ABHINAND K PRASAD

S7 CSA

ROLL NO: 4

#include <stdio.h>

#include <string.h>

#include <stdbool.h>

char reg\_data[4][6];  // Array to store registers' data

int next\_reg = 0;  // Keep track of the next register number

// Function to check if a variable is already in a register and return its register number

int get\_reg(char\* var) {

    for (int i = 0; i < 4; i++)

        if (strcmp(var, reg\_data[i]) == 0) return i;

    return -1;

}

// Function to allocate a new register

int get\_new\_reg() { return next\_reg++; }

// Function to handle the operation based on the operator

void handle\_operation(char\* op1, char\* op2, char\* opp, int rid1, int rid2) {

    if (rid2 == -1) {

        // If operand2 is not in a register, load it based on the operator

        switch (opp[0]) {

            case '+': printf("ADD %s R%d\n", op2, rid1); break;

            case '-': printf("SUB %s R%d\n", op2, rid1); break;

            case '\*': printf("MUL %s R%d\n", op2, rid1); break;

            case '/': printf("DIV %s R%d\n", op2, rid1); break;

        }

    } else {

        // If operand2 is already in a register, perform operation with it

        switch (opp[0]) {

            case '+': printf("ADD R%d R%d\n", rid2, rid1); break;

            case '-': printf("SUB R%d R%d\n", rid2, rid1); break;

            case '\*': printf("MUL R%d R%d\n", rid2, rid1); break;

            case '/': printf("DIV R%d R%d\n", rid2, rid1); break;

        }

    }

}

// Main function to process the expressions from user input

int main() {

    char expr[100], opp[2], result[6], op1[6], op2[6];

    printf("Enter the Expressions (Type 'exit' to terminate) : \n");

    // Continuously take input until the user types 'exit'

    while (1) {

        // Get input expression

        printf("Expression: ");

        fgets(expr, sizeof(expr), stdin);

        expr[strcspn(expr, "\n")] = '\0';  // Remove trailing newline

        if (strcmp(expr, "exit") == 0) {

            break;  // Exit if user types 'exit'

        }

        // Parse the expression into the result, operand1, operator, and operand2

        if (sscanf(expr, "%s = %s %1[+-\*/%%] %s", result, op1, opp, op2) == 4) {

            // Check if operand1 is in a register

            int rid1 = get\_reg(op1);

            if (rid1 == -1) {

                rid1 = get\_new\_reg();

                printf("MOV %s R%d\n", op1, rid1);  // Move the operand into the register

            }

            // Check if operand2 is in a register

            int rid2 = get\_reg(op2);

            handle\_operation(op1, op2, opp, rid1, rid2);

            // Store the result into the corresponding register

            printf("MOV R%d %s\n", rid1, result);

            strcpy(reg\_data[rid1], result);  // Store the result in the register

        } else {

            printf("Invalid expression format. Please enter in the form 'result = op1 operator op2'.\n");

        }

    }

    return 0;

}

**OUTPUT**

A screen shot of a computer

Description automatically generated