

**Ahsanullah University of Science and Technology (AUST)**

Department of Computer Science and Engineering

**Course No:** CSE4130

**Course Title:** Formal Languages and Compilers Lab

**Assignment Number:** 02

|  |  |
| --- | --- |
|  |  |

**Name:** Devopriya Tirtho

**ID:** 16.02.03.033

**Section:** A

**Lab Group:** A2

**Question:**

**Suppose, we have a C source program scanned and filtered as it was done in Session 1. We now take that modified file as input, and separate the lexemes first. We further recognize and mark the lexemes as different types of tokens like keywords, identifiers, operators, separators, parenthesis, numbers, etc.**

**Solution Code:**

/\*

char c; int x1, x\_2; float y1, y2; x1=5; x\_2= 10; y1=2.5+x1\*45; y2=100.o5-x\_2/3; if(y1<=y2) c='y'; else

c='n';

\*/

#include<stdio.h>

#include<ctype.h>

int main(void)

{

FILE \*p1,\*p2,\*p3;

char c;

int x=0;

int flag=0;

int y=1;

char m;

int kw1=0;

int kw2=0;

int kw3=0;

int kw4=0;

int kw5=0;

int kw6=0;

p1=fopen("input.txt","r");

p2=fopen("out.txt","w");

if(!p1)

{

printf("there is no input file");

}

else

{

while((c=fgetc(p1))!=EOF)

{

//if new line then space

if(c=='\n')

{

c=' ';

fputc(c,p2);

x=1;

}

//for seperating operators and seperators

if((c==';')||(c==39)||(c=='<')||(c==',')||(c=='=')||(c=='==')||(c=='+')||(c=='-')||(c=='\*')||(c=='/')||(c=='(')||(c==')'))

{

fputc(' ',p2);

y=0;

}

if(x==0)

{

fputc(c,p2);

if(y==0)

{

fputc(' ',p2);

}

}

x=0;

y=1;

}

}

fclose(p1);

fclose(p2);

//open modified file

p3=fopen("out.txt","r");

while((c=fgetc(p3))!=EOF)

{

//for operators

if((c=='=')||(c=='+')||(c=='-')||(c=='\*')||(c=='/')||(c=='<')||(c=='>'))

{

printf("[op %c]",c);

}

//for seperators

if((c==';')||(c==39)||(c==','))

{

printf("[sep %c]",c);

}

//for parenthesis

if((c=='(')||(c==')'))

{

printf("[par %c]",c);

}

//for keywords

if(c=='c')

{

c=fgetc(p3);

if(c=='h')

{

c=fgetc(p3);

if(c=='a')

{

c=fgetc(p3);

if(c=='r')

{

if((c=fgetc(p3))==' ')

{

printf("[kw char]");

c=fgetc(p3);

}

}

else

printf("[id cha]");

}

else

printf("[id ch]");

}

else

printf("[id c]");

}

if(c=='f')

{

c=fgetc(p3);

if(c=='l')

{

c=fgetc(p3);

if(c=='o')

{

c=fgetc(p3);

if(c=='a')

{

c=fgetc(p3);

if(c=='t')

{

if((c=fgetc(p3))==' ')

{

kw2=1;

printf("[kw float]");

}

}

else

printf("[id floa]");

}

else

printf("[id flo]");

}

else

printf("[id fl]");

}

else

printf("[id f]");

}

if(c=='i')

{

c=fgetc(p3);

if((c=='n')||(c=='f'))

{

if(c=='f')

{

if((c=fgetc(p3))==' ')

{

kw3=1;

printf("[kw if]");

}

}

if(c=='n')

{

c=fgetc(p3);

if(c=='t')

{

if((c=fgetc(p3))==' ')

{

kw4=1;

printf("[kw int]");

}

}

else

printf("[id in]");

}

}

else

printf("[id i]");

}

if(c=='e')

{

c=fgetc(p3);

if(c=='l')

{

c=fgetc(p3);

if(c=='s')

{

c=fgetc(p3);

if(c=='e')

{

if((c=fgetc(p3))==' ')

{

kw5=1;

printf("[kw else]");

}

}

else

printf("[id els]");

}

else

printf("[id el]");

}

else

printf("[id e]");

}

//for identifiers

else if(isalpha(c)||(c=='\_'))

{

printf("[id %c",c);

while((c=fgetc(p3))!=' ')

{

if(isalpha(c)||isdigit(c)||(c=='\_'))

{

printf("%c",c);

}

}

printf("]");

}

//for numbers

if(isdigit(c)||c=='.')

{

if(c=='.')

{

c=fgetc(p3);

if(c==' ')

{

printf("[invalid]");

}

}

else

{

printf("[num %c",c);

while((c=fgetc(p3))!=' ')

{

if(!(isdigit(c)||c=='.'))

{

flag=1;

}

if(flag==0)

{

printf("%c",c);

}

}

if(flag==1)

{

printf("[unknown]",c);

}

flag=0;

printf("]");

}

}

}

fclose(p3);

return 0;

}