PROGRAMMİNG LANGUAGES

**PROJE NO – ADI**

Writing a lexical analyzer for BigAdd Language

**ÖĞRENCİ NO – AD SOYAD**

05140000680 – Emin Soner TÜRK

05150000676 – Sefer KARA

**TESLİM TARİHİ:** 15.05.2017

**İçindekiler**

1. Analiz....................................................................................................3

2.Tasarım..................................................................................................4

2.1. Kaynak Kodlar.................................................................................4

3.Programcı Kataloğu.............................................................................17

4.Test Verileri ve Ekran Görüntüleri.......................................................17

**1. Analiz**

Bizden, BigAdd dili için bir lexical analyzer yazmamız bekleniyor.

Bu lexical analyzer dosya adı alıp, bu dosyanın içinde bulunan sözcükleri türlerine göre ayırıp başka bir dosyaya kaydetmektedir.

Bu türler: keyword, identifier ve operatörlerdir.

Dilin başlıca özelliklerine değinecek olursak:

BigAdd dili, sadece int tipi verilerden oluşan bir dildir. Bu dilde atama, toplama , çıkarma işlemleri ve döngü , kod blokları ve yorum satırları bulunmaktadır.

Dil tek bir tane (-) operatörü kullanırken, (+) operatörüne kapalıdır. Dilimiz gerçek sayıları desteklememektedir.

Dildeki değişkenler sadece harfle başlayabilirken devamında (\_) veya rakam içerebilmektedir. Ayrıca değişkenler büyük ve küçük harflere karşı duyarlıdırlar.

Dilde döngü yapısı bulunur. Kod blokları sayesinde döngü içinde döngü veya döngü içinde kod bloğu bulunabilir.

Dilde yorum satırları bulunur ve lexical analyzer işletilirken bu satırlar görmezden gelinir. Dil iç içe yorum satırları içermemektedir.

Ayrıca dilde “ “ karakterleri içinde yazılmış string sabitleri de bulunmaktadır.

**2. TASARIM**

**Kaynak Kod:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

FILE \*ptr\_2;

FILE \*ptr\_1;

int lexeme\_kontrol(char dizi[],int k);

int out\_mu(char dizi[]);

int keywords(char dizi[]);

int gecerli\_mi(char hrf);

int gecerli\_mi(char hrf);

int main() {

FILE \*ptr, \*kontrol\_ptr;

int sayilar[4]={0}; // 0). keywords , 1).identifiers , 2).paranthesis , 3).end of line.

char dosya\_adi[20];

char karakter, bayrak;

int i=0,key=0,sayac=0,k=0,satir\_say=1, flag=0, flag\_2=0, yorum\_flag=0, yorum\_kapandi\_mi=0, flag\_1=0, koseli\_parantez\_flag=0;

printf("Dosya Adini Giriniz:"); //kullanicidan dosya adini istiyoruz

scanf("%s",dosya\_adi);

while(dosya\_adi[k]!='\0')

k++;

dosya\_adi[k]='.';

dosya\_adi[k+1]='b';

dosya\_adi[k+2]='a';

dosya\_adi[k+3]='\0';

ptr=fopen(dosya\_adi,"r");

kontrol\_ptr=fopen(dosya\_adi, "r");

if(ptr==NULL)//dosya açilmadi ise...

printf("Dosya acilamadi!\n");

else//dosya açildiysa...

{

ptr\_2=fopen("a.txt","w");

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

while(karakter!=EOF)//dosya sonuna kadar okuyabilmem için gerekli

{

char lexeme[100]= {'\0'};

if(isdigit(karakter))//okunan karakterin sayi olma olasiligi

{

i=0;

lexeme[i++]=karakter;//eger karakter bir rakamsa lexeme dizimize ekliyoruz.

