T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

Fonksiyona Ait Yorumları Tespit Etme

B211210031 - Eren KARA

SAKARYA

Nisan, 2023

Programlama Dillerinin Prensipleri Dersi

Fonksiyona Ait Yorumları Tespit Etme

Eren Kara

Numara: B211210031 Grup: 1.C

Özet

Java dilinde yazılmış ve derlenebilen bir kod içerisinde, fonksiyonlara ait yorumların Java dilinde yazılmış bir programla bulunması istendi. Bende bu problemi şu şekilde çözmeye çalıştım: Regex ifadeleri kullanarak sınıf tanımını, fonksiyon tanımını, string ve yorum tanımlarını standardize ettim. Regex ifadelerini 6 başlık olmak üzere genel bir ifadede topladım. Öncelik ilk olarak stringlerde, ifade eğerki string değilse öncelik yorumlarda, eğerki ifade yorum değilse öncelik sınıfta o da değilse fonksiyonlarda olmak üzere genel bir regex ifadesi yazdım. Böylelikle stringlerin ve yorumların içerisindeki fonksiyonları ve sınıfları algılamadım. Sonrasında java dilinde bu Regex ifadelerini kullanarak sınıf, fonksiyon ve yorum tespiti yaptım. Bulduğum yorumları liste yapısında tuttum. En son olarak da bizden istenilen şekilde bu yorumları ekrana veya dosyaya çıktı olarak verdim.

Anahtar Kelimeler:Regex,Java,Yorum bulma,Dosya işlemleri.

1. GELİŞTİRİLEN YAZILIM

1.1. Regex ifadeleri

Ödev için regex ifadelerini kaynakça[1]'de belirttiğim siteden öğrendim ondan sonra kaynakça[2]'de belirttiğim siteden oldukça fazla sayıda deneme yaparak sınıfları, fonksiyonları ve yorumları bulacak regex ifadelerini ayrı ayrı yazmaya çalıştım. Regex ifadelerini oluştururken her durumu düşünerek ilerlediğimden aklıma fonksiyon veya sınıf tanımlamalarının string ifadelerinin içinde olma durumu veya yorum satırlarının içinde olma durumu geldi. Bunu hesaba katmadığımdan bir çözüm düşünmem gerekiyordu. Ben de regex ifadelerini biraz daha kurcalayarak regex ifadelerinde kullanılan veya (|) operatörünün soldan sağa doğru sırayla algıladığını fark ettim. Sonra bu birbirinden ayrı yazdığım regex ifadelerini genel bir ifade olarak yazmayı düşündüm. Birinci sırada string regexi, ikinci sırada doc, üçüncü sırada çoklu yorum, dördüncü sırada tekli yorum beşinci sırada sınıf ve son olarakta fonksiyon regexi olacak şekilde genel bir regex yazdım. Böylelikle yorumların ve stringlerin içerisindeki sınıf veya fonksiyon tanımlaması gibi görünen ifadeleri algılamayacaktım. Bu ifadelerin yazımları ve denemeleri bittikten sonra bunu java içerisinde kullanmayı öğrenmek vardı sırada. Regexlerin java içerisinde kullanımını da kaynakça[3]'te belirttiğim Java'nın orijinal dokümantasyonundan ve hocamızın kaynakça[8]'de attığı videodan öğrendim.

1.2.Dosyadan okuma ve komut satırı

Dosyadan okuma işlemlerini kaynakça[9]'daki videodan hallettim. Komut satırı parametresi alma işini ise kaynakça[11]'den.

1.3. Algoritma ve kodlama

Kodlamaya ilk olarak dosyadan okumakla başladım. Bunun için DosyadanOkuma sınıfı diye bir sınıf oluşturup constructor'ını private yaptım ki nesne oluştulamasın çünkü içerisinde sadece static olarak oku() fonksiyonunu yazdım. Bu fonksiyon aldığı string ifadeyi dosya olarak algılıyor ve bu dosya mevcutsa içeriğini FileReader sınıfını kullanarak okuyor geriye ise bir string ifade olarak döndürüyor.

Dosyayı okuma işlemini hallettikten sonra dosya içerisindeki yorumları algılamak için algoritma kurmak gerekiyordu. Aklımdaki algoritmik yapı 3 sınıf oluşturup bu sınıfların görevi birbirine devrederek yorumları bulması şeklindeydi. Böylece fonksiyonların ve sınıfların üstündeki – Bizden istenmedi ama ben buna da elverişli bir şekilde yapı kurmak istedim – yorumları kolaylıkla tespit edip aktarabilecektim. Bu 3 sınıfın ortak alanları ve methodları olduğundan bu sınıflara şablon oluştursun diye TespitEtme sınıfını abstract olarak oluşturdum. İçerisinde field olarak doc yorumları, çoklu yorumları ve tekli yorumları içinde tutabilecek 3 adet ArrayList yapısı

bulunuyordu. Ayrıca dosyanın içeriğini tutabilecek String bir field ve hangi dosyalara yazılacağını belirten 3 adet File tipi field da bulunuyordu. Fonksiyon olarak abstract tespitEt() methodu ile Sınıfın ve fonksiyonun küme parantezlerini eşleştirmeye yarayan yani scope bölgesini söyleyen KumeParanteziEslestirme() methodu bulunuyordu. KumeParanteziEslestirme() metodu { gördüğünde içerisindeki int değişkenini 1 arttırıyor ve } gördüğünde bir azaltacak şekildeydi.

