Implement the back end of the compiler using C program

PROGRAM :

#include <stdio.h>

#include <string.h>

#include <math.h>

#include <stdlib.h>

int pm();

int plus();

int division();

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

char a[10], b[10], c[10], d[10], id1[5], op[5], id2[5];

int main()

{

// clrscr();

while (1)

{

printf("\n1.assignment\n2.arithmetic\n3.relational\n4.Exit\nEnter the choice:");

scanf("%d", &ch);

switch (ch)

{

case 1:

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

scanf("%s", b);

l = strlen(b);

d[0] = '\0';

i = 0;

while (b[i] != '=')

{

i++;

}

strncat(d, b, i);

strrev(b);

c[0] = '\0';

strncat(c, b, l - (i + 1));

strrev(c);

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

break;

case 2:

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

scanf("%s", a);

strcpy(b, a);

l = strlen(b);

c[0] = '\0';

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

{

if (b[i] == '+' || b[i] == '-')

{

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

{

pm();

break;

}

else

{

plus();

break;

}

}

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

{

division

();

break;

}

}

break;

case 3:

printf("Enter the expression with relational operator");

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("Expression is error");

else

{

printf("\n%d\tif %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);

}

}

}

int pm()

{

strrev(b);

j = l - i - 1;

strncat(c, b, j);

strrev(c);

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

}

int division()

{

strncat(c, b, i + 2);

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

}

int plus()

{

strncat(c, b, i + 2);

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

}

OUTPUT :

Enter the set of intermediate code :

a=a\*b

c=f\*h

g=a\*h

f=Q+W

t=q-j

target code generation

Mov a,R0

MUL b,R0

Mov R0,a

Mov f,R1

MUL h,R1

Mov a,R2

MUL h,R2

Mov Q,R3

ADD w,R3

Mov R3,f

Mov q,R4

SUB j,R4

Mov R4,t