

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Date : 29/03/2022 | | | | | | | |
|  | CSPC62 : COMPILER DESIGN  **LAB-6** | | | | | |  |
|  |  | | | | |  | |
|  | | |  |  | | | |
|  | | | Roll no. : 106119100Name : Rajneesh PandeySection : CSE-B |  | | | |
|  | |  | | |  | | |

Generate intermediate code for if, if-else and while loop in C.

Code: If-else

lexer.l

%{

#include "parser.tab.h"

#include <stdlib.h>

#include <string.h>

#include <math.h>

%}

ALPHA [A-Za-z]

DIGIT [0-9]

%%

if                 return IF;

then                 return THEN;

else                 return ELSE;

{ALPHA}({ALPHA}|{DIGIT})\*    return ID;

{DIGIT}+             {yylval=atoi(yytext); return NUM;}

[ \t]                 ;

\n                yyterminate();

.                 return yytext[0];

%%

parser.y

%{

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int yylex();

void yyerror (char const \*s) {

   fprintf (stderr, "%s\n", s);

 }

void push();

void pusha();

void pushab();

void abc();

void abcde();

void second();

void second1();

void first();

void third();

void codegen();

void codegen\_umin();

void codegen\_assigna();

void codegen\_assign();

void codegen\_assignb();

void lab();

void lab1();

void lab2();

void lab3();

%}

%token ID NUM IF THEN ELSE

%right '='

%left '+' '-'

%left '\*' '/'

%left UMINUS

%%

S : IF '(' Y ')'{lab();} THEN '{' X '}'{lab2();} ELSE '{' X '}' {lab3();}

  ;

X : E ';'|X X;

Y : B {abc();codegen\_assigna();first();}

  | B '&''&'{abc();codegen\_assigna();second();} Y

  | B {abc();codegen\_assigna();third();}'|''|' Y

  | '!'B{abcde();codegen\_assigna();first();}

  ;

B : V '='{push();}'='{push();}D

  | V '>'{push();}F

  | V '<'{push();}F

  | V '!'{push();}'='{push();}D

  |'(' B ')'

  | V{pushab();}

;

F :'='{push();}D

  |D{pusha();}

;

D :NUM{push();}

  |ID{push();}

;

E :V '='{push();} E{codegen\_assign();}

  | E '+'{push();} E{codegen();}

  | E '-'{push();} E{codegen();}

  | E '\*'{push();} E{codegen();}

  | E '/'{push();} E{codegen();}

  | '(' E ')'

  | '-'{push();} E{codegen\_umin();} %prec UMINUS

  | V

  | NUM{push();}

  | S

;

V : ID {push();}

  ;

%%

#include "lex.yy.c"

#include<ctype.h>

char st[100][10];

int top=0;

char i\_[2]="0";

char temp[2]="t";

int abcd=0;

int label[20];

int lnum=0;

int ltop=0;

int i=0;

int main()

 {

 printf("Enter the expression : ");

 yyparse();

 }

int yywrap(){return(1);}

void pusha()

{

strcpy(st[++top]," " );

}

void pushab()

{

strcpy(st[++top]," ");

strcpy(st[++top]," ");

strcpy(st[++top]," ");

}

void push()

 {

  strcpy(st[++top],yytext);

 }

void abc()

{

abcd++;

printf("\nX%d : if ",abcd);

}

void abcde()

{

abcd++;

printf("\nX%d :not ",abcd);

}

void second1()

{

printf("\nif x%d true goto L%d\n",abcd,lnum);

printf("\nif x%d false goto L%d\n",abcd,++lnum);

lnum=lnum-1;

}

void second()

{

int xyz=0;

xyz=abcd+1;

printf("falg=true else flag=false");

printf("\n if flag(true) goto x%d",xyz);

printf("\n if flag(false) goto L1");

}

void first()

{

printf("flag=true else flag=false");

printf("\n if flag(true) goto  L0");

printf("\n if flag(false) goto L1");

}

void third()

{

int xyz=0;

xyz=abcd+1;

printf("flag=true else flag=false");

printf("\n if flag(true) goto L0 ");

printf("\n if flag(false) goto x%d",xyz);

}

void codegen()

 {

 strcpy(temp,"t");

 strcat(temp,i\_);

  printf("%s = %s %s %s\n",temp,st[top-2],st[top-1],st[top]);

  top-=2;

 strcpy(st[top],temp);

 i\_[0]++;