3 sınıfın ilki olan DosyaIciTespitEtme sınıfı TespitEtme sınıfından kalıtım alıyor. DosyaIciTespitEtme sınıfının görevi sınıfın üstünde bulunan yorumları ve sınıfı tespit etmekti. Yani dosya içeriğini genel olarak taramak istenildiğinde bu methoda başvurulması gerekiyor. Bunun için içerisinde tespitEt() mehodunu barındırıyor. Bu method geriye boolean değer döndürüyor ve regexGenelFonksiyonsuz (regex genel ifadesinin 6 başlığından fonksiyonun çıkarılmış hali) regexi ile eşleşen yapıları buluyor. Bunu yapabilmek için Pattern ve Matcher sınıflarını kullanıyor. Her yorum eşleşmesinde ArrayList yapılarına aktarıyor bu yorumları. Sınıf eşleşmesinde ise tespit etme görevini SınıfIciTespitEtme sınıfına bırakıyor. Bunu ise içinde bulunan SınıfIciTespitEtme nesnesinin tespitEt() methodunu kullanarak yapıyor. Bu methoda parametre dosya içeriğini olarak sınıfın ismini ve sınıf küme parantezleri arasında kalan bölgenin başlangıcını ve sonunu veriyor. Bir sınıf bulduğunda döngüden çıkıyor. Eğer dosyanın içeriği boş ise bir yarı mesajı döndürüyor. dosyayaYazdir() methodu ilk önce dosyanın içeriği boşaltıyor ondan sonra sınıf bulunduysa SınıfIciTespitEtme sınıfının dosyayaYazdir() methodunu çağırıyor. Eğer sınıf bulunamadıysa uyarı mesajı veriyor. degistir() methodu içerisine bir dosya içeriği alıyor parametre olarak eğerki dosya içeriğini değiştirerek tekrar kullanıma yarıyor. toString() methodu tespitEt()'ten olumlu geri dönüş değeri alırsa bulduğu sınıf üstü yorumları ve sınıf ismini ekrana yazıyor devamını yapması için SınıfIciTespitEtme sınıfının toString() methoduna bırakıyor görevi.

2. Sınıf olarak SınıfIciTespitEtme sınıfı var. TespitEtme sınıfından kalıtım alıyor. Field olarak String isim, integer start ve integer end değişkenlerini barındırıyor. Ayrıca diğer sınıf olan FonksiyonIcıTespitEtme tipinde veriler tutan bir liste tutuyor. Yani fonksiyonları tutuyor diyebiliriz. Bu sınıfın amacı sınıf küme parantezleri arasında kalan yorumları ve fonksiyonları bulmak. Fonksiyon bulunca fonksiyon içi yorum bulma görevini FonksiyonIcıTespitEtme sınıfına devrediyor ve yeni yorumlar için ve fonksiyonlar için taramaya devam ediyor ayrıca fonksiyon üstü doc yorumları ise altındaki fonksiyona ekliyor. Bunu FonksiyonIcıTespitEtme sınıfının listeleriAktar() methoduyla yapıyor. dosyayaYazdir() methodu ise dosyaya yazdırması için FonksiyonIcıTespitEtme sınıfının dosyayaYazdir() methodunu çağırıyor. toString() methodu ise döngü şeklinde fonksiyonun ismini yazdırıyor ve kalanı yapması için için FonksiyonIcıTespitEtme toString() methodunu çağırıyor.

3. Sınıf olarak FonksiyonIcıTespitEtme sınıfı var. TespitEtme sınıfından kalıtım alıyor. Field olarak String isim, integer start ve integer end değişkenlerini barındırıyor. dosyayaYazdir() methodu FileWriter sınıfı kullanarak 3 adet nesne oluşturuyor. Bunlar javadoc.txt, coklusatir.txt, teksatir.txt dosyalarına yazdırıyor.Sırayla her bir liste içerisindeki yorumlar ilgili dosyaya yazdırılıyor. tespitEt() methodu fonksiyon parantezleri içerisinde kalan yorumları buluyor ve bunu ilgili listeye ekliyor. toString() methodu ekrana ilgili listedeki yorum sayısını ekrana yazdırıyor. listeleriAktar() methodu SınıfIciTespitEtme sınıfının bulduğu fonksiyon üstü doc yorumları FonksiyonIcıTespitEtme sınıfın listesine aktarıyor.

