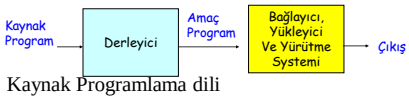
Derleyici Nedir?

- Derleyici:
  - Bir dildeki programı bir diğerine çeviren program
  - Kaynağın anlamsal niteliklerini aynen taşımalıdır
  - Hedef dilde etkin bir program yaratmalıdır
- Tarihsel gelişim:
  - Makina Dili: kodlama, hata ayıklama zor, zahmetli
  - Birleştirici Dilde Programlama: donanıma bağımlı
  - Yüksek Düzeyli Diller: Fortran, Pascal, C, C++,Java
  - İşlemcilerin donanımları çok karmaşıklaştığı ve yazılımı denetlemek çok zorlaştığı için düşük düzeyli dillerde programlama artık mümkün olmamaktadır

1

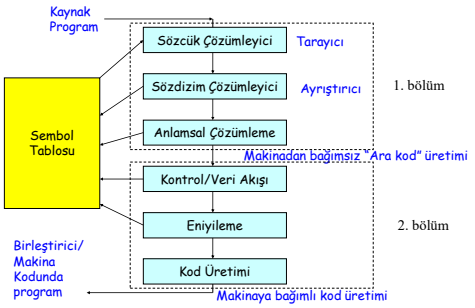
Derleyici Yapısı



- Kaynak Programlama dili
  - Fortran, Pascal, C, C++, Java, vs.
- Hedef Programlama Dili
  - Makina dili, birleştirici dil (assembly code)
  - Yüksek düzeyli diller
- Derleme zamanı – Yürütme zamanı
  - Derleme zamanı statiktir– değişkenler konumlandırılır
  - Yürütme zamanı dinamiktir – SP (yığın işaretçisi) değeri, heap

2

Bir Derleyicinin Genel Yapısı



3

Sözcük Çözümleyici (Tarayıcı)

- Kaynak programın en alt düzeyli birimlerini (sözcükler) belirler ve ayırdeder. Sözcük tek başına bir anlam taşıyan karakter katarıdır:
  - Anahtar sözcükler: for, if, while, vs.
  - Değişken adları: "i", "j", "toplam"
  - Sabitler: 3.14159, 17
  - Noktalama İşaretleri ve operatörler: "(", ")", ",", "+"
- Programlama diline ait olmayan simgeleri siler, örneğin boşluklar, açıklamalar, vs.
- Bir "Sonlu Durum Makinası" ile gerçeklenir

4

Sözcük Çözümleyici (2)

- Sözcükler bir bilgi çifti ile gösterilirler:
  (sözcük tipi, sözcük değeri)
  - sözcük tipi: sözcüğün ait olduğu grup
  - sözcük değeri: sözcüğün sembol tablosundaki adresi
- Tarayıcı Sözdizim Çözümleyicinin bir altprogramı olarak çalışır.
  - Ayrıştırma işlemi sırasında gerek duyuldukça, bir sonraki program simgesini elde etmek üzere tarayıcı çağrılır.

5

Sözdizim Çözümleyici (Ayrıştırıcı)

- Kaynak programın dilin gramer yapısına uygunluğunu irdeler
  - Sözcüklerin dilin belirlemelerine uygun bir sırada bulunup bulunmadıklarını belirler (sentaks analizi)
  - Sözcükleri, kod üretme aşamasında yararlanılacak olan bir ağaç yapısına yerleştirir
- Derleyicinin gerçeklenmesi en zor bölümüdür
  - Programlama dilinin sözdizimsel tanımları BNF formunda tanımlanır
  - Bu tanımlardan yola çıkarak bir ayrıştırıcı üretilir

6

Anlamsal Çözümleme

- Yerine getirilmesi gereken işlemler
  - Değişkenlerin tanımlarını ve doğru kullanıldıklarını kontrol etme
  - Yüklenmiş operatörlere ilişkin işlemler
  - Kaynak koddan "arakod" üretme
    - Ara kod (IR-intermediate representation) donanımdan bağımsız olan, düşük gelişmişlik düzeyinde kod
- Ara kod üretimi ayrıştırma işlemlerine paralel olarak yürütülür
  - Sözdizimle yönlendirilmiş çeviri-oluşturulan ağaç yapısına dayanarak program deyimleri için ara kod üretilir

7

Anlamsal Çözümleme (2)

- Arakod üretimi
  - Deyimler, çevrimler, vs. gibi yüksek düzeyli yapılar üzerinde çalışır.
  - Bu yapıların içerdiği anlamsal işlemler simgesel dil düzeyindeki arakod deyimleriyle ifade edilir
- Arakod
  - Donanımdan bağımsızdır
  - 3-adresli kod –> A = B op C
  - Sonsuz sayıda sanal saklayıcılar ve kaynaklar varsayılır
  - "Standart" işlemler kümesine sahiptir
    - load/store mimarisi

8

Anlamsal Çözümleme (3)

- Amaç
  - Kod niteliğinin iyileştirilmesi için uygun bir altyapı oluşturma
  - Tekrar kullanılabilirlik
    - Farklı donanımlar için, bir derleyicinin farklı versiyonlarına gerek duyulur. Derleyicinin bu bölümü donanımdan bağımsız olacağından, farklı versiyonların geliştirilmesinde aynen kullanılabilir.

Kontrol/Veri Akışı

- Bir dönüşümün doğru olup olmadığına karar verebilmek için değişkenlerin kullanımı ve kontrol akışı hakkında gerekli bilgileri sağlamak
- Veri akışı çözümlemesi
  - Değişkenlerin ne zaman bozulmamasi gereken değerler taşıdıklarını belirleme
  - Hangi komutların değerler ürettikleri veya değerleri tükettiklerini belirleme
- Kontrol akışı çözümlemesi
  - Kontrol deyimlerinin yarattığı yürütmeye ilişkin davranışların belirlenmesi - If, for/while çevrimleri, goto

Eniyileme

- Amaç daha hızlı çalışan kod üretmek-etkin ve küçük
- Genel eniyileme yöntemleri
  - Ölü kodun yok edilmesi- yararsız kodun silinmesi
  - Ortak alt ifadelerin kaldırılması - aynı ifadeyi tekrar hesaplayan komutların silinmesi (üretilen arakod içinde sık rastlanabilir)
  - Boyutlu elemanlara erişirken indis hesabında eniyileme
  - Döngü eniyileme – her çevrimde aynı sonucu üreten ifadelerin döngünün dışına taşınması

Kod Üretimi

- Donanımdan bağımsız arakoddan hedef donanımın simgesel/makina koduna dönüşüm
- Sanal makinadan fiziksel makinaya geçiş
  - Komut seçimi – arakod işlemlerini gerçekleyecek en iyi makina komutlarının seçimi
  - Saklayıcı kullanımı – sonsuz sanal saklayıcıdan donanımın belirlediği saklayıcılara geçiş
  - Simgesel dilde kod üretilmiş ise, birleştirici program devreye girer ve makina kodunda hedef program üretir

Program

1	Giriş
2	Derleyici ve dönüştürücüler: temel kavramlar
3	Sözcük çözümleyici : düzgün ifadeler, otomatlar
4	Sözcük çözümleyici Tasarımı
5	Programlama dillerinin sözdizimsel tanımları
6	Sözdizimsel çözümleme (ayrıştırma ağaçları)
7	Ayrıştırma yöntemleri: aşağıdan-yukarı ve yukarıdan-aşağı ayrıştırma
8	Rekürsif-iniş ayrıştırıcısı
9	Sınıf çalışması
10	LR (Left-Right) ayrıştırıcısı
11	Sözdizimle yönlendirilen çeviri, sözdizim ağaçları,
12	Aritmetik ve lojik ifadeler ve kontrol deyimleri için ara kod üretimi
13	Sembol tablosu ve bellek yönetimi
14	Kod optimizasyonu ve üretimi

Başarı Değerlendirme Bilgisi

- arasınav - %40 : **14 Kasım 2018**
- final - %40
- Proje - %20

- Vize alma (final sınavına girebilme) koşulu:
  %70 devam koşulunu sağlama

- Kitap: COMPILERS: PRINCIPLES, TECHNIQUES, AND TOOLS", A.V.AHO, R.SETHI, J.D.ULLMAN, PRENTICE HALL, 2003