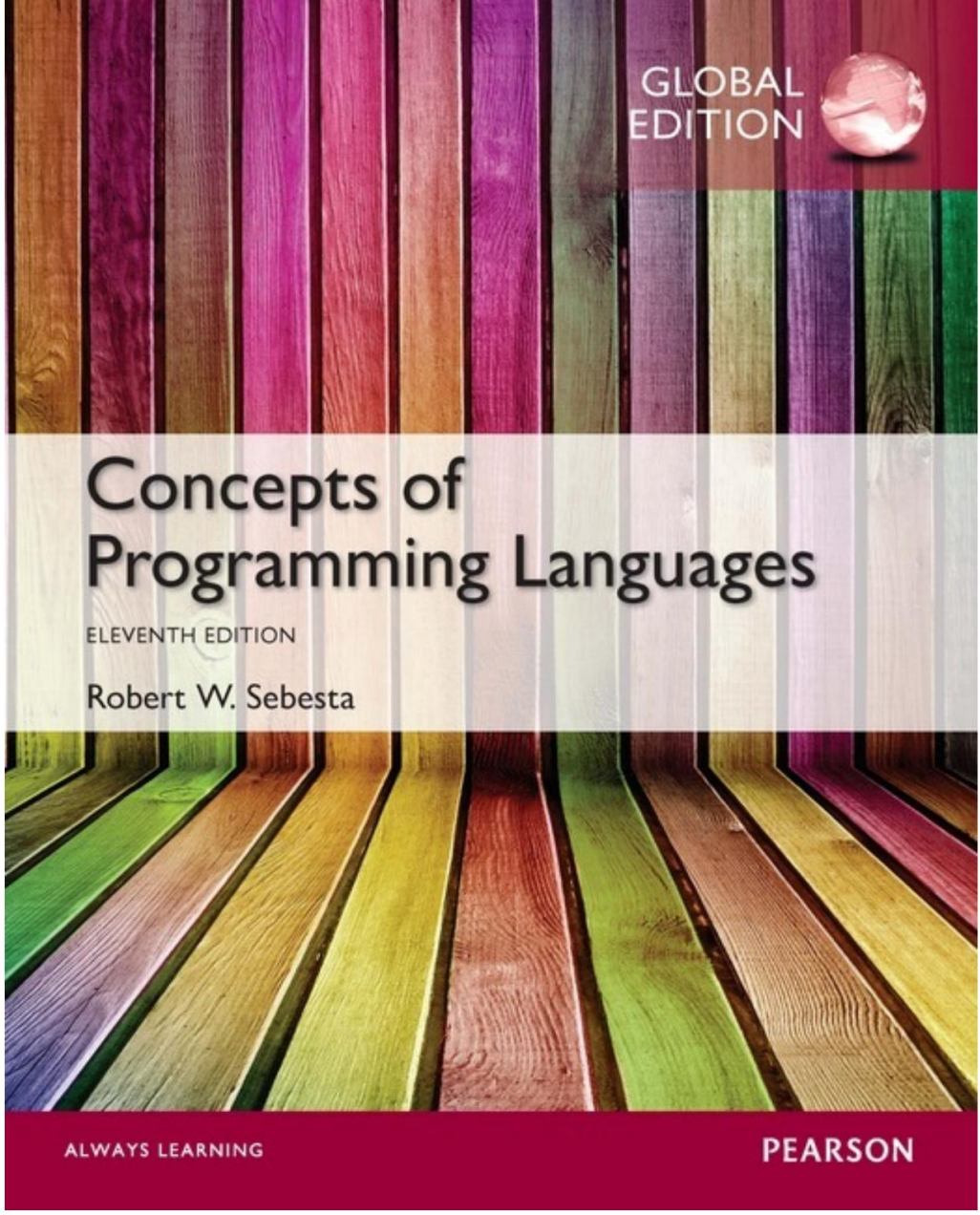
Bölü m 1

ön elemeler



1. Bölü m Konuları

- Kavramları Çalış mak için Nedenler
  Programlama dilleri
- Programlama Etki Alanları •

