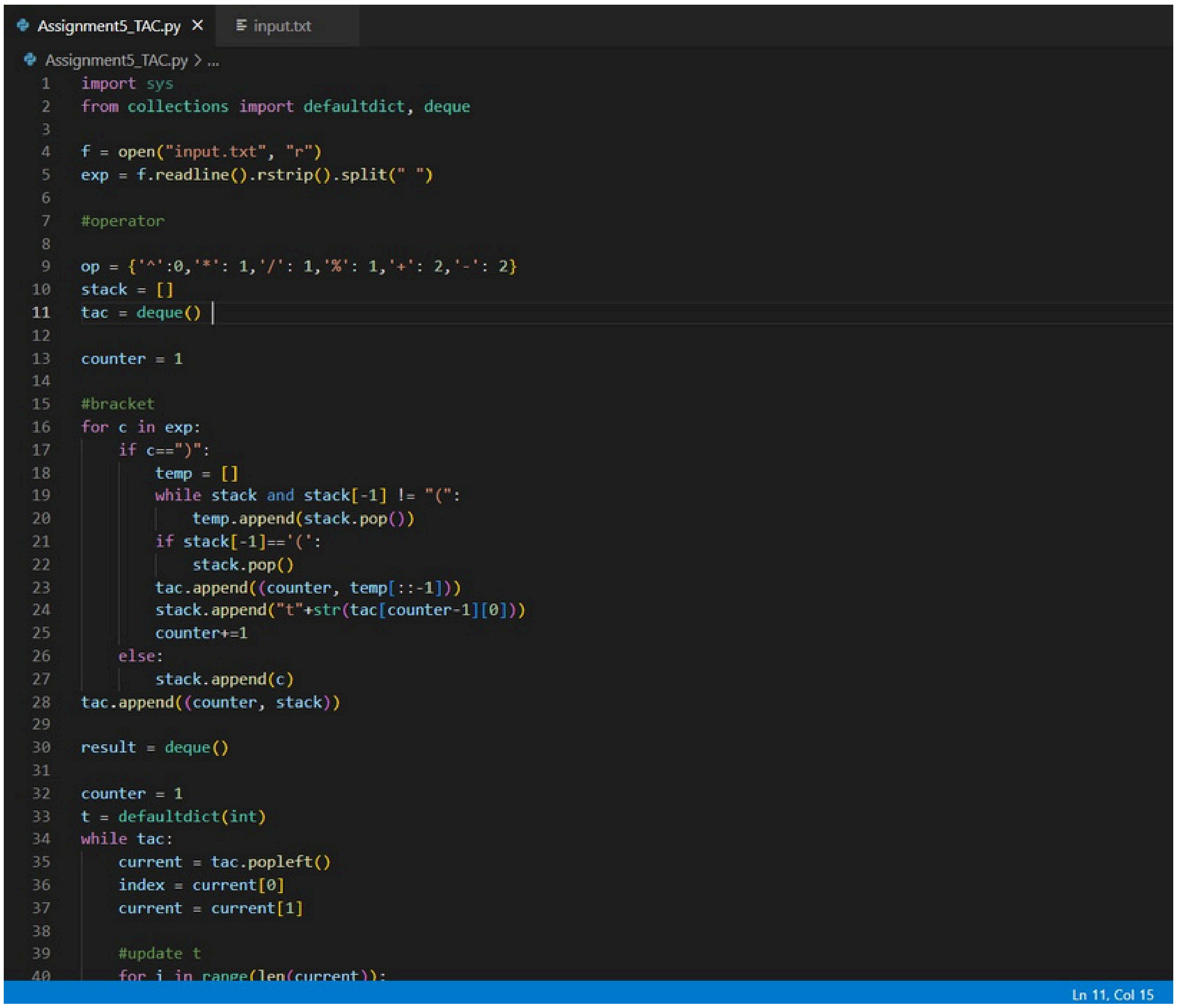
Sample Inputs:

```
- (a * b) + (c + d) – (a + b + c + d)
```

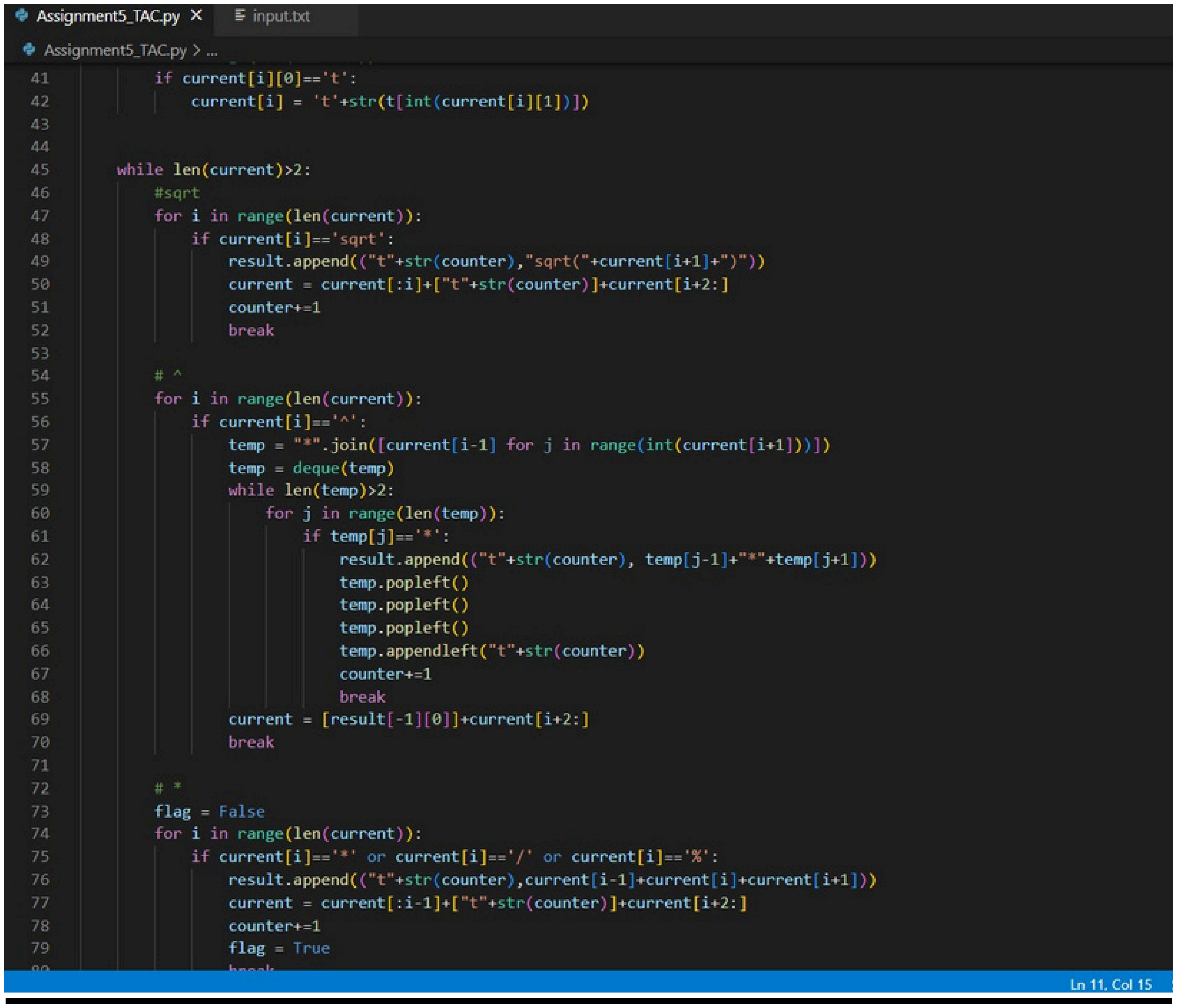
Sample Outputs:

```
T1 = a x b
T2 = uminus T1
T3 = c + d
T4 = T2 +T3
T5 = a + b
T6 = T3 + T5
T7 = T4
```