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);//ve bir sonraki karakteri aliyoruz.

while((isdigit(karakter) && karakter!=10)||karakter=='.')//karakter rakam oldugu sürece ve null olmadiği sürece diziye eleman ekle

{

lexeme[i++]=karakter;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

if(isalpha(lexeme[0])==0 && isalpha(karakter)>0 ){ // değişken ismi kontrolü

fprintf(ptr\_2,"%d. satirda hata! degisken isimleri harfle baslamalidir.\n",satir\_say);

printf("%d. satirda hata! degisken isimleri harfle baslamalidir.\n", satir\_say);

karakter=getc(ptr);

while(karakter!='\n')

karakter=getc(ptr);

continue;

}

lexeme[i]='\0';//ve lexeme dizisini sonuna null ata,lexeme dizisini yazdirirken önemli..

if(karakter=='\n')//eger satir sonu isaretini gördüysen

satir\_say++;//satir sayisini arttir.Böylece nerde hata olduğunu bilebiliriz.

if(lexeme\_kontrol(lexeme,i+1)==1)//lexeme dizisinin int const mu real const mu olduğunu bulan fonksiyon

{

// Dosyaya hata yazilacak mi???

fprintf(ptr\_2,"%d. satirda Hata bulundu. Bu lexical analyzer boyle bir sayiyi desteklememektedir. (%s)\n", satir\_say, lexeme);

printf("%d. satirda Hata bulundu. Bu lexical analyzer boyle bir sayiyi desteklememektedir. (%s)\n", satir\_say, lexeme);

}

else

{

fprintf(ptr\_2,"int constant (%s)\n",lexeme);

printf("int constant (%s)\n",lexeme);

}

continue;

}

if(isalpha(karakter) || karakter=='{' || karakter=='}')//eger ki karakter harf ise

{

i=0;

if(karakter=='{') //yani yorum satırı ise

{

yorum\_kapandi\_mi=1;

yorum\_flag=1;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

while(karakter!='}' ) //satir sonunu görene kadar devam et.

{

while(1){

if(flag\_1==1)

break;

if(flag\_2==1)

break;

if(bayrak=='}'){

yorum\_kapandi\_mi=0;

long deger = ftell(ptr);

fseek(kontrol\_ptr, deger, SEEK\_SET);

flag\_2=1;

break;

}

if(bayrak==EOF){

long deger = ftell(ptr);

fseek(kontrol\_ptr, deger, SEEK\_SET);

flag\_1=1;

break;

}

bayrak=getc(kontrol\_ptr);

}

karakter=getc(ptr);

if(karakter=='\n'){//eger satir sonunu görürsen

satir\_say++;//satir sayisini arttir

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

}

// bayrak = getc(kontrol\_ptr);

if(yorum\_kapandi\_mi==1)

yorum\_flag=0;

karakter= getc(ptr);

continue;

}else if(yorum\_kapandi\_mi!=0 && karakter=='}'){

if(karakter=='}'){

fprintf(ptr\_2,"%d. satirda tanimsiz ifade.\n", satir\_say);

printf("%d. satirda tanimsiz ifade.\n", satir\_say);

}

karakter= getc(ptr);

}

lexeme[i++]=karakter;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

sayac=1;//en başta bir karakter aldigimiz için sayac 1 den baslar.

while((isalpha(karakter)&&karakter!=46)||karakter=='\_' ||isdigit(karakter))

{

sayac++; //kelimenin uzunluğunu tutmamız icin gerekli olan sayac

lexeme[i++]=karakter;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

bayrak=getc(kontrol\_ptr);

}

koseli\_parantez\_flag=1;

lexeme[i]='\0';

key=keywords(lexeme);

if(sayac>20)// BigAdd'de bir kelime max 20 karakterden oluşur.

{

fprintf(ptr\_2,"%d . satirda hata! Bu kelime 20 karakterden uzun..\n",satir\_say);

printf("%d . satirda hata! Bu kelime 20 karakterden uzun..\n",satir\_say);

continue;

}

if(karakter==46){

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

flag=1;

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

satir\_say++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}else if(karakter=='\n')

flag=-1;

if(strcmp("times" ,lexeme)==0)

flag=0;

if(out\_mu(lexeme)==1)// Out deyimini kontrol eden fonksiyon cagrilir.

{

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

while(isspace(karakter))//out kelimesinden sonra gelebilecek bosluklari atlamaliyiz.