Dil Değ erlendirme Kriterleri • Dil

Tasarımı Üzerindeki Etkiler • Dil

Kategorileri • Dil Tasarımı Dengelemeleri

- Uygulama Yöntemleri • Programlama

Ortamları

Kavramları Çalış mak için Nedenler Programlama dilleri

- Artan fikirleri ifade etme yeteneğ i
- Uygun dilleri seçmek için iyileş tirilmiş arka plan
- Artan yeni diller öğ renme yeteneğ i •

Uygulamanın öneminin daha iyi anlaş ılması

- Halihazırda var olan dillerin daha iyi

kullanılması bilinen

- Bilgisayarın genel geliş imi

Programlama Etki Alanları

- Bilimsel uygulamalar
  - – Çok sayıda kayan nokta hesaplaması; dizilerin kullanımı - Fortran
- İ ş uygulamaları
  - – Raporlar ü retin, ondalık sayılar ve karakterler kullanın – COBOL • Yapay zeka – Manipü le edilen sayılar yerine

semboller; bağ lantılı listelerin kullanımı – LISP • Sistem

programlama – Sü rekli kullanım nedeniyle verimlilik gerekiyor – C • Web Yazılımı – Eklektik dil koleksiyonu: iş aretleme (örn. HTML),

komut dosyası oluş turma (örn. PHP), genel amaçlı (örn. Java)

Dil Değ erlendirme Kriterleri

- Okunabilirlik: kullanım kolaylığ ı
  programlar okunabilir ve anlaş ılabilir
- Yazılabilirlik: bir
  programlar oluş turmak için dil kullanılabilir
- Gü venilirlik: spesifikasyonlara uygunluk
  (yani, spesifikasyonlarına uygundur)
- Maliyet: nihai toplam maliyet

Değerlendirme Kriterleri: Okunabilirlik

- Genel sadelik
  - Yönetilebilir bir dizi özellik ve yapı
  - Minimum özellik çokluğu
  - Minimum operatör aşırı yüklemesi
- Ortogonallik
  - Nispeten küçük bir dizi ilkel yapı, nispeten az sayıda yolla birleştirilebilir.
  - Mümkün olan her kombinasyon yasaldır
- Veri türleri
  - Yeterli önceden tanımlanmış veri türleri
- Sözdizimi konuları
  - Tanımlayıcı formlar: esnek kompozisyon
  - Özel kelimeler ve birleşik ifadeler oluşturma yöntemleri
  - Biçim ve anlam: kendini açıklayan yapılar, anlamlı anahtar kelimeler

Değerlendirme Kriterleri: Yazılabilirlik

- Sadelik ve diklik
  - Birkaç yapı, az sayıda ilkel, bunları birleştirmek için küçük bir dizi kural
- Soyutlama desteği
  - Karmaşık yapıları veya işlemleri ayrıntıların göz ardı edilmesine izin verecek şekilde tanımlama ve kullanma becerisi
- dış avurumculuk
  - İşlemleri belirlemenin bir dizi nispeten uygun yolu
  - Operatörlerin ve önceden tanımlanmış fonksiyonların gücü ve sayısı

Değ erlendirme Kriterleri: Gü venilirlik

- Tip kontrolü
  - Tip hataları iğin test •

İ stisna iş leme
  - Çalış ma zamanı hatalarını durdurun ve dü zeltici önlemler

alın • Aliasing
  - İ ki veya daha fazla farklı referans verme yönteminin varlığ ı
    aynı bellek konumu •

Okunabilirlik ve yazılabilirlik
  - "Doğ al" ifade biğimlerini desteklemeyen bir dil
    bir algoritma "doğ al olmayan" yaklaş ımların kullanılmasını gerektirecek ve
    dolayısıyla gü venilirliğ i azaltacaktır.