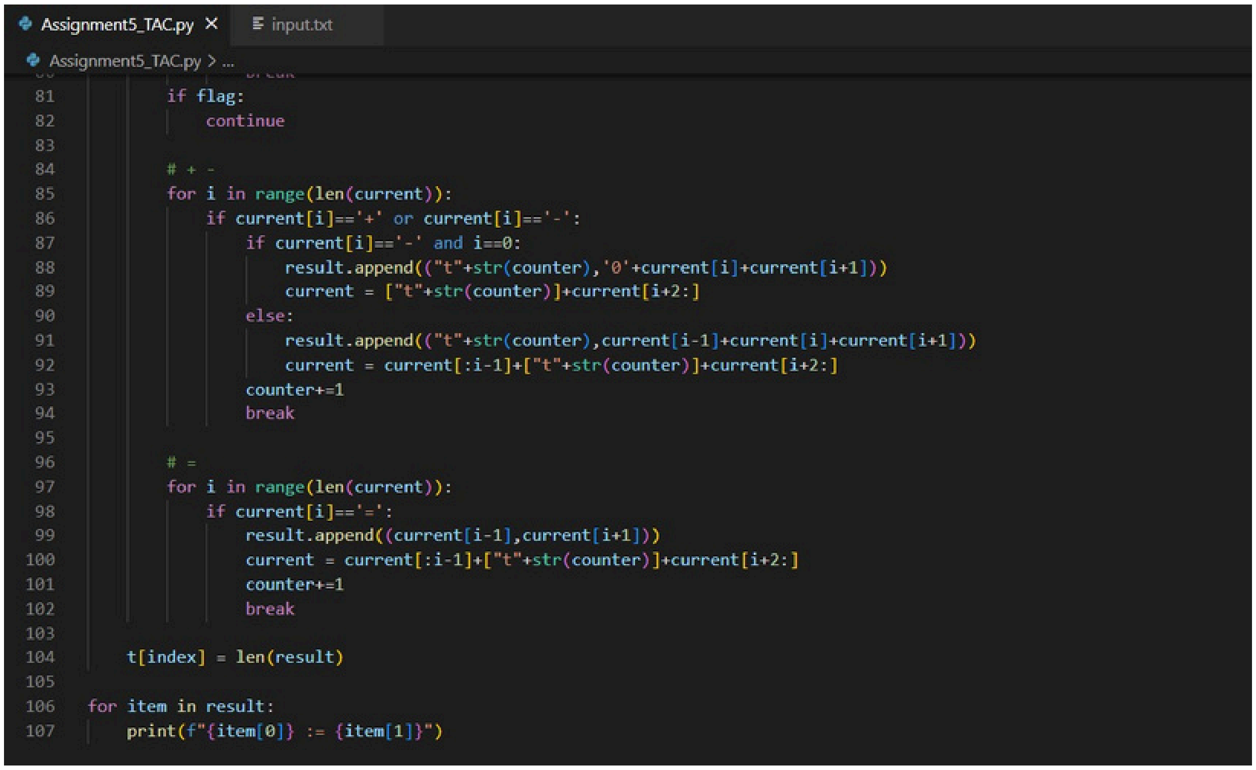
Code:



```python
import sys
from collections import defaultdict, deque

f = open("input.txt", "r")
exp = f.readline().rstrip().split(" ")

#operator

op = {'^':0,'*': 1,'/': 1,'%': 1,'+': 2,'-': 2}
stack = []
tac = deque()

counter = 1

#bracket
for c in exp:
    if c==")":
        temp = []
        while stack and stack[-1] != "(":
            temp.append(stack.pop())
        if stack[-1]=='(':
            stack.pop()
        tac.append((counter, temp[::-1]))
        stack.append("t"+str(tac[counter-1][0]))
        counter+=1
    else:
        stack.append(c)
tac.append((counter, stack))

result = deque()

counter = 1
t = defaultdict(int)
while tac:
    current = tac.popleft()
    index = current[0]
    current = current[1]

    #update t
    for i in range(len(current)):
```



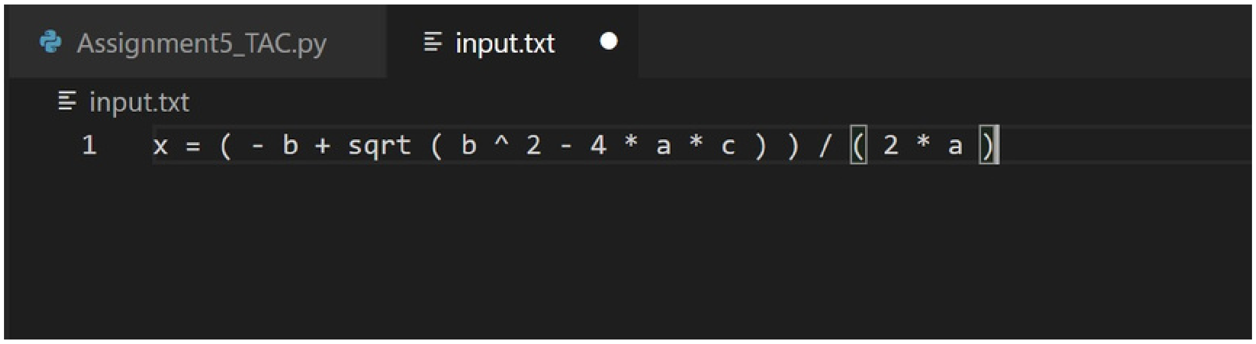
```python
        if current[i][0]=='t':
            current[i] = 't'+str(t[int(current[i][1])])


    while len(current)>2:
        #sqrt
        for i in range(len(current)):
            if current[i]=='sqrt':
                result.append(("t"+str(counter),"sqrt("+current[i+1]+")"))
                current = current[:i]+["t"+str(counter)]+current[i+2:]
                counter+=1
                break

        # ^
        for i in range(len(current)):
            if current[i]=='^':
                temp = "*".join([current[i-1] for j in range(int(current[i+1]))])
                temp = deque(temp)
                while len(temp)>2:
                    for j in range(len(temp)):
                        if temp[j]=='*':
                            result.append(("t"+str(counter), temp[j-1]+"*"+temp[j+1]))
                            temp.popleft()
                            temp.popleft()
                            temp.popleft()
                            temp.appendleft("t"+str(counter))
                            counter+=1
                            break
                current = [result[-1][0]]+current[i+2:]
                break

        # *
        flag = False
        for i in range(len(current)):
            if current[i]=='*' or current[i]=='/' or current[i]=='%':
                result.append(("t"+str(counter),current[i-1]+current[i]+current[i+1]))
                current = current[:i-1]+["t"+str(counter)]+current[i+2:]
                counter+=1
                flag = True
```


Assignment5_TAC.py × input.txt

Assignment5_TAC.py > ...

```python
        if flag:
            continue

        # + -
        for i in range(len(current)):
            if current[i]=='+' or current[i]=='-':
                if current[i]=='-' and i==0:
                    result.append(("t"+str(counter),'0'+current[i]+current[i+1]))
                    current = ["t"+str(counter)]+current[i+2:]
                else:
                    result.append(("t"+str(counter),current[i-1]+current[i]+current[i+1]))
                    current = current[:i-1]+["t"+str(counter)]+current[i+2:]
                counter+=1
                break

        # =
        for i in range(len(current)):
            if current[i]=='=':
                result.append((current[i-1],current[i+1]))
                current = current[:i-1]+["t"+str(counter)]+current[i+2:]
                counter+=1
                break

    t[index] = len(result)

for item in result:
    print(f"{item[0]} := {item[1]}")
```

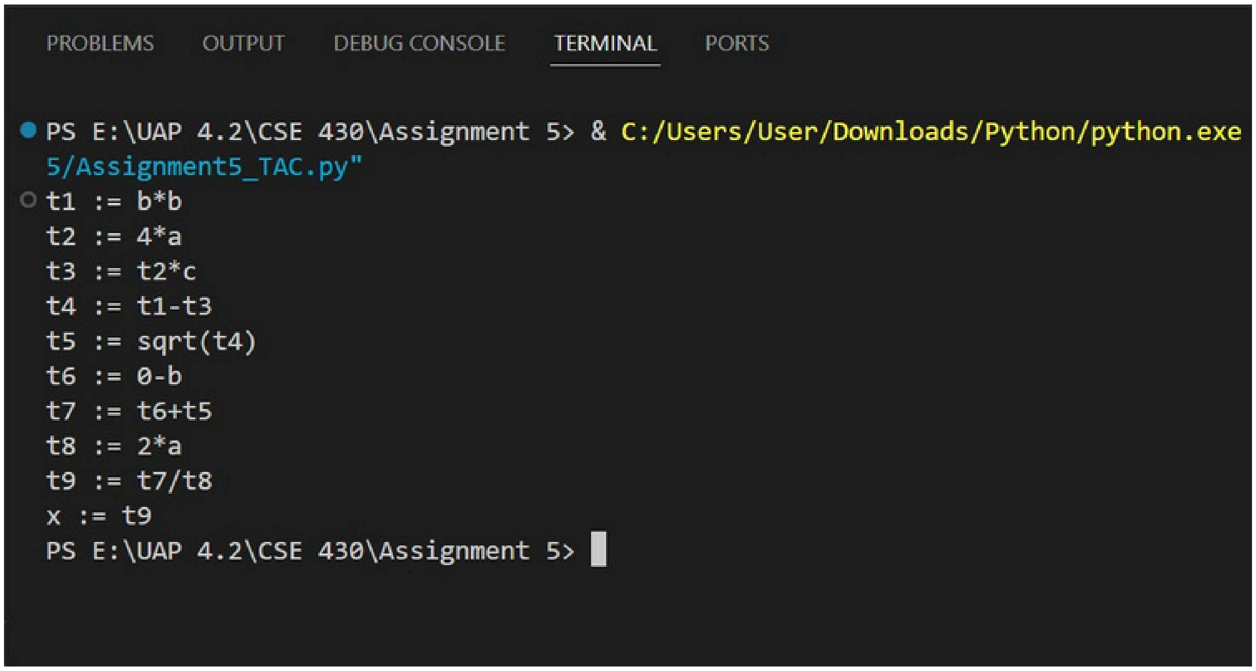
Sample Input:


Assignment5_TAC.py input.txt ●

input.txt

```
x = ( - b + sqrt ( b ^ 2 - 4 * a * c ) ) / ( 2 * a )
```

Observed Output:



```
PS E:\UAP 4.2\CSE 430\Assignment 5> & C:/Users/User/Downloads/Python/python.exe
5/Assignment5_TAC.py"
t1 := b*b
t2 := 4*a
t3 := t2*c
t4 := t1-t3
t5 := sqrt(t4)
t6 := 0-b
t7 := t6+t5
t8 := 2*a
t9 := t7/t8
x := t9
PS E:\UAP 4.2\CSE 430\Assignment 5>
```