Sharvan Ram Kumaran

Reg No:185001143

CSE C

**Exercise 2**

Objective:

Develop a Lexical analyzer to recognize the patterns namely, identifiers, constants, comments and operators using the following regular expressions. Construct symbol table for the identifiers with the following information.

**Code:**

%{

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int i = 0;

int j = 0;

int flag = 0;

int flag1 = 0;

int addr = 1000;

char buf[100];

typedef struct

{

char type[100];

char name[100];

char value[100];

}table;

table st[100];

void func(char str[],char type[]){

if(type=="KW"){

if(strcmp(str,"int")==0||strcmp(str,"float")==0||strcmp(str,"char")==0){

strcpy(buf,str);

flag1 = 1;

}

}

else if(flag1 == 0)return;

else if(type == "ID"){

if(flag==1){

i++;

strcpy(st[i].value,"--");

}

flag = 1;

strcpy(st[i].name ,str);

strcpy(st[i].type ,buf);

}

else if(type=="VAL"){

strcpy(st[i].value,str);

i++;

flag = 0;

strcpy(st[i].value,"--");

}

else if(strcmp(str,";")==0){

if(flag==1){

i++;

strcpy(st[i].value,"--");

}

flag1 = 0;

printf("HI\n");

}

}

%}

assign\_ops ("="|"+="|"-="|"%="|"/="|"\*=")

rel\_ops ("<"|">"|"<="|">="|"=="|"!=")

arith\_ops ("+"|"-"|"%"|"/"|"\*")

spl\_chars [{}(),;\[\]]

keyword (int|float|char|unsigned|typedef|struct|return|continue|break|if|else|for|while|do|extern|auto|case|switch|enum|goto|long|double|sizeof|void|default|register)

int\_const [0-9]+

float\_const [0-9]+.[0-9]+

char\_const [\'].[\']

string\_const [\"].\*[\"]

identifier [a-zA-Z\_][a-zA-Z0-9\_]\*

function [a-zA-Z\_][a-zA-Z0-9]\*[(].\*[)]

single\_cmt [/][/].\*

/\*Rules\*/

%%

{keyword} {

printf("KW ");

func(yytext,"KW");

}

{function} printf("FUNCT ");

{identifier} {

printf("ID ");

func(yytext,"ID");

}

{single\_cmt} printf("SCMT ");

{int\_const} {

printf("INT\_CONST ");

func(yytext,"VAL");

}

{float\_const} {

printf("FLT\_CONST ");

func(yytext,"VAL");

}

{char\_const} {

printf("CHAR\_CONST ");

func(yytext,"VAL");

}

{string\_const} {

printf("STR\_CONST ");

func(yytext,"VAL");

}

{rel\_ops} printf("REL\_OP ");

{arith\_ops} printf("ARITH\_OP ");

{assign\_ops} printf("ASSIGN\_OP ");

{spl\_chars} {

printf("SP");

func(yytext,"SP");

}

\n {

printf("\n");

}

[ \t] { }

%%

int yywrap(void){

return 1;

}

int main(int argc, char \*argv[]){

strcpy(st[0].value,"undefined");

yyin = fopen(argv[1], "r");

yylex();

printf("TYPE\tNAME\tVALUE\tSIZE\tADDR\n");

for(;j<i;j++){

printf("%s\t%s\t%s\t",st[j].type,st[j].name,st[j].value);

if(strcmp(st[j].type,"int")==0){printf("2\t");addr+=2;}

else if(strcmp(st[j].type,"float")==0){printf("4\t");addr+=4;}

else {printf("1\t");addr++;}

printf("%d\n",addr);

}

return 0;

}

**Output:**

**Text

Description automatically generated**