{

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

lexeme[i]='\0';

fprintf(ptr\_2,"keyword (%s)\n",lexeme);

printf("keyword (%s)\n",lexeme);

sayilar[0]++;

if(flag==1){

fprintf(ptr\_2,"end of line (.)\n");

printf("end of line (.)\n");

sayilar[3]++;

flag=0;

}else if(flag==-1 && yorum\_flag==0){

fprintf(ptr\_2,"%d. satirda End of line (.) komutu bulunamadi.\n",satir\_say);

printf("%d. satirda End of line (.) komutu bulunamadi.\n", satir\_say);

flag=0;

}

continue;

}

else if(key==1)//eger kelime bir special word ise

{

if(karakter!='}'){

fprintf(ptr\_2,"keyword (%s)\n",lexeme);

printf("keyword (%s)\n",lexeme);

sayilar[0]++;

}

if(flag==1){

fprintf(ptr\_2,"end of line (.)\n");

printf("end of line (.)\n");

sayilar[3]++;

flag=0;

}else if(flag==-1 && yorum\_flag==0){

fprintf(ptr\_2,"%d. satirda end of line (.) komutu bulunamadi.\n", satir\_say);

printf("%d. satirda end of line (.) komutu bulunamadi.\n", satir\_say);

flag=0;

}

continue;

}

else

{

if (lexeme[0]!='\n'){

fprintf(ptr\_2,"identifier (%s)\n",lexeme);

printf("identifier (%s)\n",lexeme);

sayilar[1]++;

}

if(flag==1){

fprintf(ptr\_2,"end of line (.)\n");

printf("end of line (.)\n");

sayilar[3]++;

flag=0;

}else if(flag==-1 && yorum\_flag==0){

fprintf(ptr\_2,"%d. satirda end of line (.) komutu bulunamadi.\n", satir\_say);

printf("%d. satirda end of line (.) komutu bulunamadi.\n", satir\_say);

flag=0;

}

continue;

}

}

if(karakter==34)// Karakterin " ile baslamasi yani string olma olasiligi icin gerekli

{

i=0;

lexeme[i++]=karakter;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

while(karakter!=34 && karakter!=EOF && karakter!=46)

{

lexeme[i++]=karakter;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

if(karakter==46)

{

fprintf(ptr\_2,"end of line (.)\n");

printf("end of line (.)\n");

sayilar[3]++;

karakter=getc(ptr+1);

bayrak=getc(kontrol\_ptr+1);

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

if(karakter=='\n')

satir\_say++;

}

if(karakter==34)//eger ki birdaha " karakteri gelmisse yani string dogru bir sekilde tamamlanmis ise

{

lexeme[i++]=karakter;

lexeme[i]='\0';

fprintf(ptr\_2,"stringConstant %s\n",lexeme);

printf("stringConstant %s\n",lexeme);

}

else //yada ' karakteri hiç gelmedi yani "ali gibi tamamlanmamis bir string ise....

{

fprintf(ptr\_2,"%d. satirda hata!!! Tamamlanmamis string sabiti.\n",satir\_say);

printf("%d. satirda hata!!! Tamamlanmamis string sabiti.\n",satir\_say);

}

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

continue;

}

if(gecerli\_mi(karakter))//gecersiz bir karakterle baslıyorsa

{

fprintf(ptr\_2,"%d. satirda gecersiz bir karakter!!\n", satir\_say);

printf("%d. satirda gecersiz bir karakter!!\n", satir\_say);

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

continue;

}

if(isspace(karakter)&& karakter!=46) // Karakter bosluksa

{

if(karakter=='\n')

satir\_say++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

while(isspace(karakter)&& karakter!=46)

{

if(karakter=='\n')

satir\_say++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

if(karakter==46){

fprintf(ptr\_2,"end of line (.)\n");

printf("end of line (.)\n");

sayilar[3]++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

if(karakter=='\n')

satir\_say++;

}

continue;

}

if(karakter==',')// karakter , sembolü ise

{

fprintf(ptr\_2,"comma\n");

printf("comma\n");

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

continue;

}

if(karakter=='\n')

satir\_say++;

if(karakter=='-')//karakter - sembolü ise

{

fprintf(ptr\_2,"eksi\_op (%c)\n",karakter);

printf("eksi\_op (%c)\n",karakter);

