Gerçek Zamanlı Söz Dizimi Vurgulayıcı (GUI ile) - Final Raporu

# 1. Giriş

Bu proje, biçimsel bir dil bilgisine dayalı olarak grafiksel kullanıcı arayüzüne (GUI) sahip, gerçek zamanlı çalışan bir söz dizimi vurgulayıcı (syntax highlighter) uygulamasını konu almaktadır. Proje, en az beş farklı türdeki token'ı gerçek zamanlı olarak vurgularken, hem leksiksel hem de söz dizimsel analiz gerçekleştirmektedir. Proje gereklilikleri doğrultusunda, vurgulama için herhangi bir üçüncü parti kütüphane kullanılmamıştır.

# 2. Programlama Dili

Proje, Python programlama dili kullanılarak geliştirilmiştir. GUI geliştirmek için Tkinter kütüphanesinden faydalanılmıştır. Python’un okunabilirliği ve düzenli ifadeler (regex) ile güçlü desteği nedeniyle tercih edilmiştir.

# 3. Dil ve Gramer Seçimi

Analiz için, C/Java benzeri basit ve statik türde bir programlama dili modeli seçilmiştir. Dil; değişken tanımlama, atama, aritmetik ifadeler ve temel kontrol yapılarını desteklemektedir.

Örnek Gramer (CFG):

Program -> StmtList  
StmtList -> Stmt StmtList | ε  
Stmt -> DeclStmt | AssignStmt  
DeclStmt -> Type id ;  
AssignStmt -> id = Expr ;  
Type -> int | float | string  
Expr -> Term Expr'  
Expr' -> + Term Expr' | - Term Expr' | ε  
Term -> Factor Term'  
Term' -> \* Factor Term' | / Factor Term' | ε  
Factor -> id | number | ( Expr )

# 4. Leksiksel Analiz

Yaklaşım: Durum Diyagramı & Program Uygulaması  
  
Leksik analizör, düzenli ifadeler ve deterministik sonlu otomatlar (DFA) kullanılarak elle geliştirilmiştir.

Tanımlanan Token Türleri:

- Yorumlar: /\* … \*/ arasındaki ifadeler  
- Anahtar Kelimeler: int, String, if, else, while, return  
- Tanımlayıcılar: değişken/fonksiyon adları  
- Operatörler: +, -, \*, /, =  
- Ayraçlar: ;, :, (), {}  
- Sayılar: 3, 15 vb.  
- String: “…” içindeki ifadeler

# 5. Sözdizimsel Analiz

Yaklaşım: Yukarıdan Aşağıya Çözücü (Top-Down Parser) (Recursive Descent)  
  
CFG’ye dayalı olarak recursive descent yöntemi ile bir sözdizim analizörü (parser) uygulanmıştır. Parser, program yapısının geçerliliğini kontrol eder ve hata durumlarını raporlar.

# 6. Vurgulama Şeması

Gerçek zamanlı vurgulama, her tuş basımında tetiklenir. Token türlerine göre renk şeması aşağıdaki gibidir:

* Anahtar kelimeler → **kırmızı**
* Sayılar → **mavi**
* Tanımlayıcılar → **mor**
* Operatörler→ **yeşil**
* Delimiter (Ayraç)→ **turuncu**
* Yorumlar → **gri**
* String → **kahverengi**

7. GUI Uygulaması

Kullanılan Çerçeve: Tkinter  
  
Arayüz aşağıdaki bileşenleri içerir:  
- Kod girişi için Text widget  
- Gerçek zamanlı renklendirme ve analiz  
- Alt panelde satırlara göre hata mesajları  
  
Kullanıcı kod yazdıkça hem leksik hem de sözdizim analiz işlemleri eş zamanlı çalışır ve kodların uygunluğunu görmek için ortada bulunan tokenize-analiyze butonuna tıklanır.

# 8. Proje Teslimatları

- ✅ Leksiksel Analizör: Python ile DFA tabanlı  
- ✅ Sözdizim Analizörü: Recursive descent parser  
- ✅ Gerçek Zamanlı Vurgulama: Her tuş basımında güncellenir  
- ✅ GUI: Kullanıcı girişi ve çıktı paneli içeren Tkinter arayüzü  
- ✅ Dokümantasyon: Bu rapor ve GitHub README  
- ✅ Demo Videosu: YouTube gibi platformda yayınlandı  
- ✅ Makale: Medium'da yayımlandı

# 9. Karşılaşılan Zorluklar ve Çözümler

• Zorluk: Gerçek zamanlı vurgulamada performans sorunları  
 → Çözüm: Satır satır yerine dize tabanlı tarama ve hafif regex kullanımı  
  
• Zorluk: GUI'nin donmaması  
 → Çözüm: Yoğun hesaplamalar olay döngüsü dışında ertelendi.

# 10. Sonuç

Bu proje, gerçek zamanlı sözdizim vurgulayıcı gerekliliklerini başarıyla yerine getirmiştir. Gerçek zamanlı token tanıma, yukarıdan aşağıya sözdizim analizi, kullanıcıya anlık geri bildirim ve sade bir grafik arayüz sunulmuştur. Gelecekte, daha karmaşık gramer destekleri, çok satırlı analiz ve kod dışa aktarma özellikleri eklenebilir.

Hazırlayan: Aynur Adıbelli

Okul/Bölüm: Bursa Teknik Üniversitesi/ Bilgisayar Mühendisliği

Numara: 22360859008

Tarih: 5 Haziran 2025