Programlama Dilleri Dersi
Proje Ödevi Raporu

Nihat Efe Bozkan

22360859033

Haziran 2025

1. Proje Özeti

Bu projede, belirli bir gramer yapısına uygun olarak çalışan, gerçek zamanlı bir **sözdizimi vurgulayıcı (syntax highlighter)** geliştirilmiştir. Kullanıcı dostu bir grafik arayüz (GUI) üzerinden metin girişini analiz eden uygulama, en az beş farklı token türünü anlık olarak renklendirir. Proje, hazır bir vurgulama kütüphanesi kullanılmadan, sıfırdan yazılmış bir *lexical analyzer* ve *syntax analyzer* ile hayata geçirilmiştir. Amaç, sözcüksel ve sözdizimsel analiz süreçlerini entegre ederek kullanıcıya hatasız ve görsel bir deneyim sunmaktır.

2. Kullanılan Teknolojiler ve Gramer

2.1 Programlama Dili

Proje, **Python** programlama dili ile geliştirilmiştir.

2.2 Gramer Tanımı

Gramer, **Backus-Naur Form (BNF)** ile tanımlanmış ve *recursive descent* (top-down) parser yöntemiyle çözümlenmiştir.

2.3 Token Türleri

- **Anahtar Kelimeler (Keywords):** `if`, `else`, `print`, `int`, `char`, `elif`
- **Operatörler (Operators):** `+`, `-`, `*`, `/`, `=`, `==`, `!=`, `>`, `<`
- **Sayılar (Numbers):** Sayısal değerler
- **Tanımlayıcılar (Identifiers):** Kullanıcı tanımlı değişken isimleri
- **Semboller (Symbols):** `(`, `)`, :
- **Karakter Sabitleri (Char Literals):** `'a'`, `'x'` gibi tek tırnak içindeki karakterler
- **Yorumlar (Comments):** `#` ile başlayan satır içi açıklamalar

3. Sözcüksel Çözümleyici (Lexical Analyzer)

Dosya: `lexer.py`
Yöntem: Durum diyagramı (state diagram) ve programlama tabanlı uygulama
Girdi metni, karakter karakter taranarak *regular expression* (regex) ile token'lara ayrılır. Anahtar kelimeler, operatörler, semboller, sayılar ve karakter sabitleri doğru şekilde etiketlenir. `int` ve `char` ile tanımlanan değişkenler kendi türleriyle ayrıştırılır. `tokenize` fonksiyonu, `lexer_test.py` dosyasındaki testlerle doğrulanmıştır.

4. Sözdizimsel Çözümleyici (Syntax Analyzer)

Dosya: `parser.py`
Yöntem: Top-down parsing
Parser, token akışını BNF kurallarına göre analiz eder ve değişken tanımlama, atama, `if-else` yapıları ve `print` komutlarını destekler. Doğruluğu, `parser_test.py` dosyasındaki testlerle kontrol edilmiştir.

5. Grafik Arayüz (GUI) Tasarımı

Dosya: `highlighter_gui.py`
Kullanılan Kütüphane: Tkinter
Özellikler:

- Kod girişi için metin alanı
- Gerçek zamanlı renklendirme
- **Renk Kodları:**
  - Anahtar Kelimeler: Mavi
  - Operatörler: Pembe
  - Sayılar: Turuncu
  - Karakter Sabitleri: Açık pembe / Çikolata
  - Semboller: Yeşil sarı
  - Yorumlar: Açık yeşil
  - `else`/`if` uyumsuzluğu: Kırmızı / Altın

6. Gerçek Zamanlı Vurgulama

Metin girişiyle eşzamanlı olarak `tokenize` ve `parser` fonksiyonları tetiklenir. Her tuş basımında `highlight()` fonksiyonu çalışır ve GUI'de doğru token'lar renklendirilir. `else` deyimlerinin `if` ile eşleşip eşleşmediği analiz edilerek hata gösterimi yapılır.

7. Kod Yapısı

Proje aşağıdaki dosyalardan oluşur:

- `lexer.py`: Tokenizer
- `parser.py`: Parser
- `highlighter_gui.py`: GUI ve vurgulama
- `lexer_test.py`: Lexer testleri
- `parser_test.py`: Parser testleri

8. Test ve Optimizasyon

Kodlar, lexer ve parser düzeyinde ayrı testlerle kontrol edilmiştir. Token eşleşmesi, atama ifadeleri, `if-else` yapısı, `print` fonksiyonu ve karakter literal desteği başarıyla doğrulanmıştır. Tüm testler başarılıdır.

9. Yayın ve Tanıtım

- **Video Tanıtımı:** https://github.com/nihatefebozkan/Real-Time-Grammar-Based-Syntax-blob/main/Demo_video
- **Makale:** https://github.com/nihatefebozkan/Real-Time-Grammar-Based-Syntax-Highlig blob/main/Makale

10. Sonuç

Bu proje, gramer kurallarına uygun, GUI destekli ve gerçek zamanlı vurgulama yapan özgün bir sözdizimi analiz aracı sunar. Kullanıcı dostu arayüzü ve sıfırdan yazılmış tokenizer-parser modülleriyle başarıyla tamamlanmıştır.

11. Ekler

- **Kod Dosyaları:** https://github.com/nihatefebozkan/Real-Time-Grammar-Based-Syntax-H tree/main/pdproje2
- **Test Örnekleri:** `lexer_test.py`, `parser_test.py`