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

if(!isdigit(karakter)){

fprintf(ptr\_2,"%d. satirda hata. '-' operatorunden sonra sayi disinda bir sey gelemez.\n", satir\_say);

printf("%d. satirda hata. '-' operatorunden sonra sayi disinda bir sey gelemez.\n", satir\_say);

}

continue;

}

if(karakter=='+')//karakter + sembolü ise

{

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

if(!isdigit(karakter)){

fprintf(ptr\_2,"%d. satirda hata. '+' operatoru kullanilamaz.\n", satir\_say);

printf("%d. satirda hata. '+' operatoru kullanilamaz.\n", satir\_say);

}

continue;

}

if(karakter=='[')//karakter [ sembolü ise

{

fprintf(ptr\_2,"open\_square\_bracket (%c)\n",karakter);

printf("open\_square\_bracket (%c)\n",karakter);

sayilar[2]++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

if(karakter=='\n')

satir\_say++;

//Loop içinde bir problem olursa satir sayisi direk yazdirilacak.

continue;

}

if(karakter!='.'){

if(karakter==']' && koseli\_parantez\_flag==1){

fprintf(ptr\_2,"%d. satirda end of line (.) komutu bulunamadi\n", satir\_say);

printf("%d. satirda end of line (.) komutu bulunamadi\n", satir\_say);

koseli\_parantez\_flag=0;

}else if(isspace(karakter)){

while(isspace(karakter)){

if(karakter=='\n')

satir\_say++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

}

}

if(karakter==']')// karakter ] sembolü ise

{

fprintf(ptr\_2,"close\_square\_bracket (%c)\n",karakter);

printf("close\_square\_bracket (%c)\n",karakter);

sayilar[2]++;

long deger = ftell(ptr);

fseek(kontrol\_ptr, deger, SEEK\_SET);

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

if(karakter=='\n'){

satir\_say++;

bayrak=getc(kontrol\_ptr);

karakter=getc(ptr);

}

continue;

}

}

fclose(ptr\_2); //yazma yaptigimiz dosya kapanir

fclose(kontrol\_ptr);

fclose(ptr);

}

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

printf("Toplam Bulunan Değerler:\n");

printf("Keyword Sayisi: %d\n", sayilar[0]);

printf("Identifier Sayisi: %d\n", sayilar[1]);

printf("Parantez Sayisi: %d\n", sayilar[2]);

printf("End of Line Karakteri Sayisi: %d\n", sayilar[3]);

return 0;

}

int lexeme\_kontrol(char dizi[],int k){

int i=0,bulundu=0;

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

{

if(dizi[i]=='.')

{

bulundu=1;

break;

}

}

if(bulundu==1)

return 1;

else return 0;

}

int out\_mu(char dizi[])

{

if(strcmp("out",dizi)==0)

return 1;

else return 0;

}

int keywords(char dizi[]) // Keywordlari tanitmak için bir fonksiyon

{

if((strcmp("out",dizi)==0 || strcmp("add",dizi)==0 || strcmp("move",dizi)==0 || strcmp("to",dizi)==0 || strcmp("int",dizi)==0 || strcmp("sub",dizi)==0 || strcmp("loop",dizi)==0 || strcmp("from",dizi)==0 || strcmp("times",dizi)==0 || strcmp("newline",dizi)==0))

return 1;

else return 0;

}

int gecerli\_mi(char hrf) // Bir karakterin geçerli olup olmadigina bakar.

{

if(hrf==35 ||hrf==36 || hrf==37 ||hrf==38 ||hrf==94 ||hrf==62 ||hrf==60 ||hrf==33|| hrf==64|| hrf==40 || hrf==41 || hrf==58|| hrf==59|| hrf==64 || hrf==61|| hrf==63|| hrf==39|| hrf==34|| hrf==42 || hrf==47)

return 1;

else

return 0;}

**3. PROGRAMCI KATALOĞU**

**Projedeki Kişi Sayısı:** 2 kişi

**Analiz:** 1 gün

**Tasarım:** 2 gün

**Gerçekleştirme:** 3 gün

**Test Ve Raporlama:** 1 saat

**4. TEST VERİLERİ VE EKRAN GÖRÜNTÜLERİ:**

**Test Verileri:**



**Ekran Görüntüsü:**

****