Değerlendirme Kriterleri: Maliyet

- Programcıları dili kullanmak için eğitmek • Program yazmak (belirli uygulamalara yakınlık) • Programları derlemek • Programları yürütmek • Dil uygulama sistemi: serbest derleyicilerin mevcudiyeti • Güvenilirlik: Düşük güvenilirlik yüksek sonuçlara yol açar

  maliyetler

- Programları sürdürmek

Değ erlendirme Kriterleri: Diğ erleri

- Taş ınabilirlik
  - Programların bir uygulamadan diğ erine geçiş kolaylığ ı • Genellik
  - Çok çeş itli uygulamalara uygulanabilirlik
- İ yi tanımlanmış lık
  - Dilin resmi tanımının eksiksizliğ i ve kesinliğ i

Dil Tasarımına Etkiler

- Bilgisayar Mimarisi
  - Diller, mimari olarak bilinen yaygın bilgisayar mimarisi etrafında geliş tirilir. von Neumann
- Program Tasarım Metodolojileri –

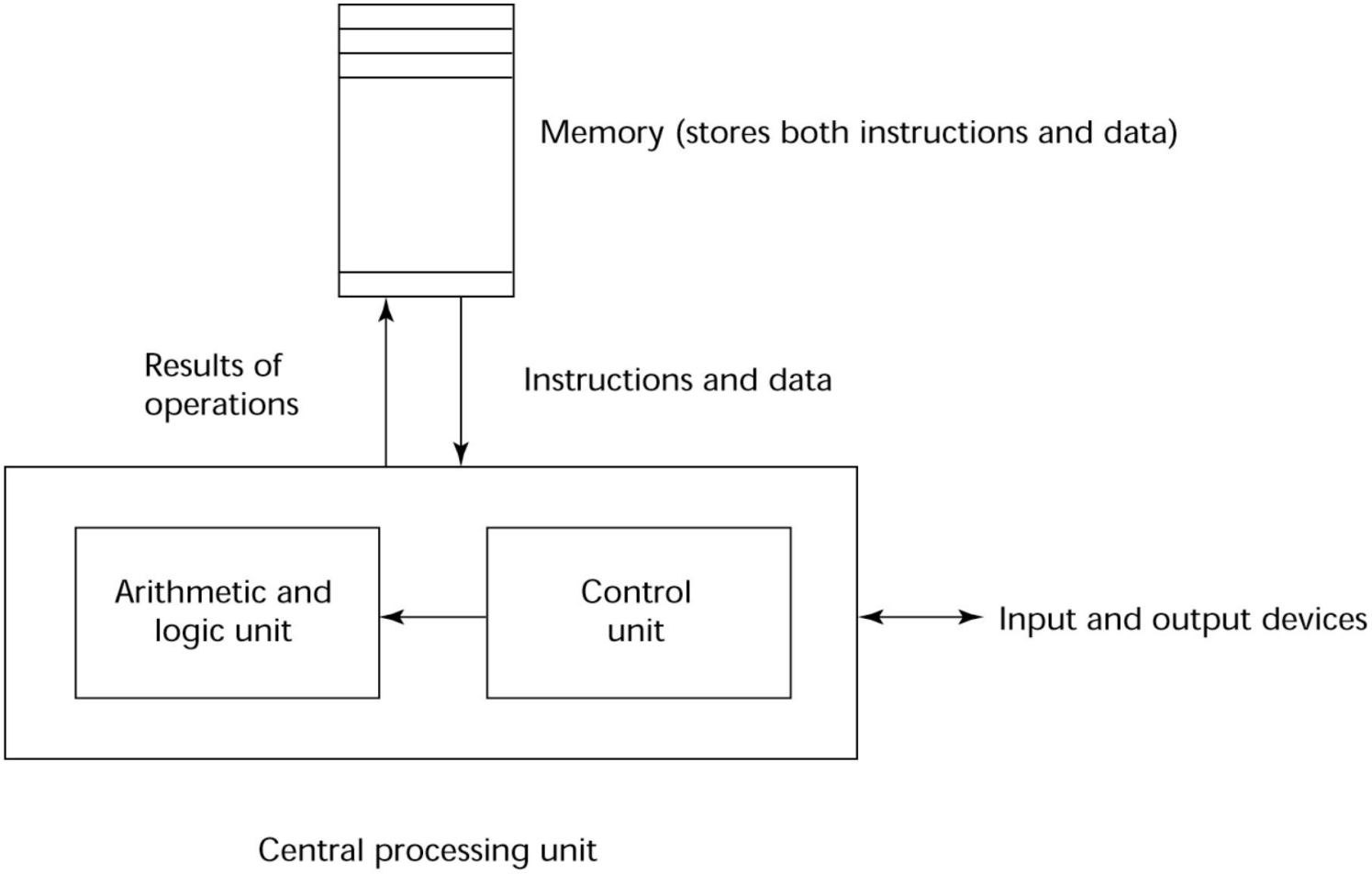
  Yeni yazılım geliş tirme metodolojileri (örneğ in, nesne yönelimli yazılım geliş tirme) yeni programlama paradigmalarına ve buna bağ lı olarak yeni programlama dillerine yol açtı

