Külyutmaz 2

Ege Üniversitesi Bilgisayar Mühendisliği
Programlama Dilleri
Ödev-3

Umut BENZER 05-06-7670 http://www.ubenzer.com/
Gül DELİORMAN 05-06-7699 http://www.guldeliorman.com/

Teslim tarihi: 04 Mayıs 2009

Amaç

Daha önce oluşturduğumuz lexical analyzerı, geliştireceğimiz bir parse tree algoritması ile birleştirip dosyada verilen kaynak kodu işletip, sonuç bulmak.

Programın İşleyişi

Program çalıştırılınca, işletilecek kaynak koduna ulaşmak için, kodun bulunduğu dosyanın yolu sorulacaktır. Örneğin, **text.txt** dosyasında işletilecek kodlar bulunuyor olsun. Bu dosyayı **C:**'ye koyarsak, sorulan dosya yoluna verilecek cevap: **C:/text.txt** olacaktır.

Program, dosyadan satırları okuyarak işlemeye başlıyor. Bu nedenle dosyanın bulunamaması durumunda **'Check path for the file'** hatasıyla karşılaşıyoruz. EBNF'deki kurallara uygun olarak parse etmeye başladığımız her bir **";"** ile ayrılmış komutlara göre işlemler yapılıyor. Bir komutun geçerli olabilmesi için ";" ile bitmesi gerektiğinden, ";" bulunmayan komut satırı için; **''; 'expected at the end of the line'** hatası veriliyor.

";"leri ayırma işleminden sonra, programın çalışmasını sağlayan önemli kriterlerden biri, kaynak kodunda operation ve variable kısmının **':='** ile ayrılması gerektiğidir. Bu operatörün olmaması durumunda **'Assigment operator ':=' exptected.'** ya da ':=' ifadesinden sonra veya önce bir kod parçası bulunmaması durumunda **'Identifier expected on the left side of assignment operator.' 'An operation expected on the right side of assignment operator.'** hataları verilecektir.

İşlem ve değişken olarak ayırdıktan sonra, hepsi kendi içinde işleyen metotlara ayrılıyor. Değişken adını kontrol etmek üzere **varName** adlı metot yazıldı. Burada karakter harici bir şey (sayı veya boşluk) okunursa **'Unexpected character at line i column j'** hatası vermektedir. Harfleri bulundukları satır ve sütunla birlikte tutmak istediğimizden **CString** classı yaratıldı. Böylece string işlemleri ile ilgili metodlar, yaratılan bu class içinden çağrılabiliyor. Cstring classının bir dezavantajı trim metodunu kullanamamak oldu. Bunu da boşluk ' ', tab '/t' ve satır sonu '/n' karakterlerini kontrol için kullanarak, trim metodunun yaptığını sağladık.

Her bir satır komutundaki işlemleri **nexpr** metoduyla gerçekleştirdik. Bu metod recursivedir. Parse ederek önce sağ tarafta yapılabilecek minimum işlemi yapar. Böyle azalarak biter. Parse etme sırasında, işletilecek kodda yanlışlık olması göz önünde bulundurularak uygun hata mesajları verilmiştir. Tüm hatalarda satır ve sütun numarası verilmektedir**.**

Örnekler:

- **Paratesis mismatch.**
- **Number or variable expected on the right side of (-|+|*).**
- **Variable or constant expected inside the paranthesis.**
- **Variable used before it was initialized.**
- **Unexpected character at line i, cloumn j.**

Ödevde belirtildiği gibi identifier name max **32 karakter** olarak sınırlandırıldı. Alabileceği değer Java **maxInt** ve Java **minInt** arasındadır. Sonuç bu değerler arasında olmazsa **'Overflow adding'**, **'Overflow multiplying'**, **'Overflow subtracting'** hatalarından biri verilir. Derlenirken yani compile timeda sınırların aşılması durumunda **'Constant value out of bounds.'** hatası verilir.

Program, kaynak kodunun içindeki kodları işleterek çalışmaya devam eder ve sonucunda integer bir değeri console'da görebiliriz. Eğer integer bir değer üretilemiyorsa, uygun hata mesajı console'da görülecektir.

Örnekler

Kod

```
aVar:=11+5;
myVar:=aVar-((70*9)-9);
result:=(((aVar-my5Var)*3)-(myVar*467));
```

Sonuç

```
Unexpected character '5' in variable name. Line: 3 Column: 19
```

Kod

```
aVar:=11+5;
myVar:=aVar-((70*9)-9);
result:=(((aVar-myVar)*3)-(myVar*467));
```

Sonuç

```
Result: 284398
```

Kod

```
a:=783-49*b;
c:=a-67*(62+8);
```

Sonuç

```
Variable used before it was initialized. Line: 1 Column: 11
```

Kod

```
A:=4;
B:=12;
result:=(A-3*(B-A))
```

Sonuç

```
-20
```

Külyutmaz'ın İşletişi