**BLG447**

**Derleyici Tasarımı**

**2013 – 2014 Güz Dönemi**

**Ödev 3**

**Ozan Arkan CAN**

**040090573**

**22.12.2013**

**GİRİŞ**

Bu ödevde aşağıda belirtilen gramer için geçerli olmak üzere verilen bir girdinin ayrıştırılabilirliğini kontrol etmek için bir ötele - indirge ayrıştırıcısı tasarlanıp gerçeklenmiştir. Ayrıştırma tablosu oluşturma adımı 2. Ödevde gerçeklenmiştir. Bu ödevde bu tablo yardımıyla verilen bir girdinin ayrıştırılabilirliğini kontrol eden kısmı gerçeklenmiştir.

**Gramer:**

(0) S→ Deyimler $

(1) Deyimler → Deyim

(2) Deyimler → Deyimler ; Deyim

(3) Deyim→ Değişken = İfade

(4) Değişken → id [İfade]

(5) Değişken → id

(6) İfade → id

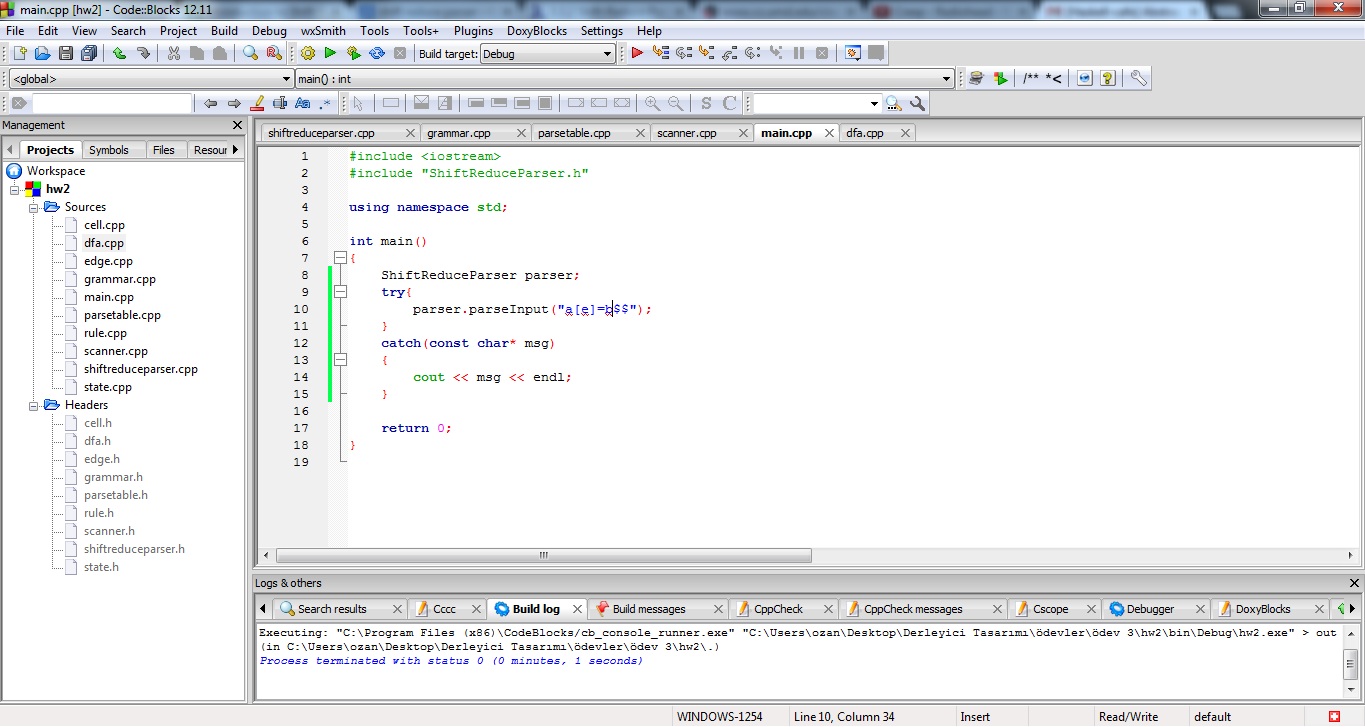
(7) İfade → ( İfade )

Gramerde yer alan terminal simgeler: {[, ], (, ), =, ;, id, $}

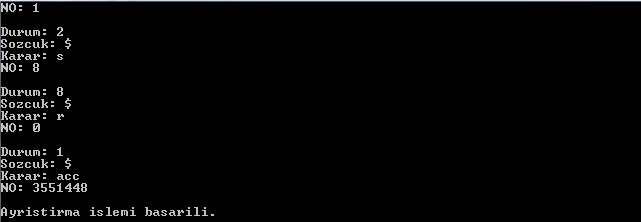
**GELİŞTİRME VE ÇALIŞTIRMA ORTAMLARI**

**Windows:**

Ödev Windows 7 işletim sisteminde g++ destekleyen mingw içeren, Code::Blocks geliştirme ortamında yazılmış ve test edilmiştir.

