Programlama Dilleri

TECHNISCHE UNIVERSITÄT WIEN

Doc. Dr .Mehmet Akif Cifci

- Viyana Teknik Üniversitesi (Avusturya)
- Klaipeda Üniversitesi (Litvanya)
- Bandırma Onyedi Eylül Üniversitesi

Klaipeda University

PROGRAMLAMA DİLLERİ

https://github.com/themanoftalent

İçerik

https://github.com/themanoftalent

Program Nedir?

Program bir bilgisayar sistemine yapması gereken işleri tam olarak tanımlamaya yarayan tanımlar kümesidir.

https://github.com/themanoftalent

Bu programların yazım kurallarına da programlama dili denir. Doğal dillerde olduğu gibi programlama dillerinin de kuralları vardır.

Doğal dillerin aksine bu kurallardan en küçük bir sapma yapıldığında bile program çalışmaz .

https://github.com/themanoftalent

Örneğin kullanılmakta olan Windows işletim sistemi on binlerce sayfa, milyonlarca bilgisayar kodunun bir araya gelmesinden oluşur. İşte işin tüm zorluğu, o kodların arasında dolaşan, hata mesajlarını günlerce kodları gözden geçirerek arayan, sandalye tepesinde saatlerce aynı ekrana bakarak ilginç görüntüler sergileyen programcıların üzerindedir

https://github.com/themanoftalent

PROGRAMLAMA DİLLERİNİN ÖZELLİKLERİ

1-Giriş / Çıkış Komutları: Bu komutlar, bilgisayara verileri almalarını ve sonuçlarını sergilemelerini belirtir. Verilen veri, çıktıların yönlendirildiği cihazlar bu komutlarda belirtilir.

2-Hesaplama Komutları: Bu komutlar, temel matematik işlemlerini yapmaya yarayan komutlardır.

- **Kontrolün Yönlendirilmesi İçin Kullanılan Komutlar:** Bu komutlar, programın normal komut akışından sapabilmek için kullanılır.

- **Verilerin Bilgisayar İçinde Taşınması, Saklanması ve Geri Çağrılmasına Yönelik Komutlar:** Bu komutlar, verileri çeşitli bellek adresleri arasında olduğu kadar diskle bellek arasında da hareket ettirmek amacıyla kullanılır.

Programlama dillerinin en önemli özelliklerinden birisi makineden bağımsız olabilmeleridir. Programların taşınabilirliği olarak da adlandırılan bu özellik programın geliştirildiği bilgisayardan başka bir bilgisayarda da çalışabilmesi anlamına gelir.

https://github.com/themanoftalent

PROGRAMLAMA DİLİ SINIFLANDIRMALARI

Bir algoritmanın doğrudan doğruya makine diline çevrilmesi ile elde edilen programlarla karmaşık işler yapabilen yazılımların geliştirilmesi neredeyse imkansızdır.

Algoritmalardaki kavramsal adımları makine diline çevirebilmek için birçok ara dil geliştirilmiştir. Bu diller, programcıya saklayıcılar, bellek adresleri, makine döngüsü gibi donanımla ilgili detaylar arasında kaybolmadan programlama olanağı sağlarlar.

2-1-Geleneksel Sınıflandırma

- Çok yüksek seviyeli diller : VisualBasic, Paradox, Acces , Foxpro ...
- Yüksek seviyeli diller: Pascal ,Basic ,Fortran...

  Orta seviyeli diller: C ,C++, Java ,ADA...
- Düşük seviyeli diller: Assembly...
- Makina dilleri: Bilgisayarın çalışma dilleri 1 ve 0'lardan oluşur...

1-Makine Dilleri

Makine dili mikroişlemci ya da mikro denetleyici gibi komut işleme yeteneğine sahip entegrelerin işleyebilecekleri komutlardan ve buna uygun söz diziminden oluşan dile verilen addır.

https://github.com/themanoftalent

1 0011 0111

- Sol baştaki 1 bit indirect adresleme yapıldığını
- Sonraki 4 bit çarpma işleminin yapılması gerektiğini
- Diğer 4 bit ise hafızanın 0111 adresine gidilmesi gerektiğini belirtir

https://github.com/themanoftalent

Makine diline çevrilmiş bir komut iki parçadan oluşur

Operatör: Herhangi bir komuta karşılık gelen işlemin kodudur.

Operant: Üzerinde işlem yapılan veriyi ya da verinin adresini tutan koddur.

https://github.com/themanoftalent

2-Düşük Seviyeli Diller (Assembly)

Assembly dili, karmaşık programlar yazmak için kullanılan düşük seviyeli bir programlama dilidir. Assembly insanlar tarafından anlaşılması zor olan makine dilinin sayısal ifadelerini, insanlar tarafından anlaşılarak programlanması daha kolay olan alfabetik ifadelerle değiştirerek düşük seviyede programlama için bir ortam oluşturur.

https://github.com/themanoftalent

Assembly dilini makina koduna çeviren programlara assembler denir. Bir assembler'ı derleyiciden ayıran en önemli özellik bire bir dönüşüm yapmasıdır. Derleyiciler kodun tamamını okurlar ve kodun tamamını anlamlı bir programa dönüştürürler. Kodun her satırını tek tek okuyan ve uygulayan programlar ise yorumlayıcı denir.

https://github.com/themanoftalent

3-Orta Seviyeli Diller

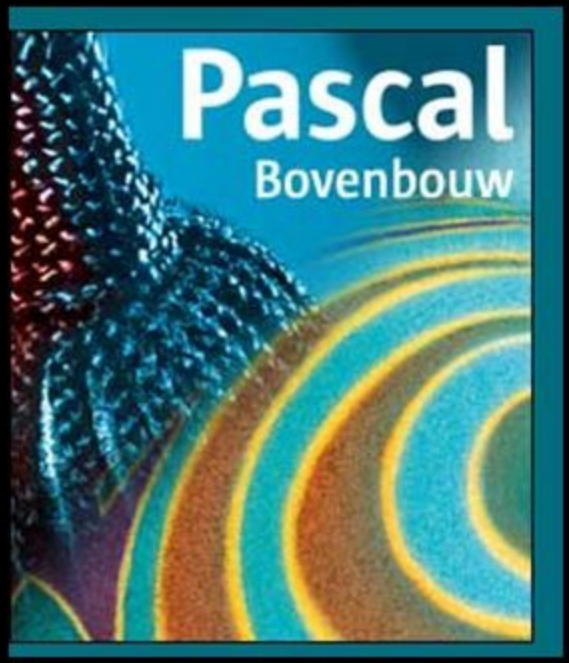
Doğal dillere ve matematik işlemlere makine dili ve Assembly'e göre daha yakındırlar. En önemli tarafi ise üst düzey dillerin makinelerden bağımsız olarak çalışmalarıdır.

C, Java, C++ orta seviyeli programlama dillerine örnek olarak verilebilir.

https://github.com/themanoftalent

4- Yüksek Seviyeli Diller

1960'larda yaygınlaşmıştır.3. kuşak dillerde denir. İngilizceye benzer ifadelerin kullanıldığı, aritmetik işlemlerde kullandığımız işaretlerin kullanıldığı, daha hızlı ve kolay program yazmayı sağlayan, programcının daha az çaba sarfettiği dillerdir

