AKSHAY G BL.EN.U4CSE14005

/\* lex program \*/

%{

#include<stdio.h>

%}

%%

IF|ELSE|FOR|WHILE printf("Keyword:%s\n",yytext);

"&&"|"||"|"!" printf("Logical operators:%s\n",yytext);

"<"|">"|"<="|">=" printf("Relational operators:%s\n",yytext);

"+"|"-"|"\*"|"/" printf("Arithmetic operators:%s\n",yytext);

"//".\* printf("comment:%s\n",yytext);

" " printf("space:%s\n",yytext);

"/\*"([a-z A-Z 0-9]+|"\t"|"\n") printf("Multiline:%s\n",yytext);

"#include<"[a-z A-Z 0-9]+".h>" printf("Preprocessor Directives:%s\n",yytext);

"<".\*".h>" printf("Headerfile:%s\n",yytext);

[0-9]+ printf("Integer:%s\n",yytext);

[0-9]+"."[0-9]+ printf("Float:%s\n",yytext);

[a-z A-Z \_][a-z A-Z 0-9]+ printf("Identifier:%s\n",yytext);

%%

int main()

{

yylex();

return 0;

}

OUTPUT:

123

Integer:123

avd

Identifier:avd

#include<d.h>

Preprocessor Directives:#include<d.h>

1.2

Float:1.2

IF

Keyword:IF

1+2

Integer:1

Arithmetic operators:+

Integer:2

1>4

Integer:1

Relational operators:>

Integer:4

//abc

comment://abc

AKSHAY G BL.EN.U4CSE14005

/\* lex program using external files \*/

%{

#include<stdio.h>

%}

%%

IF|ELSE|FOR|WHILE printf("Keyword:%s\n",yytext);

"&&"|"||"|"!" printf("Logical operators:%s\n",yytext);

"<"|">"|"<="|">=" printf("Relational operators:%s\n",yytext);

"+"|"-"|"\*"|"/" printf("Arithmetic operators:%s\n",yytext);

"//".\* printf("comment:%s\n",yytext);

" " printf("space:%s\n",yytext);

"/\*"([a-z A-Z 0-9]+|"\t"|"\n") printf("Multiline:%s\n",yytext);

"#include<"[a-z A-Z 0-9]+".h>" printf("Preprocessor Directives:%s\n",yytext);

"<".\*".h>" printf("Headerfile:%s\n",yytext);

[0-9]+ printf("Integer:%s\n",yytext);

[0-9]+"."[0-9]+ printf("Float:%s\n",yytext);

[a-z A-Z \_][a-z A-Z 0-9]+ printf("Identifier:%s\n",yytext);

%%

int main()

{

yyin = fopen("abc.txt","r");

yyout = fopen("xyz.txt","w");

yylex();

fclose(yyin);

fclose(yyout);

return 0;

}

OUTPUT:

//abc.txt

123

avd

#include<d.h>

1.2

IF

1+2

1>4

//abc

//xyz.txt

Integer:123

Identifier:avd

Preprocessor Directives:#include<d.h>

Float:1.2

Keyword:IF

Integer:1

Arithmetic operators:+

Integer:2

Integer:1

Relational operators:>

Integer:4

comment://abc

AKSHAY G BL.EN.U4CSE14005

/\* (a+b)\*abb \*/

#include <iostream>

using namespace std;

int nodeId = 1;

class Node{

public:

Node \*lptr,\*rptr;

int id;

int visited;

char data;

int lvalue,rvalue;

Node(){

lvalue = -1;

rvalue = -1;

lptr = NULL;

rptr = NULL;

id = nodeId++;

visited = 0;

}

};

class List{

public:

Node \*head;

List(){

head = NULL;

}

void createNode(char info){

Node \*temp = new Node();

temp->data = info;

head = temp;

Node \*temp2 = new Node();

temp2->data = '$';

temp->rptr = temp2;

temp->rvalue = temp2->id;

}

void dot(List l){

Node \*temp1 = head;

Node \*temp2 = l.head;

while(temp1->rptr->data!='$'){

temp1 = temp1->rptr;

}

temp1->rptr = temp2;

temp1->rvalue = temp2->id;

temp1 = head;

if(temp1->lptr != NULL){

temp1 = temp1->lptr;

while(temp1->rptr->data!='$'){

temp1 = temp1->rptr;

}

temp1->rptr = temp2;

temp1->rvalue = temp2->id;

}

}

void plus(List l){

Node \*l1 = head;

Node \*l2 = l.head;

Node \*temp1 = new Node();

temp1->data = '#';

temp1->lptr = head;

temp1->lvalue = l1->id;

temp1->rptr = l2;

temp1->rvalue = l2->id;

Node \*temp2 = new Node();

temp2->data = '$';

while(l1->data!='$')

l1 = l1->rptr;

while(l2->data!='$')

l2 = l2->rptr;

l1->data = '#';

l2->data = '#';

l1->rptr = temp2;

l1->rvalue = temp2->id;

l2->rptr = temp2;

l2->rvalue = temp2->id;

head = temp1;

}

void star(){

Node \*temp1 = new Node();

temp1->data = '#';

Node \*temp2 = new Node();

temp2->data = '$';

Node \*cur = head;

temp1->lptr = cur;

temp1->lvalue = cur->id;

temp1->rptr = temp2;

temp1->rvalue = temp2->id;

while(cur->rptr!=NULL){

cur = cur->rptr;

}

cur->data = '#';

cur->rptr = temp2;

cur->rvalue = temp2->id;

cur->lptr = head;

cur->lvalue = head->id;

head = temp1;

}

void display(){

BFS(head);

initVisited(head);

}

void BFS(Node \*h){

if(h==NULL){

return;

}

if(h->visited!=1)

cout<<h->data<<" "<<h->id<<" "<<h->lvalue<<" "<<h->rvalue<<endl;

else

return;

h->visited = 1;

BFS(h->lptr);

BFS(h->rptr);

}

void initVisited(Node \*h){

if(h == NULL)

return;

if(h->visited == 0)

return;

h->visited = 0;

initVisited(h->lptr);

initVisited(h->rptr);

}

};

int main(){

List l1,l2,l3,l4,l5;

l1.createNode('a');

l2.createNode('b');

l3.createNode('a');

l4.createNode('b');

l5.createNode('b');

l4.dot(l5);

l1.plus(l2);//a+b

l1.star();//(a+b)\*

l1.dot(l3);//(a+b)\*.a

l1.dot(l4);//(a+b)\*.a.b.b

l1.display();

return 0;}

OUTPUT:

# 13 11 5

# 11 1 3

a 1 -1 2

# 2 -1 12

# 12 11 5

a 5 -1 7

b 7 -1 9

b 9 -1 7

b 3 -1 4

# 4 -1 12