T.C.

SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

Github Repository Java Kod Analiz

B201210071 - Ali Kutay KILINÇ

SAKARYA

Nisan, 2024

Programlama Dillerinin Prensipleri Dersi

Github Repository Java Kod Analiz

Ali Kutay KILINÇ

[a] *B201210071*

Özet

Proje Java dilinde bir konsol uygulaması. Program açıldığında kullanıcıdan Github Repository Url ister. Daha sonra bu git deposunu klonlar. İçinden sadece *.java uzantılı sınıf dosyalarını seçer ve analizde bulunur. Her bir dosya satır satır incelenir, tipine göre sayım işlemine konur. Bulunan sayım sonuçları, bir formül kullanılarak okunan kodun yorum satırlarının yeterliliği hakkında bilgi sağlar. Daha sonra okunan tüm dosyaların analiz edilen değerleri ekrana yazdırılır.

İncelenen dosya kod ve yorumlardan oluşur. Birden fazla kod ve yorum çeşidi var. Bir satır aynı anda birden fazla tipte olabilir. Bazıları ise aynı anda asla olamaz. Programlama dillerinin bu ayrımı yapan kuralları vardır. Bu kurallara uyarak kaynak kod metni yorumlanır. Java için bu kuralları inceledim ve gerekli şartların sağlandığı, aykırı durumların sayıma alınmadan engellendiği bir analiz sınıfı yazdım.

Anahtar Kelimeler: Yorum satırları, Derleyici, Yorumlayıcı, Sözdizimi ve semantik,

1. GELİŞTİRİLEN YAZILIM

Klonlama işlemi için GitRep adında bir sınıf oluşturdum. Kullanıcıdan Github Repository URL'sini istedim. ProcessBuilder kullanarak Git komutlarına eriştim, klonladım. "File.mkdirs()" kullanarak bu klonlama işleminin yapılacağı "repository" isimli bir klasör oluşturdum. Eğer daha önce klonlama işlemi yapılmışsa bu klasör önceden üretilmiş demektir ve klonlama işleminde hata oluşur. Bu yüzden her klonlama işleminden önce silme yapılmalı. Klasör boşken silmesi kolay olsa da doluyken tek seferde silinmedi. İçindeki her bir klasörü gezip tamamen silen recursive bir fonksiyon yazdım. Klonlama işlemi tamamlandı.

"repository" klasörünün konumunu GitRep sınıfından alan ve içinden "*.java" uzantılı sınıfları seçen Analiz adında bir sınıf oluşturdum. Analiz edilecek dosyaları seçmek için dosya isimlerini "File.getName()" ile alarak ".java" uzantısı var mı diye sorguladım. Bu koşulu sağlayanları "BufferReader" ile okuyarak içinde "class" olup olmadığına baktım. Olanları yazdığım "satirAnalizi()" fonksiyonuyla analiz edip ekrana yazdım.

"satirAnalizi()" fonksiyonuyla dosya isimlerini alarak satır satır her bir dosyayı inceledim. İstenen değerler için sayaç değişkenleri ürettim.

- Javadoc olarak yorum satır sayısı
- Diğer yorumlar satır sayısı
- Kod satır sayısı (tüm yorum ve boşluk satırları hariç)
- LOC (Line of Code) (Bir dosyadaki her şey dahil satır sayısı)
- Fonksiyon Sayısı (Sınıfın içinde bulunan tüm fonksiyonların toplam sayısı)
- Yorum Sapma Yüzdesi (Yazılması gereken yorum satır sayısı yüzdelik olarak ne kadar sapmış)

Tablo 1. Kod ve açıklama satırlarının aynı satırda olma kuralları tablosu

	Javadoc	Tekli Yorum	Çoklu Yorum	Kod	Fonksiyon	Boşluk
Javadoc	**0**	-	-	-	-	-
Tekli Yorum	-	**0**	-	+	+	-
Çoklu Yorum	-	-	**0**	-	-	-
Kod	-	+	-	**0**	+	-
Fonksiyon	-	+	-	+	**0**	-
Bosluk	-	-	-	-	-	**0**

Aynı satırda hangi durumların olabileceğini hangilerinin olamayacağını inceledikten sonra bu şartlara bağlayarak satır tipi incelemelerini yaptım. Javadoc ve çoklu yorum satırını sayarken başlangıç ve bitiş değerleri sayılmayacağı için satırın konumuna göre sayaç işlemi yaptım. Bu değerleri boolean değişkenleriyle tuttum. Bu ikisin en önemli diğer bir ortak noktası ortadaki sayılacak satırların başlangıcının aynı olmasıydı.

Bir önceki satırın hangi tip olduğuna bakarak doğru tipi arttırmayı başardım. Satır okunduğunda ikisi de tepki verse de doğru olan sayacı arttırdı.

Fonksiyon sayısını satırın fonksiyon başlangıcı olup olmadığına göre inceledim. Bunun için "<access_modifier> <return_type> <method_name>(<parameter_list>) {" yapısını kontrol etmeye çalıştım. Public, private, protected erişim belirleyicileri ve int, float, double gibi dönüş değerlerini aramaya karar verdim. Ama bunları kullan başka kod parçaları da var. Bunları listeledim class, if, while… Daha sonra bunların olmadığı "( )" içeren kod sonu olmayan ";" satırları seçtim. Böyleye fonksiyonları doğru bir şekilde sayabildim. Bu işlemleri "String.contains" ve "String.startsWith" gibi fonksiyonlarıyla yaptım. Benzer işlemleri diğer satır aramaları için de yaptım. Satır başındaki boşlukları es geçtim. Tamamen boş olan satırları "String.isBlank" fonksiyonuyla buldum.

Bulduğum sonuçlarla verilen Yorum Sapma formülünü kullanarak sonuç buldum. Burada elimdeki int değerlerini double'a çevirerek işleme soktum. Daha sonra virgülden sonra iki basamağı olacaka şekilde ekrana yazdırmak için "Math.round" fonksiyonuyla işlem yaparak örnek çıktıya benzer bir sonuç aldım. Tüm bilgileri sınıf ismiyle beraber ekrana yazdırdım

Ödev derleyici ve yorumlayıcıların işleme mantığını anlamamı sağladı. Arka planda nasıl incelendiği ve grupladığı hakkında fikrim oluştu. Yazdığım kod kurallara uygun olarak yazılmış bir kodu incelemek için yazılmış olsa da dikkat edilmesi gereken birçok şey vardı

2. ÇIKTILAR

Verilen kod kurallara uygun olduğu sürece sayım işlemleri doğru bir şekilde yapılmakta:

Sınıf: Atm.java Javadoc Satır Sayısı: 10 Yorum Satır Sayısı: 1 Kod Satır Sayısı: 11 LOC: 28 Fonksiyon Sayısı: 2 Yorum Sapma Yüzdesi: % 166.67	Sınıf: Hesap.java Javadoc Satır Sayısı: 3 Yorum Satır Sayısı: 4 Kod Satır Sayısı: 35 LOC: 53 Fonksiyon Sayısı: 6 Yorum Sapma Yüzdesi: % -46.67	Sınıf: Kart.java Javadoc Satır Sayısı: 5 Yorum Satır Sayısı: 1 Kod Satır Sayısı: 17 LOC: 33 Fonksiyon Sayısı: 3 Yorum Sapma Yüzdesi: % -5.88
Sınıf: MasterKart.java Javadoc Satır Sayısı: 0 Yorum Satır Sayısı: 0 Kod Satır Sayısı: 17 LOC: 22 Fonksiyon Sayısı: 3 Yorum Sapma Yüzdesi: % 100.00	Sınıf: Program.java Javadoc Satır Sayısı: 4 Yorum Satır Sayısı: 6 Kod Satır Sayısı: 18 LOC: 33 Fonksiyon Sayısı: 1 Yorum Sapma Yüzdesi: % 48.15	

SONUÇ

Yazılan bir kodun hem bilgisayar hem de insanların anlaması, istenilen şeyin doğru bir şekilde bilgisayara aktarılabilmesi için programlama dilleri üretildi. Bu dillerin bazıları makine bazıları insan diline daha yakın olsa da hepsinin ortak yanı belli kurallar bütününe uymak zorunda oluşu. Bu kuralları üretip bilgisayarların kullanımının kolaylaştırılması insanlığın daha hızlı gelişmesine yardımcı olur.