Son olarak başta bahsetmeyi unuttuğum private constructor'a sahip olan RegexIfadeleri sınıfı var. Sınfıın her bir üyesi static. Belirli regex ifadelerini field olarak barındırıyor. 2 adet methodu bulunuyor bunlar ise dışardan gelen string isim (sınıfın ismi olmalı) değerine göre sınıfın yapıcı constructor'ın da düzgün bir şekilde algılanmasını sağlıyor. Çünkü constructor örneğin erişim belirteçsiz kullanılırsa algılamak çok zor. O yüzden sınıfın ismine ihtiyaç var. Bu yüzden bu 2 fonksiyon geriye sadece string bir ifade döndürmesine rağmen field olamıyor çünkü dışardan gelen parametreye ihtiyacı var.

1.4. Ödevin verilme amacı

Ödevin amacı bir programlama dilinin içerdiği tanımlamaların, anahtar kelimelerin, yorum satırlarının, string ifadelerinin diğer şeylerden nasıl ayrıldığını fark edip bir derleyici edasıyla bunları seçebilmemiz. Bu tanımlamaların diğer yapılardan tanımlanırken farklılıkları ne, ne olunca fonksiyon, ne olunca sınıf, ne olunca yorum satırı oluyor, ne gibi durumlara müsaade ediyor ne gibilerine müsaade etmiyor, bu ve bunun gibi sorulara cevap aramak. Örneğin ben şunu fark ettim: Kitap [] dizi =new Kitap[]{ new Kitap(),new Kitap() }; gibi bir ifadeyi fonksiyon tanılamasından ayıran yapı Kitaptan sonra () ifadesi yerine [] ifadesinin geliyor oluşu. Eğerki () ifade gelirse { } koymamıza izin vermiyor. Ya da c/**/l/**/a/**/s/**/s Kitap {} gibi bir tanımlamaya da müsaade etmiyor kaynakça[12]'den öğrendiğim kadarıyla yorum satırlarını çıkarıp atmakla kalmıyor, onlara whitespace muamelesi uyguluyormuş. Bu yüzden kabul etmiyormuş bu tanımı. Ya da tanımlama arasına istediğimiz kadar whitespace karakteri koyabiliyor oluşumuz class Kitap {} ya da string veya yorum satırının öncelikli olarak algılandığını fark etmek (ya da ben böyle olduğunu zannediyorum).

Ödevinin amacı programlama dillerinin içerdiği yapıların derleyici tarafından algılanma mantığını anlatmaya çalışmak yani programlama dillerinin prensiplerinin farkına varmamızı sağlamak.

2. ÇIKTILAR

1.Eğerki komut satırı parametresi girilmez ise
"Main.main() : Konsol parametresi girilmedi" ,
"DosyaIciTespitEtme.tespitEt() : Dosya bulunamadığından tespit etme islemi gerçekleşmedi",
"DosyaIciTespitEtme.toString() : Sınıf ici tespit gerceklesemedi cunku dosya bulunamadı",
"DosyaIciTespitEtme.dosyayaYazdir() : Sınıf bulunamadığından dosyaya yazılamadı." mesajları çıkıyor.

2. Eğer ki dosya bulunamadıysa
"DosyadanOkuma.oku() : Dosya bulunamadı",
"DosyaIciTespitEtme.tespitEt() : Dosya bulunamadığından tespit etme islemi gerçekleşmedi",
"DosyaIciTespitEtme.toString() : Sınıf ici tespit gerceklesemedi cunku dosya bulunamadı",
"DosyaIciTespitEtme.dosyayaYazdir() : Sınıf bulunamadığından dosyaya yazılamadı." mesajları çıkıyor.

3. Eğerki dosya içeriği boş ise
"Sınıf tespit edilemedi"
"DosyaIciTespitEtme.dosyayaYazdir() : Sınıf bulunamadığından dosyaya yazılamadı" mesajları çıkıyor.

4. Ben diğer durumlarda istenilen şekilde çıktı verdiğini düşünüyorum.

3. SONUÇ

Eğerki bir kod dosyasından istediğim verileri çekmek istersem artık nasıl yapacağımı biliyorum. Yorum bloklarını ise bu program sayesinde çekebilirim.

Referanslar

[1] regexlearn.com

[2] regex101.com

[3] https://docs.oracle.com

[4] https://docs.oracle.com/javase/7/docs/api/index.html?java/util/regex/Pattern.html

[5] https://docs.oracle.com/javase/7/docs/api/java/util/regex/Matcher.html

[6] https://docs.oracle.com/javase/8/docs/api/java/io/OutputStreamWriter.html#write-char:A-int-int-

[7] https://docs.oracle.com/javase/7/docs/api/java/io/File.html#File(java.lang.String)

[8] https://www.youtube.com/watch?v=9N6Oj0QJ4M8

[9] https://www.youtube.com/watch?v=qzTug27N54w

[10] https://www.w3schools.com/

[11] https://stackoverflow.com/questions/1055318/using-command-line-argument-for-passing-files-to-a-program

[12] https://www.quora.com/How-exactly-does-the-compiler-deal-with-comments-in-code-specifically