 }

void codegen\_umin()

 {

 strcpy(temp,"t");

 strcat(temp,i\_);

 printf("%s = -%s\n",temp,st[top]);

 top--;

 strcpy(st[top],temp);

 i\_[0]++;

 }

void codegen\_assigna()

{

printf("%s %s %s %s ",st[top-3],st[top-2],st[top-1],st[top]);

top-=3;

}

void codegen\_assign()

 {

 printf("%s = %s\n",st[top-2],st[top]);

 top-=2;

 }

void codegen\_assignb()

{

printf("%s %s %s ",st[top-3],st[top-2],st[top-1]);

top-=3;

}

void lab()

{

printf("\nL0 :\n");

}

void lab1()

{

 strcpy(temp,"t");

 strcat(temp,i\_);

 printf("\n%s = not arguement \n",temp);

 printf("if %s goto L%d\n",temp,lnum);

 i\_[0]++;

 label[++ltop]=lnum;

}

void lab2()

{

int x;

lnum++;

x=label[ltop--];

printf("goto L2\n");

printf("L%d: \n",++x);

label[++ltop]=lnum;

}

void lab3()

{

int y;

y=label[ltop--];

printf("L2: \n");

}

Input:

if(a>1){a=b+1;}else{b=a+1;}

Run :

A picture containing text

Description automatically generated

Output :

Text

Description automatically generated

Code: While

lexer.l

%{

#include"parser.tab.h"

%}

ALPHA [A-Za-z]

DIGIT [0-9]

%%

while                return WHILE;

{ALPHA}({ALPHA}|{DIGIT})\*    return ID;

{DIGIT}+             {yylval=atoi(yytext); return NUM;}

[ \t]                 ;

\n                yyterminate();

.                 return yytext[0];

%%

parser.y

%{

#include "lex.yy.c"

#include<ctype.h>

#include <string.h>

int yylex();

void yyerror (char const \*s) {

   fprintf (stderr, "%s\n", s);

 }

void push();

void codegen()  ;

void codegen\_umin();

void codegen\_assign();

void lab1();

void lab2();

void lab3();

%}

%token ID NUM WHILE

%right '='

%left '+' '-'

%left '\*' '/'

%left UMINUS

%%

S : WHILE{lab1();} '(' E ')'{lab2();} E ';'{lab3();}

  ;

E :V '='{push();} E{codegen\_assign();}

  | E '+'{push();} E{codegen();}

  | E '-'{push();} E{codegen();}

  | E '\*'{push();} E{codegen();}

  | E '/'{push();} E{codegen();}

  | '(' E ')'

  | '-'{push();} E{codegen\_umin();} %prec UMINUS

  | V

  | NUM{push();}

  ;

V : ID {push();}

  ;

%%

char st[100][10];

int top=0;

char i\_[2]="0";

char temp[2]="t";

int lnum=1;

int start=1;

int main()

 {

 printf("Enter the expression : ");

 yyparse();

 }

int yywrap(){

  return(1);

}

void push()

 {

  strcpy(st[++top],yytext);

 }

void codegen()

 {

 strcpy(temp,"t");

 strcat(temp,i\_);

  printf("%s = %s %s %s\n",temp,st[top-2],st[top-1],st[top]);

  top-=2;

 strcpy(st[top],temp);

 i\_[0]++;

 }

void codegen\_umin()

 {

 strcpy(temp,"t");

 strcat(temp,i\_);

 printf("%s = -%s\n",temp,st[top]);

 top--;

 strcpy(st[top],temp);

 i\_[0]++;

 }

void codegen\_assign()

 {

 printf("%s = %s\n",st[top-2],st[top]);

 top-=2;

 }

void lab1()

{

printf("L%d: \n",lnum++);

}

void lab2()

{

 strcpy(temp,"t");

 strcat(temp,i\_);

 printf("%s = not %s\n",temp,st[top]);

 printf("if %s goto L%d\n",temp,lnum);

 i\_[0]++;

 }

void lab3()

{

printf("goto L%d \n",start);

printf("L%d: \n",lnum);

}

input

while(k=c/s)k=k\*c+d;

run

Text

Description automatically generated

Output

Text

Description automatically generated