# T.C.

# SAKARYA ÜNİVERSİTESİ

## BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

## BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

## Lexical Analysis Kavramı


## Grup Elemanları:

**B161210074 – Ahmet Yasir AKBAL**


**SAKARYA**

**Mart, 2018**

Programlama Dillerinin Prensipleri Dersi

# Lexical Analysis Kavramı

Ahmet Yasir AKBAL

[a] *b161210074*

**Özet**

Ödev çözülmesi istenen problem .java dosya uzantılı bir sınıfın elemanlarını, fonksiyonlarını, fonksiyonlarının parametrelerini dönüş tipi ve ad olarak listelememiz idi. Bunu yapabilmek için öncelikle JAVA programlama dilinin syntax yapısını, temel veri tiplerini kodumun daha düzenli, modern bir şekilde yazıbilmek için sınıf yapısını ve son olarak dosya okuma yapabilmek için JAVA programa dilindeki dosya okuma yapısını incelemem gerekiyordu. Daha önce C# dilini ders olarak aldığım için JAVA'nın syntax yapısı yabancı gelmedi. Fakat yine de tam anlamıyla hakim olmak için Sadi Evren ŞEKER'in Youtube kanalındaki referanslar kısmında belirtilen 3 videoyu([1],[2],[3]) izledim. Bu videoları izledikten sonra JAVAda dosya okuma yapısını incelemem gerekiyordu. Bunun için video kaynak yerine internetteki yazılı kaynakları([41],[5],[6]) tercih ettim.Ödevde dosya okumak için BufferedReader sınıfını kullandım. Ödeve ilk başlarken .java uzantılı dosyayı satır satır okuyup her satırı bir String dizisinin içinde tuttum fakat bu yaklaşım ödevin devamında sıkıntı çıkardı. Sonrasında dosyayı satır satır okumak değil de kelime kelime okumam gerektiğini anladım. Fakat bu kelimeler Türkçede kullandığımız kelimenin anlamını tam olarak karşılamıyordu. Ödevimde '{' ifadesini de '}' ifadesini de ';' ifadesini de bir kelime olarak kullandım. Yani programlama dillerinde kullanılan ve benim ödevde yapmam istenilen eylemlerle alakalı ayraçlar da benim için bir kelimeydi. Bu ayraçları referans alarak sonrasında döngüler ve kontrol ifadeleri ile benden istenilen eylemleri gerçekleştirecektim. Dosya okuma işlemi yaparken yorum satırlarını/bloklarını ve tırnak içlerindeki ifadeleri es geçtim. Döngü ile okunan satırlar üstünde işlem yaparken ise fonksiyonların, metotların köşeli parantezleri içerisindeki kodları atladım. Çünkü üye elemanları, fonksiyonları ve bu fonksiyonların parametlerini bulmak için bunlara ihtiyacım yoktu.



Anahtar Kelimeler: oku, böl, ata, analiz

## 1. GELİŞTİRİLEN YAZILIM

Programımda yapılan ilk iş Program.java dosyasındaki benim belirlediğim kelimeleri, karakterleri okumak oldu. Okumasını istediğim özel karakterler ',' ,';' ,'(', ')', '{', '}', '= ' idi. Bunlar özel karakterlerdi çünkü bunları özel kılan durumlar vardı. Küme parantezlerini özel olarak string dizisine attım çünkü bu parantezler bir sınıfın başlangıcını, bitişini ve bir fonksiyonun/metodun bitişini belirliyordu. Fonksiyon/metot bloklarını açmak için tanımlanan küme parantezleri görüldüğünde bu metodu/fonksiyonu kapatan küme parantezine kadar olan ifadeler üzerinde bir kontrol ifadesi yapılmayacak ve es geçilecekti. Bu ilk bakışta çözülecek bir problem gibi dursa da biraz detaylı düşününce ortaya bir sorun çıkıyordu. Bu sorun '{' ifadesini gördükten sonra gelen ilk '}' ifadesinin her zaman metodun/fonksiyonun bloğunu kapatmak için olmama olasılığıydı. Örneğin bir fonksiyonun içinde bir while döngüsü açılıp kapatıldığında görülen ilk '}' ifadesi while bloğunu kapatmak için olacaktı. Bu durumda fonksiyonun içindeki diğer '{' sembollerini de dikkate almak gerekliydi.(bknz: 2.ÇIKTILAR)

Programımı nesneye dayalı bir şekilde tasarladım. Programda kullandığım sınıflardan DegiskenPAR sınıfı sınıfta bulunan üye veya static elemanların ve fonksiyon parametlerinin adını, geri dönüş tipini tutmak için tasarladığım bir sınıftı. Üye elemanlar private idi. Bunlara erişim için public metotlar tanımladım. Fonksiyonlar sınıfı ise okunacak olan Program.java adlı dosyadaki sınıftaki fonksiyon ve metotların adlarını, geri dönüş tiplerini, parametre sayılarını tutuyordu. Parametlerin adlarını ve geri dönüş tiplerini tutmak içinse bu sınıf içersinde DegiskenPAR sınıfından bir dizi oluşturdum. Daha genel bir sinif olan "Sinif" sinifi ise içerisinde siniftaki fonksiyon sayısını, eleman sayısını, sınıf ismini, DegiskenPAR ve Fonksiyonlar sınıflarından türetilmiş dizileri barındırıyordu. Ayrıca bu sınıf bizden istenen çıktıları ekrana yazan fonksiyonu da barındırıyordu. DosyaOku sınıfı ise dosya okuma işlemlerini, okunan string ve özel ifadeleri string dizisine atamayı, tırnak içlerindeki ve yorum satırı içlerindeki ifadeleri belirleme gibi bazı kontrol işlemlerini gerçekleştiriyordu. Anasınıfım olan B161210074_Odev1 sinifi ise sözlük analizi (lexical analysis) denen işlemi gerçekleştiriyordu.

