%{

#include<stdio.h>

#include<string.h>

int multi\_flag=0,count=0,addr=2000,assign=0,id\_flag=0,dec=0;

char cur\_type[20];

struct symTable{

char sym[20];

int address;

char type[20];

char value[20];

int size;

}tab[10];

void constab(char\*);

void reset\_flag();

void update(char\*);

%}

numberC ([0-9]+)|([0-9]\*(\.)[0-9]+)

charC '(.)'

stringC \"((.)\*)\"

relop ("<"|"<="|"=="|"!="|">"|">=")

arithop ("+"|"-"|"\*"|"/"|"%")

logop ("&&"|"||"|"!")

assign "="

multi\_s \/\\*

multi\_e \\*\/

single "//".\*

SP \(|\)|\{|\}|","|";"

KW if|else|while|do|for|break|continue|switch

data\_type int|float|char|double

function ([ \_ a-z A-Z ])([ \_ a-z A-Z 0-9 ])\*\((.)\*\)

id ([\_a-zA-Z])([\_a-zA-Z0-9])\*

newline "\n"

%%

{numberC} {if(!multi\_flag) printf("NUMCONST ");if(assign==1){update(yytext);reset\_flag();}}

{charC} {if(!multi\_flag) printf("CHARCONST ");if(assign==1){update(yytext);reset\_flag();}}

{stringC} {if(!multi\_flag) printf("STRCONST ");if(assign==1){update(yytext);reset\_flag();}}

{relop} {if(!multi\_flag) printf("RELOP ");reset\_flag();}

{arithop} {if(!multi\_flag) printf("ARITHOP ");reset\_flag();}

{logop} {if(!multi\_flag) printf("LOGOP ");reset\_flag();}

{assign} {if(!multi\_flag) printf("ASSIGN "); if(id\_flag==1) assign=1; }

{multi\_s} {if(!multi\_flag) printf("MULTI\_S ");multi\_flag=1;reset\_flag();}

{multi\_e} {multi\_flag=0;if(!multi\_flag) printf("MULTI\_E "); reset\_flag();}

{single} {if(!multi\_flag) printf("SINGLE ");reset\_flag();}

{SP} {if(!multi\_flag) printf("SP ");reset\_flag();}

{KW} {if(!multi\_flag) printf("KW "); reset\_flag();}

{data\_type} {if(!multi\_flag) printf("KW"); strcpy(cur\_type,yytext); dec=1; }

{function} {if(!multi\_flag) printf("FN ");reset\_flag();}

{newline} {if(!multi\_flag) printf("\n"); dec=0;}

{id} {if(!multi\_flag) printf("ID "); if(dec==1){constab(yytext); id\_flag=1;}}

%%

void main()

{

int i;

FILE \*fp = fopen("test\_file1.c", "r");

if(fp == NULL)

{

perror("Unable to open file!");

exit(1);

}

yyin=fp;

yylex();

printf("\nNAME TYPE SIZE ADDR VALUE\n");

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

{

printf("%4s %6s %4d %4d %s\n",tab[i].sym,tab[i].type,tab[i].size,tab[i].address,tab[i].value);

}

}

void update(char\* val)

{

strcpy(tab[count-1].value,val);

}

void reset\_flag()

{

assign=0;

id\_flag=0;

}

void constab(char\* id)

{

strcpy(tab[count].sym,id);

strcpy(tab[count].type,cur\_type);

tab[count].address=addr;

if(!strcmp(cur\_type,"int"))

{

addr+=2;

tab[count].size = 2;

}

else if(!strcmp(cur\_type,"float"))

{

addr+=4;

tab[count].size =4;

}

else if(!strcmp(cur\_type,"char"))

{

addr+=1;

tab[count].size =1;

}

else if(!strcmp(cur\_type,"double"))

{

addr+=8;

tab[count].size =8;

}

count++;

update("-");

}

int yywrap(void){}

/\*

MULTI\_S

MULTI\_E

FN

SP

KW ID ASSIGN NUMCONST SP ID ASSIGN NUMCONST SP

SINGLE

KW 'SP ID RELOP ID SP

FN SP

KW

FN SP

SP

FN

SP

SP

NAME TYPE SIZE ADDR VALUE

ab int 2 2000 10

b int 2 2002 20

c char 1 2004 'a'

d float 4 2005 9.134

e float 4 2009 -

\*/