# **Programlama Dilleri Final Raporu**

**1.Giriş**

Bu projede amaç belirlenen gramer kurallarına uygun olarak yazılmış programlama diline ait kodların gerçek zamanlı olarak analiz edilmesi ve renklendirilmesidir . Projede herhangi bir kütüphane kullanılmadan , tamamıyla kendimize ait bir lexicel analyzer, parser ve grafik arayüz geliştirilmiştir. Bu uygulama sayesinde kullanıcı kodu yazarken hangi öğenin ne türde biryapı olduğunu renkli olarak görebiliyor ve yazdığı kodun gramer kurallarına uygun olup olmadığını kontrol edebiliyor.

**2.Kullanılan Program Dili ve Araçlar**

Bu projede kullanılan programlama dili pythondur . Bunun sebebi python dilinin Basit sözdizimi ve Tkinter kütüphanesi ile kolayca GUI yapılabilmesidir. Projede ek olarak regex (re) modulü kullanılmıştır bu modül sayesinde Kod içindeki kelime ve sembolleri parçalandı . Ayrıca Python’un str.isidentifier() ve isdigit() gibi yerleşik fonksiyonları da lexer içinde kullanılarak daha anlaşılabilir bir yapı oluşturulmuştur.

**3.Dil Grameri ve Token Yapısı**

Bu projede analiz edilmesi hedeflenen programlama dilinde belirli bir gramer belirlenmiştir. Amaç, kullanıcı tarafından yazılan kodu bu gramer kurallarına göre parçalayıp sınıflandırmaktır.

Tanımlanan Token Türleri:

**Token Türü**  **Açıklama Örnekler**

KEYWORD Anahtar kelimeler int, if, else, while, return

IDENTIFIER Değişken adları x, sayi, toplam

NUMBER Sayısal ifadeler 5, 42, 0

OPERATOR İşlem ve karşılaştırma operatörleri =, ==, +, -

DELIMITER Ayraç karakterleri (, ), {, }, ;

**4.Lexical Analyzer(sözlük analizi)**

Lexical Analyzer, kullanıcı tarafından yazılan kodun içeriğini tarayarak onu anlamlı parçalara (tokenlere) ayırır. Her kelime, sembol ve sayı bir tür ile eşleştirilir. Bu işlem, renklendirme ve gramer kontrolü için temel oluşturur.

Lexical Analyzer (kod\_parcala) fonksiyonu. Bu yapı sayesinde kullanıcı tarafından girilen kodlar, token'lara ayrılarak türlerine göre ayrıştırılır ve parser’a hazır hale getirilir.

metin, ekran görüntüsü, yazılım, ekran, görüntüleme içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**Token Türü belirleme:**

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**5.Parser (Sözdizimi Analizi)**

Lexical Analyzer’dan gelen token’lar, bu aşamada belirlenen gramer kurallarına göre sırasal olarak kontrol edilir. Amaç: Kodun sözdizimi açısından doğru yazılıp yazılmadığını belirlemektir. Projede Top-Down Parsing yöntemi kullanılmıştır.

if\_yapisi fonksiyonu. Bu yapı sayesinde hem süslü parantezli hem tek satırlık if koşulları desteklenmektedir. İçerisinde ayrıca statement() çağrılarak blok içi çoklu ifade desteği sağlanmıştır.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

İfade fonksiyonu, koşullu ifadelerde IDENTIFIER == NUMBER veya IDENTIFIER == IDENTIFIER yazımlarının doğruluğunu kontrol eder.

ekran görüntüsü, metin içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

Parser sadece tek bir komutla sınırlı kalmaması için program adında bir fonksiyonla genişletildi:

metin, yazı tipi, ekran görüntüsü, sayı, numara içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**6. GRAFİK ARAYÜZ (GUI) YAPISI**

Kullanıcının kod yazabileceği, yazarken renklendirme göreceği ve butona basarak gramer kontrolü yapabileceği sade bir arayüz tasarlanmıştır.

Arayüz için Python’un dahili Tkinter kütüphanesi kullanılmıştır.

**Arayüz Bileşenleri:**

metin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**Renklendirme Sistemi:**

metin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**Söz Dizimi Kontrolü:**

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

Kullanıcı “Kontrol Et” butonuna bastığında:

Kod tekrar lexer ile parçalanır

parser fonksiyonları çalışır

Ekranda Kod düzgün. veya Kod hatalı. mesajı gösterilir

Arayüz kullanıcı dostu, basit ve anlaşılır olacak şekilde tasarlanmıştır. Hiçbir dış GUI framework’ü kullanılmamış, sadece Tkinter ile yapılandırılmıştır.

**7.Sonuç**

Proje hem teknik açıdan hem kullanıcı deneyimi açısından başarılı bir şekilde tamamlanmıştır.

Kodların sadeliği ve açık yapısı sayesinde hem eğitim amaçlı hem de temel dil işleme uygulamaları için uygun bir örnek oluşturmuştur.

Program yalnızca sabit değişken tanımlarını değil, koşullu ifadeleri de tanıyıp analiz edebilmektedir. Parser fonksiyonu sade bir biçimde genişletilmiş, kullanıcıya daha işlevsel bir deneyim sunulmuştur. Kodların okunabilirliği ve öğrenme kolaylığı yine ön planda tutulmuştur.

Proje kapsamında;

Hiçbir hazır syntax highlighter kütüphanesi kullanılmamıştır

Lexical analyzer, parser ve GUI kodları tamamen manuel olarak yazılmıştır

Tkinter kullanılarak sade ve işlevsel bir arayüz oluşturulmuştur

Python’un dahili modülleri dışında hiçbir dış bağımlılık yoktur