Bilgisayar Mimarisi Etkisi

• İ yi bilinen bilgisayar mimarisi: Von Neumann • Von Neumann bilgisayarları nedeniyle en baskın olan zorunlu diller

– Bellekte saklanan veriler ve programlar
- Bellek CPU'dan ayrıdır
– Talimatlar ve veriler bellekten CPU'ya iletilir
– Zorunlu diller için temel
• Değ iş kenler bellek hü crelerini modelleme • Atama ifadeleri model borulama • Yineleme verimli

Von Neumann Mimarisi



Von Neumann Mimarisi

- Getirme-yürütme döngüsü (von Neumann'da mimari bilgisayar)

```
program sayacını başlat
sonsuza kadar tekrar et
    sayacın gösterdiği talimatı getir
    sayacı artır
    talimatın kodunu çöz
    talimatı uygula
tekrarı bitir
```

Programlama Metodolojilerinin Etkileri

- 1950'ler ve 1960'ların baş ı: Basit uygulamalar; makine verimliliğ i hakkında endiş e
- 1960'ların sonu: İ nsanların verimliliğ i önemli hale geldi; okunabilirlik, daha iyi kontrol yapıları – yapılandırılmış programlama – yukarıdan aş ağ ıya tasarım ve adım adım iyileş tirme • 1970'lerin sonu: Sü reç odaklıdan veri odaklıya

  - veri soyutlama

- Orta 1980'ler: Nesneye yönelik programlama
  – Veri soyutlama + kalıtım + polimorfizm

Dil Kategorileri

- Zorunlu
  - Merkezi özellikler değ iş kenler, atama ifadeleri ve yineleme
  - Nesne yönelimli programlamayı destekleyen dilleri dahil edin – Komut dosyası dillerini dahil edin – Görsel dilleri ekleyin – Örnekler: C, Java, Perl, JavaScript, Visual BASIC .NET, C++ • Fonksiyonel – Hesaplama yapmanın ana yolu, verilen parametrelere fonksiyonları uygulamaktır
  - Örnekler: LISP, Scheme, ML, F# •
- Mantık
  - Kural tabanlı (kurallar belirli bir sırayla belirtilmez)
  - Örnek: Prolog •
- İ ş aretleme/programlama melezi
  - İ ş aretleme dilleri, bazı programlamaları destekleyecek ş ekilde geniş letildi
  - Örnekler: JSTL, XSLT

Dil Tasarımı Ödü nleri

- Gü venilirlik ve yü rü tme maliyeti – Örnek: Java, dizi
  öğ elerine tü m baş vuruları talep eder
  artan yü rü tme maliyetlerine yol açan uygun indeksleme için kontrol edilmelidir

- Okunabilirlik ve yazılabilirlik Örnek: APL
  birçok gü çlü operatör (ve çok sayıda yeni sembol) sağ layarak karmaş ık hesaplamaların kompakt bir programda yazılmasına izin verir, ancak bunun pahasına okunabilirlik zayıftır

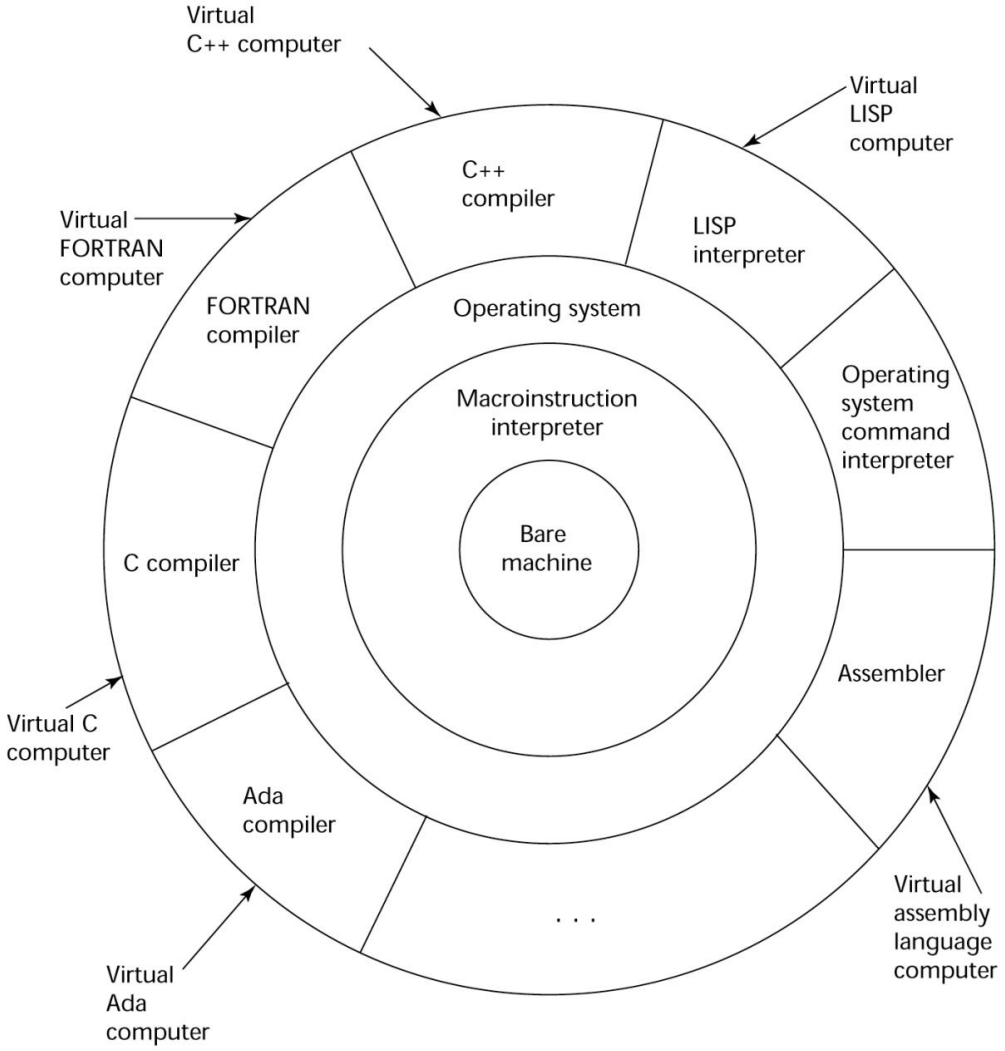
- **Yazılabilirlik (esneklik) ve gü venilirlik**
  – Örnek: C++ iş aretçileri gü çlü ve çok esnektir ancak gü venilmezdir

Uygulama Yöntemleri

- Derleme
  - Programlar makine diline çevrilir; içerir JIT sistemleri
  - Kullanım: Bü yü k ticari uygulamalar
- Saf Yorum
  - Programlar, olarak bilinen baş ka bir program tarafından yorumlanır. Bir tercü man
  - Kullanım: Kü çü k programlar veya verimlilik sorun olmadığ ında
- Hibrit Uygulama Sistemleri
  - Derleyiciler ve saf yorumlayıcılar arasında bir uzlaş ma
  - Kullanım: Verimliliğ in ilk endiş e olmadığ ı kü çü k ve orta ölçekli sistemler

Bilgisayarın Katmanlı Görü nü mü

İ ş letim sistemi ve dil uygulaması, bir bilgisayarın makine arayü zü ü zerinde katmanlanmış tır.



Derleme

- Üst dü zey programı (kaynak dil) makine koduna (makine dili) çevirin
- Yavaş çeviri, hızlı yü rü tme • Derleme

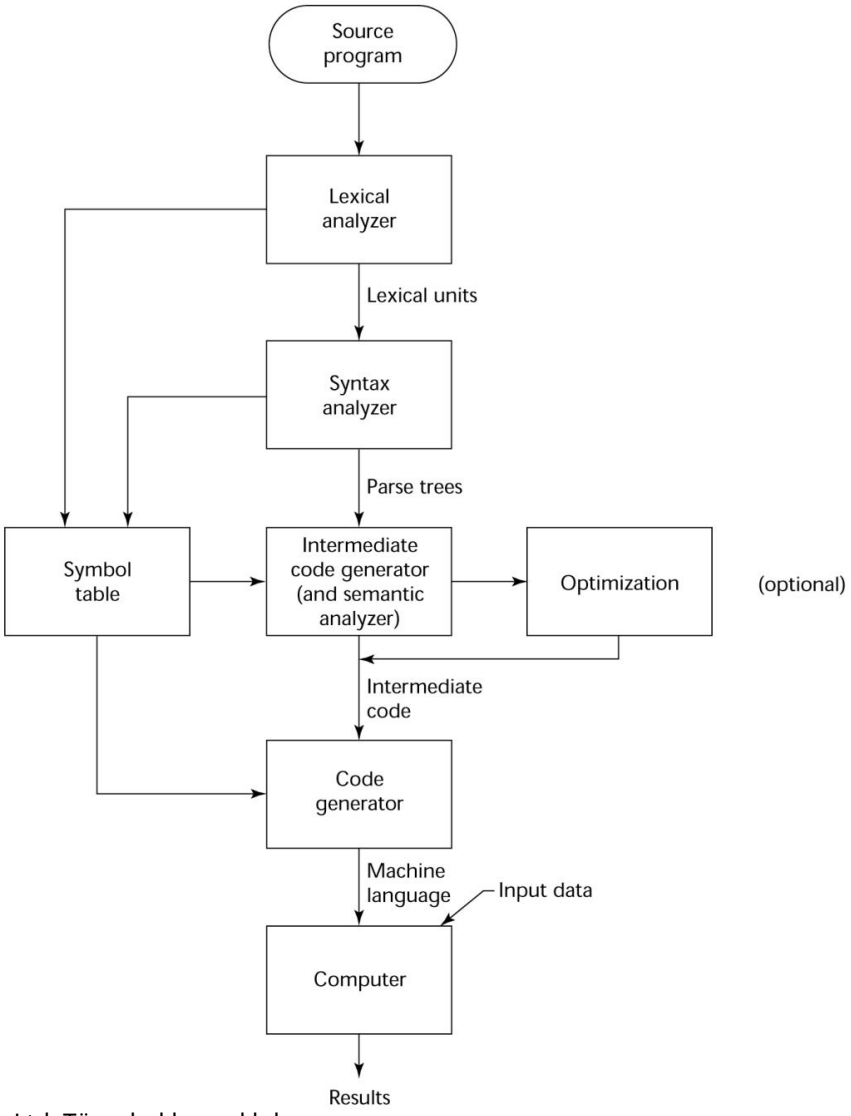
iş leminin birkaç aş aması vardır:

– sözcü k analizi: kaynak programdaki karakterleri sözcü k birimlerine dönü ş tü rü r

– sözdizimi analizi: sözcü k birimlerini programın sözdizimsel yapısını temsil eden ayrış tırmak ağ ağar

– Semantik analiz: ara kod oluş turma – kod oluş turma: makine kodu oluş turulur

Derleme Sü reci



Ek Derleme Terminolojileri

- Yü kleme modü lü (yü rü tü lebilir görü ntü): kullanıcı ve sistem kodu birlikte • Bağ lama ve yü kleme :
  sistem programı birimlerini toplamak ve bunları bir kullanıcı programına bağ lamak

Von Neumann Darboğazı

• Bir bilgisayarın belleği ile işlemcisi arasındaki bağlantı hızı bilgisayarın hızını belirler • Program talimatları genellikle bağlantı hızından çok daha hızlı yürütülebilir; bağlantı hızı böylece bir darboğaz