****

Şekil 1: Uygulamanın geliştirme ortamına dair ekran görüntüsü



C:\Users\ozan\Desktop\Derleyici Tasarımı\ödevler\ödev 3\basarisiz.jpg

Şekil 2 ve 3: Sırasıyla “a[e]=b$$” ve =ab$$” girdileri için programın çıktısı

**Linux:**

Derleme işleme için aşağıdaki komut kullanılmıştır:

g++ \*.h \*.cpp -o dt\_o3

Ancak ShiftReduceParcer.cpp dosyasında kullanılan itoa (integer to string dönüşümü yapan fonksiyon) itü Linux server üzerinde çalışmadı. Yalnız farklı bir işletim sisteminde denenirse çalışabilir neden tanınmadığını belirleyemedim. g++ nın sürümüyle alakalı olabilir.

**KAYNAK DOSYA VE SINIF YAPISI**

***Rule* Sınıfı:** Rule sınıfı gramerdeki kural ı temsil eden sınıftır. Katar yapısında leftside, rightSide, integer tipinde id ve sağ taraftaki sembol sayısını tutan numberOfSymbolsAtRightSide tutan değişkenleri vardır.

***Edge* Sınıfı:** DFA daki durumlardan çıkan ayrıtları temsil eden sınıftır. source ve target şeklinde kaynak ve hedef durumları tutar. Ayrıca arc değişkeniyle bu ayrıtın etiketi tutulur.

***Grammar* Sınıfı:** Dili temsil eden gramere ait kuralların listesini, terminal, nonterminal sembollerin listesini, dfa yı üretirken kullanılmak üzere kuralları sol taraflarındaki sembole göre gruplayan bir map yapısı barındırmaktadır.

***State* Sınıfı:** DFA daki durumları temsil eden sınıftır. İçinde türetme kural elamanlarının listesini, bu durumdan geçilebilen durumlara ait ayrıt listesi, sonlanma durumu, türetme aşamasında son durum olup olmadığını kontrol edecek değişkenleri barındırmaktadır.

***DFA* Sınıfı:** DFA yı oluşturan sınıftır. transition – geçiş, closure – kılıf operasyonlarını gerçekler. buildDFA metodu recursive olarak çalışan generateDFA metodunu kullanarak DFA yı oluşturur. searchState metodu daha önce oluşturulan bir durumun tekrar oluşturulmayıp, DFA yı oluşturma aşamasında bir durumudan bu duruma bağlanmasını sağlar. Diğer yardımcı metotlar geçiş işlemindeki nokta kaydırma hareketini gerçeklerler.

***Cell*****Sınıfı:** Ayrıştırma tablosundaki hücreyi temsil eder. İndirgeme, Ötleme, Geçiş ya da Kabul durumu olup olmadığını gösteren action, geçiş, indirgeme ya da öteleme ile alakalı no ve değişkenlerini barındırır.

***ParseTable* Sınıfı:**  Ayrıştırma tablosunu map yapısından faydalanarak, iki boyutlu şekilde tutar. Ayrıştırma tablosunu DFA ve Grammar sınıfları yardımıyla oluşturur. Ayrıştırma aşamasında kullanılmak üzere, sorgulanan hücreyi döndüren getCell metodu bulunmaktadır.

***Scanner* Sınıfı:** Scanner sınıfı verilen girdide, ayrıştırma aşamasında sıradaki sözcüğün bulunmasını sağlar. Girdi üzerinde index yardımıyla ilerler ve sıradaki sözcüğü saptar.

***ShiftReduceParser* Sınıfı:** Ötele-İndirge yöntemini kullanan ayrıştırma algoritmasının gerçeklendiği sınıftır. Ayrıştırma tablosunu ve yığınını barındırır. parseInput metoduyla verilen bir girdinin ayrıştırılıp ayrıştırılamayacağı sınanır.

***main.cpp*:** Programın örnek çalışmasının yürütüldüğü kaynak dosyadır.

**ÇALIŞMA**

Öncelikle kod ufak bir bug barındırmakta. Girdinin sonunda tek bir ‘$’ karakteri yerine iki tane ‘$’ konması gerekmektedir.

*ShiftReduceParser* sınıfının constructor’ında ayrıştırma tablosu oluşturulur. Oluşturulduktan sonra ekrana bastırılır ve parseTable adlı dosyaya bastırılır. Ayrıştırma tablosunu kullanarak ayrıştırma işlemi gerçekleştirilir.

Ayrıştırmayla ilgili sahte kod aşağıdaki gibidir:

push(s0);

sözcük = tarayıcı.sıradaki\_sözcük();

while (true) {

s = yığın.top();

if ( “indirgeme”) then {

for(i = 1; i <= 2\*|β|; i++)// pop 2\*|β| sembol

stack.pop();

s = stack.top();

push(Sol taraftaki sembol);

push(tablo[s, Sol taraftaki sembol]);

}

else if (“öteleme” ) then {

push(sözcük);

push(ötelenecek durum);

sözcük ← tarayıcı.sıradaki\_sözcük();

}

else if ( “kabul” & sözcük == $ )

“Ayrıştırma başarılı” break;

else “Ayrıştırma hatası”

}

**SONUÇ**

Yapılan bu çalışmada ödev metninde verilen gramere ait bir ötle-indirge ayrıştırıcısı gerçeklenmiştir. Gerçeklenen ayrıştırıcı verilen bir girdi için başarılı bir şekilde, o girdinin ayrıştırılıp ayrıştırılamayacağını, ayrıştırılamıyorsa nerede hata olduğunu bildirebilmektedir.

Bu ödevle birlikte;

* Bir gramere ait dfa nın oluşturulması
* Bu dfa dan ayrıştırma tablosunun oluşturulması
* Oluşturulan tablo kullanarak ayrıştıcının çalışması

İle ilgili tecrübe edinilmiş olundu.