### 1.1 Ödev ne için verilmiş olabilir?

Ödevin verilme amacı programa dillerindeki genel sözdizimini kavramak, lexical analysis kavramını anlamaktır.

---

* Ödev Sorumlusu. Ahmet Yasir Akbal, b161210074
*Mail Adresi:* ahmet.akbal@ogr.sakarya.edu.tr

## 2. ÇIKTILAR

```
acma = 1; //zaten acma parantezinin 1 oldugunu biliyor
kapama = 0; //amac acma ve kapama parantezlerinin sayı
for(i = i+2;i < kelimeSayisi-4 && acma != kapama;i++)
{
    if(kelimeler[i].equals("{"))
        acma++;
    else if(kelimeler[i].equals("}"))
        kapama++;
}
```

Şekil 1. Fonksiyon /metot bloklarının içine es geçme

Şekil 1deki kod öncelikle zaten halizırda bir fonksiyon gövdesinin içine girildiği için küme acma parantezinin sayısını belirten acma degiskenine 1 değerini, küme kapama parantezinin sayısını belirten kapama degiskenine ise henüz bu ifade okunmadığı için 0 degerini atıyordu. Daha sonra bir döngü içine giriliyordu. Bu döngü acma ve kapama degiskenleri birbirene eşit olmadığı sürece devam ediyordu. Döngü bloğunun içinde yapılan kontrol ifadelerini eğer küme açma parantezi görüldüyse acma degiskenini 1 arttıyor eğer bu durum gerçekleşmiyor ve küme kapama parantezi görüldüyse kapama degiskenini 1 arttırıyordu. Bu döngü ile fonksiyon gövdelerinin içinde kullanılması muhtemel olan döngü,kontrol vb. ifadeler es geçilmeyecek ve fonksiyon gövdesinin string dizisinin hangi indisinde kapatıldığı sorunsuz bir şekilde bulunarak fonksiyon gövdelerinin içi es geçilecekti. Bunun yanı sıra tırnak içindeki ifadeler, yorum satırları/blokları benim kontrol edeceğim bazı özel ifadeler ile aynı olabileceği için bu ifadeleri dosyadan okuma işlemi yaparken es geçtim.

Programımdaki bir diğer çıktı sorunu ise üye ve static elemanlara atama yapıldığında ortaya çıkan sorundu. Yazdığım eğer noktalı virgül gördüysen ondan önceki string değişken adı, ondan 2 önceki string de veri tipidir algoritması yetersiz kalıyordu. Çünkü işin içine eşittir işareti ve eşittir işaretinin sağ tarafına yapılan atamalar giriyordu. Bu durum özellikle bir dizi ataması yapılmışsa daha da büyük sorun teşkil ediyordu. Bu durumdan kurtulmak için Şekil 2'deki kodları kullandım.

```
for(int j=a;j>bas && !kelimeler[j].equals(";");j--)
{
    if(kelimeler[j].equals("="))
    {
        a = j-1;
        break;
    }
}
```

Şekil 2. İlk değer ataması yapılmış üye ve static elemanları analiz etme

Bu kod parçasındaki döngü direkt noktalı virgül işaretini görmediği ve j değişkenini sınıfın başlangıç indisinden büyük olduğu sürece j değişkenini her defasında bir eksiltiyordu. Eğer okunanan string "=" işaretini eşit ise de a değişkenini j'nin bir eksiğine eşitliyordu. Böylece a değişkenin eşittir işaretinin solundaki stringin indisini tutuyordu. Bu string de değişken adıydı, ondan önceki de veri tipi. Sonrasında klasik atama işlemleri yapılıyordu. Tabi eğer = işareti bulunmuyorsa a değişkenin üzerinde hiçbir değişiklik yapılınmıyor ve klasik atama işlemleri yapılıyordu.

## 3. SONUÇ

Programlama dillerinde genel sözdimini kavramış oldum. Lexical analysis kavramını Regex sınıfı ile çok daha kolayca yapabileceğimi öğrendim fakat ödevimi bu şekilde yapmadım. Her ne kadar ödev bize lexical analysis kavramını anlatmak için verilmiş olsa da ödevin bana en büyük artısı algoritma geliştirme üstüne oldu. Çünkü ödevde bir çok işlemi hazır sınıf kullanmadan kendim yaptım.

## Referanslar


[1] https://www.youtube.com/watch?v=H_SKv07-648

[2] https://www.youtube.com/watch?v=7vUlrVLOxZA

[3] https://www.youtube.com/watch?v=KoD-9p_sX7I

[4] https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-101/javada-dosya-islemleri

[5] http://javayaz.com/?page_id=98

[6] http://blog.thecodeprogram.com/java-ile-dosya-file-islemleri/