Course Code and Name: 2CS701 Compiler Construction

Name and Roll number: 20bce297 Tilak Vasu

Practical 6: - Intermediate Code Generation: To generate Three Address code for assignment statement.

Code:

#*include* <stdio.h>

#*include* <string.h>

#*include* <conio.h>

#*include* <stdlib.h>

void pm();

void plus();

void divide();

int i, ch, j, l, addr = 100;

char ex[10], expr[10], expr1[10], expr2[10], id1[5], op[5], id2[5];

void pm()

{

    strrev(expr);

    j = l - i - 1;

    strncat(expr1, expr, j);

    strrev(expr1);

    printf("Three address code :\ntemp = %s\ntemp1 = %c %ctemp\n", expr1, expr[j + 1], expr[j]);

}

void divide()

{

    strncat(expr1, expr, i + 2);

    printf("Three address code :\ntemp = %s \ntemp1 = temp %c %c\n", expr1, expr[i + 2], expr[i + 3]);

}

void plus()

{

    strncat(expr1, expr, i + 2);

    printf("Three address code:\ntemp = %s \ntemp1 = temp %c %c\n", expr1, expr[i + 2], expr[i + 3]);

}

void main()

{

*while* (1)

    {

        printf("\n1.Assignment\n2.Arithmetic\n3.Relational \n4.Exit\nEnter the choice: ");

        scanf("%d", &ch);

*switch* (ch)

        {

*case* 1:

            printf("\nEnter the exprression with assignment operator: ");

            scanf("%s", expr);

            l = strlen(expr);

            expr2[0] = '\0';

            i = 0;

*while* (expr[i] != '=')

            {

                i++;

            }

            strncat(expr2, expr, i);

            strrev(expr);

            expr1[0] = '\0';

            strncat(expr1, expr, l - (i + 1));

            strrev(expr1);

            printf("Three address code :\ntemp = %s\n %s = temp\n", expr1, expr2);

*break*;

*case* 2:

            printf("\nEnter the exprression with arithmetic operator : ");

            scanf("%s", ex);

            strcpy(expr, ex);

            l = strlen(expr);

            expr1[0] = '\0';

*for* (i = 0; i < l; i++)

            {

*if* (expr[i] == '+' || expr[i] == '-')

                {

*if* (expr[i + 2] == '/' || expr[i + 2] == '\*')

                    {

                        pm();

*break*;

                    }

*else*

                    {

                        plus();

*break*;

                    }

                }

*else* *if* (expr[i] == '/' || expr[i] == '\*')

                {

                    divide();

*break*;

                }

            }

*break*;

*case* 3:

            printf(" Enter the exprression with relational operator  \n");

            scanf("%s %s %s", &id1, &op, &id2);

*if* (((strcmp(op, "<") == 0) || (strcmp(op, ">") == 0) || (strcmp(op, "<=") == 0) || (strcmp(op, ">=") == 0) || (strcmp(op, "==") == 0) || (strcmp(op, "!=") == 0)) == 0)

                printf(" exprression is error ");

*else*

            {

                printf("\n %d\t if %s %s %s goto %d ", addr, id1, op, id2, addr + 3);

                addr++;

                printf("\n%d\t T:=0", addr);

                addr++;

                printf("\n%d\t goto %d", addr, addr + 2);

                addr++;

                printf("\n%d\t T:=1", addr);

            }

*break*;

*case* 4:

            exit(0);

        }

    }

}

Output:



