**Shri Ramdeobaba College of Engineering and Management, Nagpur**

**Department of Computer Science and Engineering**

**Session: 2024-2025**

**Compiler Design Lab**

**Name: Siddhi Kothekar**

**Roll no: 19**

**Batch: A2**

**PRACTICAL No. 5**

**Topic:** Three Address Code Generation

**Platform:** Windows

**Language to be used:** Python (based on the companies targeted for placement)

**CO Mapped:** CO4- Learn three address code generation and implement code optimization techniques for improving the performance of a program segment.

**Aim: Write a program to generate three address code for the given language construct using SDTS.**

1. For loop

**Code:**

code = "for ( int i = 0 ; i < 10 ; i++ ) { b = a + 3 ; d = c - 2 ; }"

def generate\_tac(code):

    # Split the code into non-empty tokens

    tokens = [token for token in code.split() if token]

    # Validate that the input follows the expected structure

    if (len(tokens) < 14 or tokens[0] != "for" or tokens[1] != "(" or

        tokens[2] != "int" or tokens[4] != "=" or tokens[6] != ";" or

        tokens[3] != tokens[7]):

        return "Invalid input format"

    # Extract loop variable and bounds

    loop\_var = tokens[3]

    start\_val = int(tokens[5])

    end\_val = int(tokens[9])

    tac\_lines = []

    current\_line = 1

    # Extract the loop body statements

    # (tokens from index 14 to the penultimate token, joined and then split on ';')

    body\_str = " ".join(tokens[14:-1])

    statements = [stmt.strip() for stmt in body\_str.split(';') if stmt.strip()]

    # Generate TAC for loop initialization

    tac\_lines.append(f"[{current\_line}] {loop\_var} = {start\_val}")

    current\_line += 1

    # Calculate the destination line number for loop exit.

    # This is equivalent to: len(statements)\*2 + current\_line + 3 (with tac\_loop\_lines = 3)

    jump\_dest = len(statements) \* 2 + current\_line + 3

    tac\_lines.append(f"[{current\_line}] if {loop\_var} > {end\_val} goto {jump\_dest}")

    current\_line += 1

    temp\_index = 1

    # Process each statement in the loop body

    for stmt in statements:

        if "=" in stmt:

            lhs, rhs = map(str.strip, stmt.split("=", 1))

            tac\_lines.append(f"[{current\_line}] T{temp\_index} = {rhs}")

            current\_line += 1

            tac\_lines.append(f"[{current\_line}] {lhs} = T{temp\_index}")

            current\_line += 1

            temp\_index += 1

        else:

            tac\_lines.append(f"[{current\_line}] {stmt}")

            current\_line += 1

    # Add the loop increment and jump back to the beginning of the loop

    tac\_lines.append(f"[{current\_line}] {loop\_var} = {loop\_var} + 1")

    current\_line += 1

    tac\_lines.append(f"[{current\_line}] goto 2")

    current\_line += 1

    tac\_lines.append(f"[{current\_line}] END")

    return tac\_lines

for line in generate\_tac(code):

    print(line)

**Output**:

