T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

G221210038 - Ayşe Verda Gülcemal

Programlama Dillerinin Prensipleri Dersi

GİTHUB JAVA DOSYALARININ ANALİZİ

AYŞE VERDA GÜLCEMAL

G221210038 2B

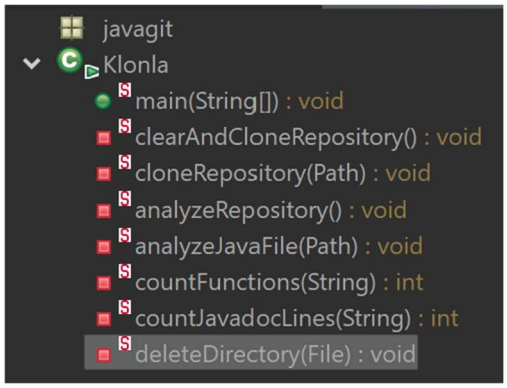
Bu Java projesi, kullanıcıdan bir GitHub repository'nin linkini alarak işleyişine başlar. Girilen linki kullanarak repository'yi klonlar ve ardından klonlanan repository'nin içerisindeki .java uzantılı dosyaları ayırır. Daha sonra, bu dosyalar arasında sınıf olanları belirler ve analiz etmeye başlar.

Analiz süreci, her bir .java dosyasını tek tek dolaşarak gerçekleşir. Dosyaların içeriğini inceleyerek Javadoc satırlarının sayısını, yorum satırlarının sayısını, kod satırlarının sayısını, toplam satır sayısını ve fonksiyon sayısını hesaplar. Bu satır sayılarını da kullanarak bir yorum sapma yüzdesi hesaplar.

Yorum sapma yüzdesi, yazılım geliştirme sürecinde kodun belgelendirilme düzeyini ve kod kalitesini değerlendirmek için kullanılan bir metriktir. Bu metrik, proje içindeki yorum satırlarının ve Javadoc belgelendirmesinin oranını ölçer ve kodun ne kadar iyi belgelendiğini gösterir.

Yani bu proje ile githubdaki herhangi bir projenin belgelendirilme düzeyini ve kod kalitesini saniyeler içinde değerlendirebiliyoruz.

KULLANDIĞIM FONKSİYONLAR :



clearAndCloneRepository()
Bu fonksiyon, klonlanacak klasörü belirler ve klasörün içi doluysa çalışmadan önce içeriğini temizler.
GitHub deposunu klonlamak için cloneRepository() fonksiyonunu çağırır.

cloneRepository(Path directory)
Bu fonksiyon, kullanıcıdan alınan GitHub repository URL'sini kullanarak klasör içinde bu repository'yi klonlar.
ProcessBuilder ve Process kullanarak git clone komutunu çalıştırır.

analyzeRepository()
Klonlanan repository'nin içeriğini analiz etmek için analyzeJavaFile() fonksiyonunu çağırır.
.java uzantılı dosyaları bulur ve her biri için analiz işlemlerini başlatır.

analyzeJavaFile(Path filePath)
Her bir .java dosyasının içeriğini okur ve analiz işlemlerini gerçekleştirir.
Dosyanın içindeki kod, yorum ve LOC satırlarını sayar, fonksiyon sayısı ile yorum sapma yüzdesini hesaplar.

countFunctions(String content)
Dosyadaki fonksiyon sayısını sayar.
Kod içinde public, private veya protected anahtar kelimelerle başlayan satırları fonksiyon olarak kabul eder.

countJavadocLines(String content)
Dosyadaki Javadoc satırlarının sayısını sayar.
/** ile başlayan ve */ ile biten blokları Javadoc olarak kabul eder.

deleteDirectory(File directory)
Verilen dizini ve içeriğini silen yardımcı bir fonksiyondur.
Klonlanan repository'nin içeriğini temizlemek için kullanılır.

ÇIKTI:
Programı ilk çalıştırdığımızda karşımıza linki girin yazısı çıkıyor. Örnek bir linki giriyoruz. Başta dosyanın adı olmak üzere analiz verilerimiz çıkıyor. En altta da sınıfın belgelendirilme düzeyini değerlendirebileceğimiz yorum sapma yüzdesini hesaplıyoruz. Tüm .java uzantılı sınıfları bu şekilde analiz ediyor ve buna göre çıktı veriyor.

```
GITHUB REPOSITORY LİNKİNİ GİRİNİZ:
https://github.com/mfadak/Odev1Ornek
--------------------------------
Dosya adı: Atm.java
Javadoc satırı sayısı: 10 lines
Yorum satırı sayısı: 1 lines
Kod satırı sayısı: 11 lines
Toplam satır sayısı(LOC): 28 lines
Fonksiyon sayısı: 2
Yorum Sapma Yüzdesi: 166,67%
```

Bu Java projesi, GitHub repository'lerinin analiz edilmesi ve kod kalitesinin değerlendirilmesi için kullanışlı bir araç sunmaktadır. Kullanıcı, bir GitHub repository'nin URL'sini girerek bu projeyi kullanabilir ve klonlanan repository'nin içeriğini analiz edebilir. Proje, her Java dosyasını detaylı bir şekilde tarayarak kod satırları, yorumlar ve Javadoc belgelendirmesi gibi önemli metrikleri hesaplar. Böylece, kullanıcılar klonlanmış bir projenin kod kalitesini, belgelendirme düzeyini ve yapısını anlayabilirler. Bu bilgiler, yazılım geliştirme sürecinde önemli kararlar almak için kullanılabilir. Örneğin, yetersiz belgelendirme ya da karmaşık kod yapıları tespit edilerek geliştirme sürecinde iyileştirmeler yapılabilir. Ayrıca, proje, otomatik klonlama ve analiz işlemleri sayesinde zaman kazandırır ve yazılım projelerinin daha etkili bir şekilde yönetilmesine katkı sağlar. Bu nedenle, bu proje yazılım geliştirme topluluğuna değerli bir araç sunarak kod kalitesini artırabilir ve geliştirme süreçlerini optimize edebilir.