**Ex No: 8**

**Date:**

**GENERATE THREE ADDRESS CODES**

**AIM:**

To generate three address code using C program.

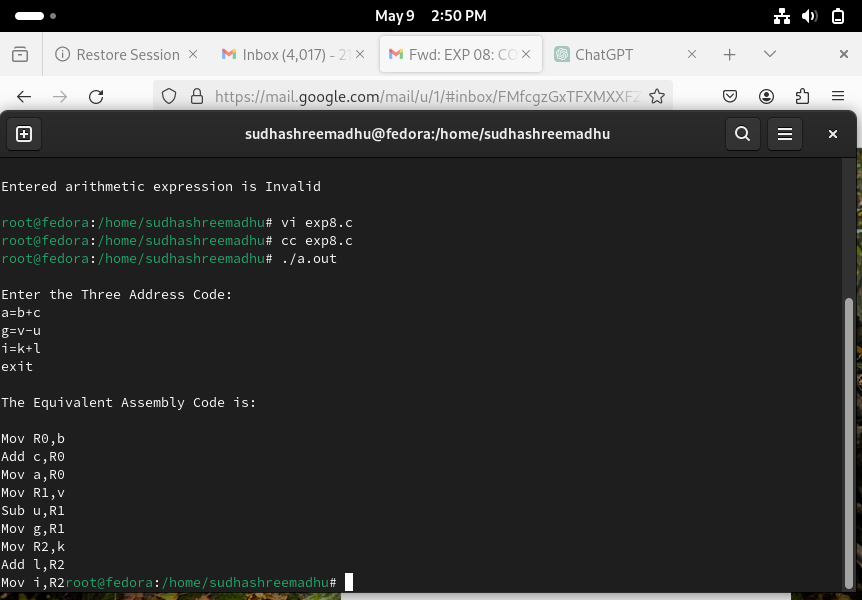
**ALGORITHM:**

* Get address code sequence.
* Determine current location of 3 using address (for 1st operand).
* If the current location does not already exist, generate move (B, O).
* Update address of A (for 2nd operand).
* If the current value of B and () is null, exist.
* If they generate operator () A, 3 ADPR.
* Store the move instruction in memory.

**PROGRAM:**

#include<stdio.h>  
#include<string.h>  
#include<ctype.h>  
  
typedef struct {  
    char var[10];  
    int alive;  
} regist;  
  
regist preg[10];  
  
void substring(char exp[], int st, int end) {  
    int i, j = 0;  
    char dup[10] = "";  
    for(i = st; i < end; i++)  
        dup[j++] = exp[i];  
    dup[j] = '\0';  
    strcpy(exp, dup);  
}  
  
int getregister(char var[]) {  
    int i;  
    for(i = 0; i < 10; i++) {  
        if(preg[i].alive == 0) {  
            strcpy(preg[i].var, var);  
            break;  
        }  
    }  
    return i;  
}  
  
void getvar(char exp[], char v[]) {  
    int i, j = 0;  
    char var[10] = "";  
    for(i = 0; exp[i] != '\0'; i++) {  
        if(isalpha(exp[i]))  
            var[j++] = exp[i];  
        else  
            break;  
    }  
    var[j] = '\0';  
    strcpy(v, var);  
}  
  
int main() {  
    char basic[10][10], var[10][10], fstr[10], op;  
    int i, j, k, reg, vc, flag = 0;  
  
    printf("\nEnter the Three Address Code:\n");  
    for(i = 0; ; i++) {  
        fgets(basic[i], sizeof(basic[i]), stdin);  
        if(strcmp(basic[i], "exit\n") == 0)  
            break;  
    }  
  
    printf("\nThe Equivalent Assembly Code is:\n");  
    for(j = 0; j < i; j++) {  
        vc = 0;  
        getvar(basic[j], var[vc++]);  
        strcpy(fstr, var[vc-1]);  
        substring(basic[j], strlen(var[vc-1]) + 1, strlen(basic[j]));  
        getvar(basic[j], var[vc++]);  
        reg = getregister(var[vc-1]);  
        if(preg[reg].alive == 0) {  
            printf("\nMov R%d,%s", reg, var[vc-1]);  
            preg[reg].alive = 1;  
        }  
  
        op = basic[j][strlen(var[vc-1])];  
        substring(basic[j], strlen(var[vc-1]) + 1, strlen(basic[j]));  
        getvar(basic[j], var[vc++]);  
        switch(op) {  
            case '+': printf("\nAdd"); break;  
            case '-': printf("\nSub"); break;  
            case '\*': printf("\nMul"); break;  
            case '/': printf("\nDiv"); break;  
        }  
  
        flag = 1;  
        for(k = 0; k <= reg; k++) {  
            if(strcmp(preg[k].var, var[vc-1]) == 0) {  
                printf("R%d, R%d", k, reg);  
                preg[k].alive = 0;  
                flag = 0;  
                break;  
            }  
        }  
  
        if(flag) {  
            printf(" %s,R%d", var[vc-1], reg);  
            printf("\nMov %s,R%d", fstr, reg);  
        }  
  
        strcpy(preg[reg].var, var[vc-3]);  
    }  
  
    return 0;  
}

**OUTPUT:**

****

**RESULT:**