Shri Ramdeobaba College of Engineering and Management, Nagpur
Department of Computer Science and Engineering
Session: 2021-2022 [EVEN SEM]

Compiler Design Lab

Name : Kush Munot

Sec : A

Roll no. : 47

Batch : A3

Subject : Compiler Design

PRACTICAL No. 8

Aim: Write a program to generate the code using **simple code generation algorithm.**

Code :

```
def generate_TAC(assignment):
    var, expression = assignment.split(" = ")
    operand1, operation, operand2 = expression.split(" ")
    tac = f"{var} = {operand1} {operation} {operand2}"
    return tac


def generate_assembly(tac):
    asm = []


    var, expression = tac.split(" = ")
    operand1, operation, operand2 = expression.split(" ")


    asm.append(f"MOV {operand1}, R0")  # Load first operand into R0
    asm.append(f"MOV {operand2}, R1")  # Load second operand into R1


    if operation == "+":
        asm.append("ADD R0, R1")  # Add R0 and R1
    elif operation == "-":
```

```
        asm.append("SUB R0, R1")  # Subtract R1 from R0
    elif operation == "*":
        asm.append("MUL R0, R1")  # Multiply R0 and R1
    elif operation == "/":
        asm.append("DIV R0, R1")  # Divide R0 by R1

    asm.append(f"MOV R0, {var}")  # Move result from R0 to the destination
variable

    return asm

def process_assignment_statements(assignments):
    tac_statements = []
    assembly_statements = []

    for assignment in assignments:
        tac = generate_TAC(assignment)
        tac_statements.append(tac)

        assembly = generate_assembly(tac)
        assembly_statements.extend(assembly)
        assembly_statements.append("")

    return tac_statements, assembly_statements

if __name__ == "__main__":
    assignments = [
        "z = x + y",
        "a = b * c",
```

```
        "d = e - f",
        "g = h / i",
    ]

    tac_statements, assembly_statements =
process_assignment_statements(assignments)

    print("Three Address Code:")
    for tac in tac_statements:
        print(tac)
    print()

    print("Assembly Code:")
    for assembly in assembly_statements:
        print(assembly)
```

Input:

```
        "z = x + y",
        "a = b * c",
        "d = e - f",
        "g = h / i",
```

Output:

```
Three Address Code:
z = x + y
a = b * c
d = e - f
g = h / i

Assembly Code:
MOV x, R0
MOV y, R1
ADD R0, R1
MOV R0, z

MOV b, R0
MOV c, R1
MUL R0, R1
MOV R0, a

MOV e, R0
MOV f, R1
SUB R0, R1
MOV R0, d

MOV h, R0
MOV i, R1
DIV R0, R1
MOV R0, g
```