University of Asia Pacific

CSE 430

Compiler Design Lab

Lab Exercise 6

Three Address Code Generation

Name: Md. Azim Islam

Registration: 19201026

Section: A2

Course Code: CSE 430

Semester: Spring 2023

Baivab Das

Lecturer, University of Asia Pacific, Dhaka

CODE

```
from collections import defaultdict
import re
from collections import deque

table = defaultdict(list)
operators = set(["/", "+", "-", "*", "^"])
def valid(string):
    if len(string) <= 3:
        return False
    for c in string:
        if c not in table and c not in operators:
            return True
    return False

def gen_var(string):
    return f"t{len(table)+1}"

def get_bracket(string):
    stack = []
    opening = 0
    closing = 0
    start = float('inf')
    end = -1
    for i, c in enumerate(string):
        if c == "(":
            opening += 1
            start = min(start, i)
        elif c == ")":
            closing += 1
            end = max(end, i)
```

```
        if opening and opening == closing:
            return (start, end+1)
    return None

def get_op(string, op):
    try:
        t = string.index(op)
        start = t-1
        end = t+2
        return (start, end)
    except ValueError:
        return False

def gen_tac(string):
    while valid(string): # While the string does not only consists of temporary variables and operators.
        # Checking for bracket "()"
        v = get_bracket(string)
        if v:
            start = v[0]
            end = v[1]
            # print(v[0], v[1], string[start:end])
            var = gen_var(string[start:end])
            table[var] = gen_tac(string[start+1:end-1])
            string[start:end] = string[start+1:end-1]
            # string[start:end] = table[var]
            continue
        # Checking for power "^"
        v = get_op(string, "^")
        if v:
            start = v[0]
```

```
    end = v[1]
    var = gen_var(string)
    table[var] = string[start:end]
    string[start:end]=[var]
    # print(string, var, table)
    gen_tac(string)
    continue
# Checking for division "/"
v = get_op(string, "/")
if v:
    start = v[0]
    end = v[1]
    var = gen_var(string)
    table[var] = string[start:end]
    string[start:end]=[var]
    # print(string, var, table)
    gen_tac(string)
    continue
# Checking for multiplication "*"
v = get_op(string, "*")
if v:
    start = v[0]
    end = v[1]
    var = gen_var(string)
    table[var] = string[start:end]
    string[start:end]=[var]
    # print(string, var, table)
    gen_tac(string)
    continue      # Checking for addition "+"
v = get_op(string, "+")
if v:
```

```
        start = v[0]
        end = v[1]
        var = gen_var(string)
        table[var] = string[start:end]
        string[start:end]=[var]
        # print(string, var, table)
        gen_tac(string)
        continue
    # Checking for substraction "-"
    v = get_op(string, "-")
    if v:
        start = v[0]
        end = v[1]
        var = gen_var(string)
        table[var] = string[start:end]
        string[start:end]=[var]
        # print(string, var, table)
        gen_tac(string)
        continue
    # Checking for assignment "="
    v = get_op(string, "=")
    if v:
        start = v[0]
        end = v[1]
        # var = gen_var(string)
        table[v] = string[end-1:]
        return string
        # string[start:end]=[var]
        # print(string, var, table)
        # gen_tac(string)
        continue
```

```
    return string


string = input()
out = gen_tac(string.split(" "))
print(f'{out[0]} := {" ".join(out[2:])}')
stack = deque()
visited = set()
for c in out:
    if c in table:
        stack.append(c)
        visited.add(c)

while stack:
    v = stack.popleft()
    print(f'{v} := {" ".join(table[v])}')
    for c in table[v]:
        if c in table and c not in visited:
            stack.append(c)
            visited.add(c)
```

INPUT

```
x = ( ( 0 - b ) + ( b ^ 2 - 4 * a * c ) ^ 0.5 ) / ( 2 * a )
```

OUTPUT

```
x := t24
t24 := t23 - t21
```

```
t23 := 0 - t22
t21 := t20 * a
t22 := b + t16
t20 := t19 * t18
t16 := b ^ 2
t19 := 4 * a
t18 := t17 / 2
t17 := c ^ 0.5
```

INPUT

```
x = ( 0 - ( a * b ) ) + ( c + d ) - ( a + b + c + d )
```

OUTPUT

```
x := t14
t14 := t13 - t12
t13 := 0 - t9
t12 := t11 + d
t9 := t8 + d
t11 := t10 + c
t8 := t7 + c
t10 := a + b
t7 := a * b
```