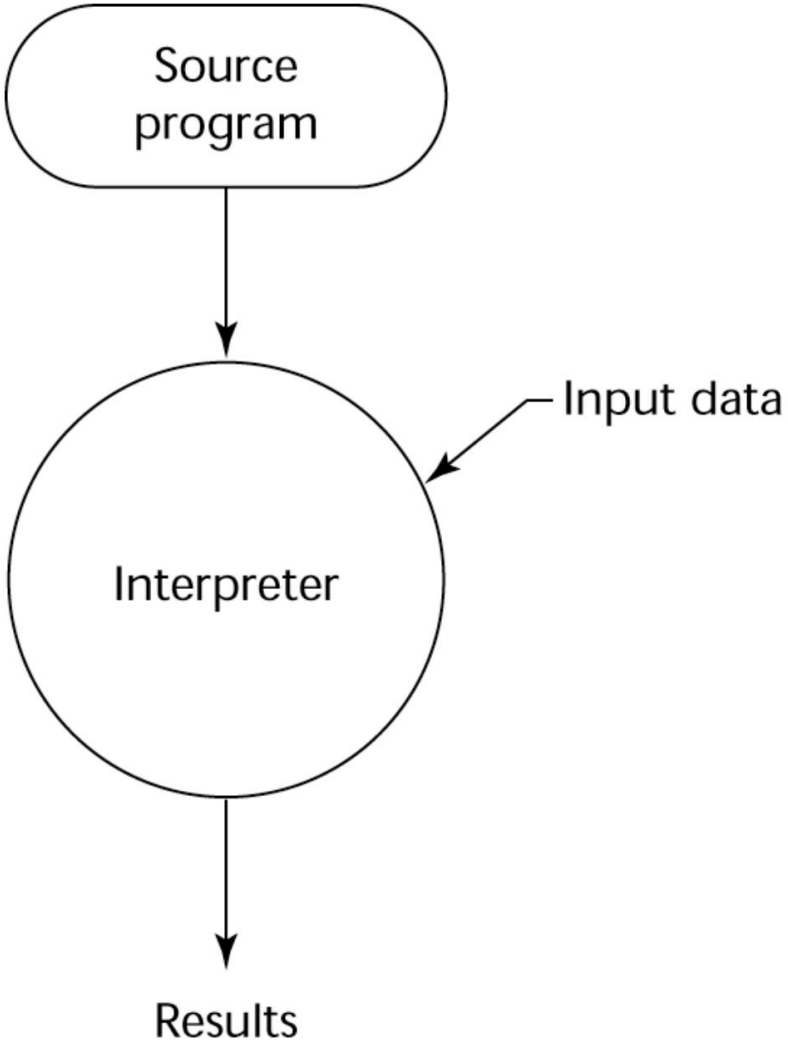
• Olarak bilinir Neumann darboğazı ; bilgisayarların hızındaki birincil sınırlayıcı faktördür

Saf Yorum

- Çeviri yok
- Programların daha kolay uygulanması (çalış ma zamanı hatalar kolayca ve hemen görü ntü lenebilir) • Daha yavaş yü rü tme (derlenmiş programlardan 10 ila 100 kat daha yavaş) • Genellikle daha fazla alan gerektirir • Artık geleneksel yü ksek seviyeli diller için nadirdir • Bazı Web komut dosyalarıyla önemli geri dönü ş

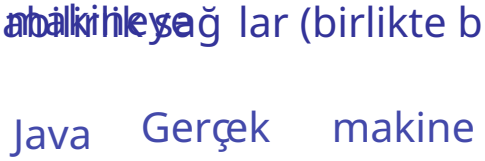
  diller (örn. JavaScript, PHP)

