PROGRAMLAMA

DİLLERİ PROJE-1

**LEXİCAL ANALYZER**

SAIFULLAH NASRULLAH 05120000826

**Rapor Analizi**

Kullanıcıdan daha önceden var olan bir dosyanın ismi alınarak içinde ki kodlar satır satır okunur. Sonrasında  BigAdd diline uygun olarak kodlar lexemlere ayrılarak lexemeler listesi ekran da gösterilir ve aynı zamanda dosya\_ismi.txt uzantılı bir dosyaya ekran çıktıları da kaydedilir. Bu işlemler C dilinde gerçekleştirilir.

**Problemin çözüm yöntemi.**

 BigAdd dilinin kurallarına uygun olarak öncelikli olan şeyler kontrol edilir. Bir satır okunduğunda satırdaki karakterler üzerinde işlem yapılır.daha sonra tüm Keywordlar,indentifierler ve parantezler ekrana ve aynı zamanda dosyaya yazdırılır.

**EKRAN ÇIKTILARI**

**1.**

****

**2.**

**a.ba İÇERİĞİ**

int size.

int sum.

move 5 to size.

loop size times

[ out size, newline.

add size to sum.

]

out newline, “Sum:”, sum.

{sonuc burada}

int no1.

int no2.

add no1 to no2.

sub no1 from no2.

ADDITION OR UNARY + (+)

**b.txt İÇERİĞİ**

Lexical Analyzer for the a.ba file:

The list of Lexemes within its type:

Lexeme Type

----------- ---------------------

int Data Type

size identifier

to Assignment Operator

size identifier

time identifier

add Addition statement

size identifier

to Assignment Operator

sum identifier

out Output statement

newline identifier

sonuc burada <-- Comment which has been Written

int Data Type

no1 identifier

to Assignment Operator

no2 identifier

Open Block Parenthesis Close Block Parenthesis

---------------------- -----------------------

1 1

Open Comment Parenthesis Close Comment Parenthesis End Of Line

------------------------ --------------------- ---------------

1 1 10

8 identifier is found

7 Keywords is found

**SOURCE CODE**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include<conio.h>

int main()

{

system("COLOR 0E");

FILE \*fptr;

FILE \*yaz\_dos;

char file\_name[]=" .ba";

char c,ch='\0';

char \*identifier="identifier";

char Keywords[9][6]= {"out","int","loop","move","add","sub","to","times"};

char lexim\_types[9][50]= {"Output statement","Data Type","Loop","Assignment Statement","Addition statement",

"Subtraction statement","Assignment Operator","loop condition"

};

int i=0,j,a=0,opnblockPrts=0,clblkprs=0,KeywodSay=0;

int eofLineCount=0,undeclared\_error=0,clcmntprts=0,opncmntprths=0,idenCount=0;

printf("Please Enter the file. ");

printf("(x).ba : ");

scanf("%c",&c);

file\_name[0]=c;

if( (fptr=fopen(file\_name,"r"))==NULL )

{

printf("The File is not found.");

return;

}

else

{

yaz\_dos = fopen("b.txt","w");//dosya yaratýlýyor..

printf("Lexical Analyzer for the %s file:\n\nThe list of Lexemes within its type:\n",file\_name);

printf("Lexeme Type\n");

printf("----------- ---------------------\n");

fseek(fptr,0,SEEK\_SET);

fseek(yaz\_dos,0,SEEK\_SET);

fprintf(yaz\_dos, "Lexical Analyzer for the %s file:\n\nThe list of Lexemes within its type:\n",file\_name);

fprintf(yaz\_dos, "Lexeme Type\n----------- ---------------------\n");//yaratýlan dosyaya yazdýrlýyor..

while(!feof(fptr))

{

char str[50]="";

ch=fgetc(fptr);

for(j=0; ch!=32 && ch!=46 && ch!='\n' && !(feof(fptr)); j++)

{

str[j]=ch;

ch=fgetc(fptr);

if(ch==32){

for(i=0; i<9; i++)

{

if(strcmp(str,Keywords[i])==0)

{

printf("%-13s %s\n", str, lexim\_types[i]);

fprintf(yaz\_dos, "%-13s %s\n",str, lexim\_types[i]);

if(strcmp(str,"int")==0 || strcmp(str,Keywords[i])==0)

{

ch=fgetc(fptr);

j=0;

while(!feof(fptr))

{

str[j]=ch;

ch=fgetc(fptr);

if(ch==32 || ch==46 || ch=='[' || ch=='{' || ch==',')

{

printf("%-13s %s\n",str,identifier);

fprintf(yaz\_dos,"%-13s %s\n",str,identifier);

idenCount++;

break;

}

j++;

}

KeywodSay++;

}

}

}

}else if(ch=='\n'){

ch=fgetc(fptr);

if(ch=='[')

{

opnblockPrts++;

ch=fgetc(fptr);

a=0;

while(ch!=']')

{

str[j]=ch;

ch=fgetc(fptr);

if((ch==32 || ch==46) && strcmp(str,Keywords[a])==0)

{

printf("%-13s %s\n", str, lexim\_types[a]);

fprintf(yaz\_dos, "%-13s %s\n",str, lexim\_types[a]);

KeywodSay++;

break;

}

else

{

printf("%-13s %s\n",str,identifier);

fprintf(yaz\_dos,"%-13s %s\n",str,identifier);

idenCount++;

break;

}

a++;

}

clblkprs++;

}

}

}if(ch==46){

eofLineCount++;

ch=fgetc(fptr);

if(ch=='\n')

{

ch=fgetc(fptr);

if(ch=='{')

{

opncmntprths++;

ch=fgetc(fptr);

while(ch!='}')

{

printf("%c",ch);

fprintf(yaz\_dos,"%c",ch);

ch=fgetc(fptr);

}

clcmntprts++;

printf("\t <-- Comment which has been Written\n");

fprintf(yaz\_dos,"\t <-- Comment which has been Written\n");

ch=fgetc(fptr);

}

}

}

}

printf("\n");

printf("Open Block Parenthesis Close Block Parenthesis\n---------------------- -----------------------\n");

fprintf(yaz\_dos,"Open Block Parenthesis Close Block Parenthesis\n---------------------- -----------------------\n");

printf("%10d %20d\n",opnblockPrts,clblkprs);

fprintf(yaz\_dos,"%10d %20d\n",opnblockPrts,clblkprs);

printf("Open Comment Parenthesis Close Comment Parenthesis End Of Line\n------------------------ --------------------- ---------------\n");

fprintf(yaz\_dos,"Open Comment Parenthesis Close Comment Parenthesis End Of Line\n------------------------ --------------------- ---------------\n");

fprintf(yaz\_dos,"%10d %20d %25d\n\n",opncmntprths,clcmntprts,eofLineCount);

printf("%10d %20d %25d\n\n",opncmntprths,clcmntprts,eofLineCount);

printf("%d identifier is found\n",idenCount);

fprintf(yaz\_dos,"%d identifier is found\n",idenCount);

printf("%d Keywords is found\n",KeywodSay);

fprintf(yaz\_dos,"%d Keywords is found\n",KeywodSay);

getch();

fclose(fptr);

}

return 0;

}