Doğu Akdeniz Üniversitesi

BLGM 318 Programlama Dilleri Prensipleri
Bahar 2024-2025

Ödev #3

Ayrıştırıcı (Parser) Uygulaması

İkili gruplar halinde yapılacaktır.

Mantık ifadeleri üretmeye yarayan gramer aşağıda verilmiştir. (+ veya, & ve anlamındadır)

G1:
E → E + T | T
T → T & F | F
F → (E) | t | f

Bu gramer, yukarıdan aşağıya özyinelemeli inen ayrıştırıcı için uygun değildir.

Sol özyinelemeyi kaldırıp aşağıdaki grameri elde ederiz.

G2:
E → T R
R → + T R | ∈
T → F S
S → & F S | ∈
F → (E) | t | f

"∈" boş kural demektir. Yani Z → ∈ ise, Z'den boş diziyi türetebiliriz.

Şimdi G2'ye göre ayrıştırma fonksiyonlarını yazabiliriz (kodumsu olarak). lex() bir sonraki jetonu döndüren fonksiyon olsun.

```
// Global değişkenler:
Bool error = FALSE;
char next_token ;

void main(){
  girdi dosyasını okuma için aç
  next_token = lex();
  E();

if error{
  print("Ayrıştırma başarısız");
  print("tüketilmeyen girdi:", unconusmed_input());
```

```
} else{
     print(“Ayrıştırma başarılı”);
    print("tüketilmeyen girdi:", unconusmed_input());

}


// E -> T R
void E(){
        if (error) return;
        print ("E -> T R");
        T();
        R();
}

// R -> + T R | ∈
void R(){
   if (error) return;
   if (next_token== '+')  {
      print ("R -> + T R");
      next_token = lex();
       T();
       R();
   }
  else  {
     print ("R->∈");
   }
}

/* T -> F S */
void T(){
    if (error) return;
     print (" T -> F S");
       F();
       S();
}

/*          S -> & F S | ∈          */
void S(){
  if (error) return;
   if (next_token=='&') {
     print ("S -> & F S");
      next_token= lex();
       F();
       S();
   }
  else {
     print "S -> ∈"
   }
}
```

```
//       F -> ( E ) | f | t
void F(){
    if (error) return;
   if (next_token=='(' )  {
      print "F->( E )";
      next_token = lex();
      E();
      if (next_token == ')'  ){
        next_token = lex();
      }
     else {  error=TRUE;
               print("hata. ‘)’ bekleniyordu ancak gelen:”, next_token);
               print("tüketilmeyen girdi:", unconusmed_input());
               return;
     }
   }
   else  if  (next_token == ‘t’) {
     print ("F->t");
     next_token = lex();
   }
   else  if  (next_token == ‘f’) {
     print ("F->f");
     next_token = lex();
   }

   else {
        error=TRUE;
        print("hata. ( veya f veya t bekleniyordu ancak gelen:”, next_token);
         print("tüketilmeyen girdi:", unconusmed_input());


   }
}
```

Yapılması gereken:

Python ile yukarıdaki kodumsuyu kullanarak bir ayrıştırıcı yazmanız gerekiyor. *lex()* adında bir fonksiyon tanımlayın. *lex()*'in görevi her çağrıldığında bir sonraki karakteri dosyadan okuyup geri göndermek. Dosyadaki boş karakterleri (space, newline vs.) de atlaması gerekiyor. *unconusmed_input()* dosyada geri kalan (lex() tarafından okunmamış) karakterleri bir string halinde vermeli. *E(), R(), T()* ve *F()* fonksiyonları yukarıda gösterildiği şekilde tanımlanmalı.

<u>Zip'leyip Teams ve Moodle'a yüklemeniz gerekenler:</u>

1. **Aşağıdakileri içeren rapor**
    a. **Problem tanımı (proje neyi başarıyor)**
    b. **Programınızın nasıl çalıştığı bilgisi**
    c. **Programınızın kullanma kılavuzu**
    d. **Programınızın aşağıdaki girdiler ile çalışmasının ekran görüntüleri**
        i. **t & t & f**
        ii. **t & t + f**
        iii. **t & (t + f)**
        iv. **(f) + t & f**
        v. **t**
        vi. **t+f**
        vii. **t&f**
        viii. f

    e. **Programınızın aşağıdaki hatalı girdiler ile çalışmasının ekran görüntüleri**
        i. **t & & f**
        ii. **& t**
        iii. **t f +**
        iv. **((t)**
        v. **(+)**
        vi. **(f + t & f**
        vii. **)) t**
        viii. **(f & f &)**
    f. Rapor ekinde program kodu
    g. Rapor ekinde girdi dosyası

2. **Yorumlar eklenmiş program kodunuz (python dosyası olarak)**

3. **Girdi olarak kullanılan dosya (text dosyası)**

Yükleyeceğiniz dosyanın ismi şu formatta olmalıdır:
Öğrenci1No_Öğrenci2No_BLGM318_odev3.zip