Saf Yorumlama Sü reci

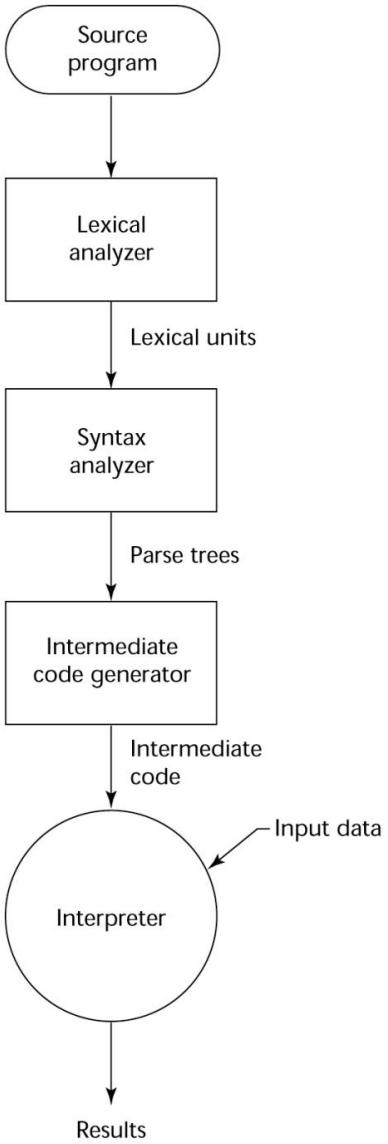


Hibrit Uygulama Sistemleri

- Derleyiciler ve saf tercü manlar arasında bir uzlaş ma
  - Yü ksek seviyeli bir dil programı, kolay yorumlamaya izin veren bir ara dile çevrilir
- Saf yorumlamadan daha hızlı
- Örnekler
  - Perl programları, yorumlamadan önce hataları algılamak için kısmen derlenir
  - Java'nın ilk uygulamaları hibritti; ara form, bayt kod yorumlayıcısı ve çalış ma zamanı sistemi olarak bilinen bayt kod gezginlerinden oluşur (birlikte bunlara JVM denir)



Hibrit Uygulama Sü reci



Tam Zamanında Uygulama Sistemleri

- Başlangıçta programları bir ara dile çevirin
- Ardından, alt programların ara dilini, çağrıldıklarında makine kodunda derleyin.
- Sonraki çağrılar için makine kodu versiyonu tutulur • JIT sistemleri Java programları için yaygın olarak kullanılır • .NET dilleri bir JIT sistemi ile uygulanır • Özünde, JIT sistemleri gecikmeli derleyicilerdir

öniş lemciler

- Öniş lemci makroları (talimatları)
  yaygın olarak baş ka bir dosyadaki kodun dahil edileceğ ini belirtmek için kullanılır
- Bir öniş lemci, gömü lü öniş lemci makrolarını geniş letmek için program derlenmeden hemen önce bir programı iş ler

- İ yi bilinen bir örnek: C öniş lemcisi – #include, #define ve benzerlerini geniş letir
  makrolar

Programlama Ortamları

- Yazılım geliş tirmede kullanılan araçlar koleksiyonu • UNIX

  – Daha eski bir iş letim sistemi ve araç koleksiyonu –

  Gü nü mü zde genellikle bir GUI (örneğ in, CDE, KDE veya UNIX'in ü zerinde çalış an GNOME)

- Microsoft Visual Studio.NET

  – Bü yü k, karmaş ık bir görsel ortam •

Herhangi bir .NET dilinde Web uygulamaları ve Web dış ı uygulamalar oluş turmak için kullanılır

- NetBeans –

  Aş ağ ıdaki uygulamalar hariç, Visual Studio .NET ile ilgili Java

Özet

- Programlama dillerinin incelenmesi birkaç nedenden dolayı değ erlidir:
  - Farklı yapıları kullanma kapasitemizi artırın
  - Dilleri daha akıllıca seçmemizi sağ layın
  - Yeni dilleri öğ renmeyi kolaylaş tırır •

Programlamayı değ erlendirmek için en önemli kriterler diller ş unları içerir:
  - Okunabilirlik, yazılabilirlik,

gü venilirlik, maliyet • Dil tasarımı ü zerindeki baş lıca etkiler makine mimarisi ve yazılım geliş tirme metodolojileri olmuş tur • Programlama dillerini uygulamanın ana yöntemleri ş unlardır: derleme, saf yorumlama ve hibrit uygulama