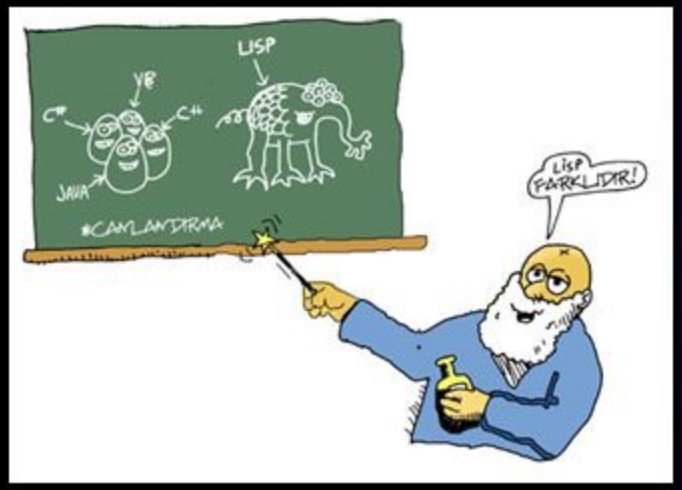


Pascal, Fortan, Cobol, Algol, vb

https://github.com/themanoftalent

5- Çok Yüksek Seviyeli Diller

4. kuşak dillerde denir. 3. kuşak bir dilde onlarca yüzlerce satır halinde kodlanan bir pogram, 4.kuşak dillerle birkaç satırda ifade edilir. Çünkü bu dillerle sadece "NE" istediğimizi bildiririz, 3. kuşak dillerde olduğu gibi "NASIL" yapılacağını söylemeyiz.



Örnek olarak LISP, SNOBOL, SQL verilebilir.

https://github.com/themanoftalent

2-2- Programlama Paradigmaları

Bu bölümde programlama dillerinin gelişimi sırasında temel kuralların programlama dillerine nasıl yansıdığına bakıp bu dilleri değişik kategorilerde sınıflandırmaya çalışacağız

https://github.com/themanoftalent

Programlama dilleri

https://github.com/themanoftalent

İmperatif Diller:

- İmperatif programlama dilleri geleneksel programlama kavramları üzerine kurulmuştur. Özellikleri, durum ve bu durumları değiştirecek komutlara dayalı olmasıdır.
- Bu dillerde, program mantığının geliştirilmesi ve programın çalışması sırasındaki kontrol işlemleri programcının sorumluluğundadır. Programda sonuçların tam olarak nasıl elde edileceği açıkça tanımlanmalıdır.
- Pascal, C, ADA imperatif dillerin örnekleridir.

Deklaratif Diller

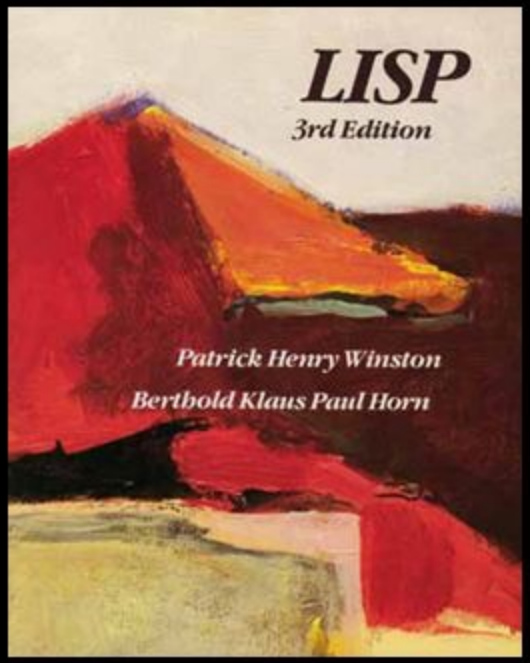
- Dekleratif diller programcıya programın mantığını, yani ulaşmak istediği hedef üzerine odaklanma olanağı sağlar. Bu diller nelerin hesaplanacağını belirleyen tanımlar veya denklemlerden oluşmuştur. Hesaplamanın nasıl yapılacağı ise programcı tarafından belirlenen bir husus değildir.

- PROLOG dekleratif dillere örnektir.

https://github.com/themanoftalent

Fonksiyonel Diller

- Fonksiyonel dillerde terimler komutlar yerine fonksiyonlardan oluşmuştur. Bu yaklaşım programcıları komutları belli bir sırada dizme zorunluluğundan kurtarıp onlara daha kavramsal düşünebilme olanağı sağlar.
- LISP



https://github.com/themanoftalent

Nesneye Dayalı Diller

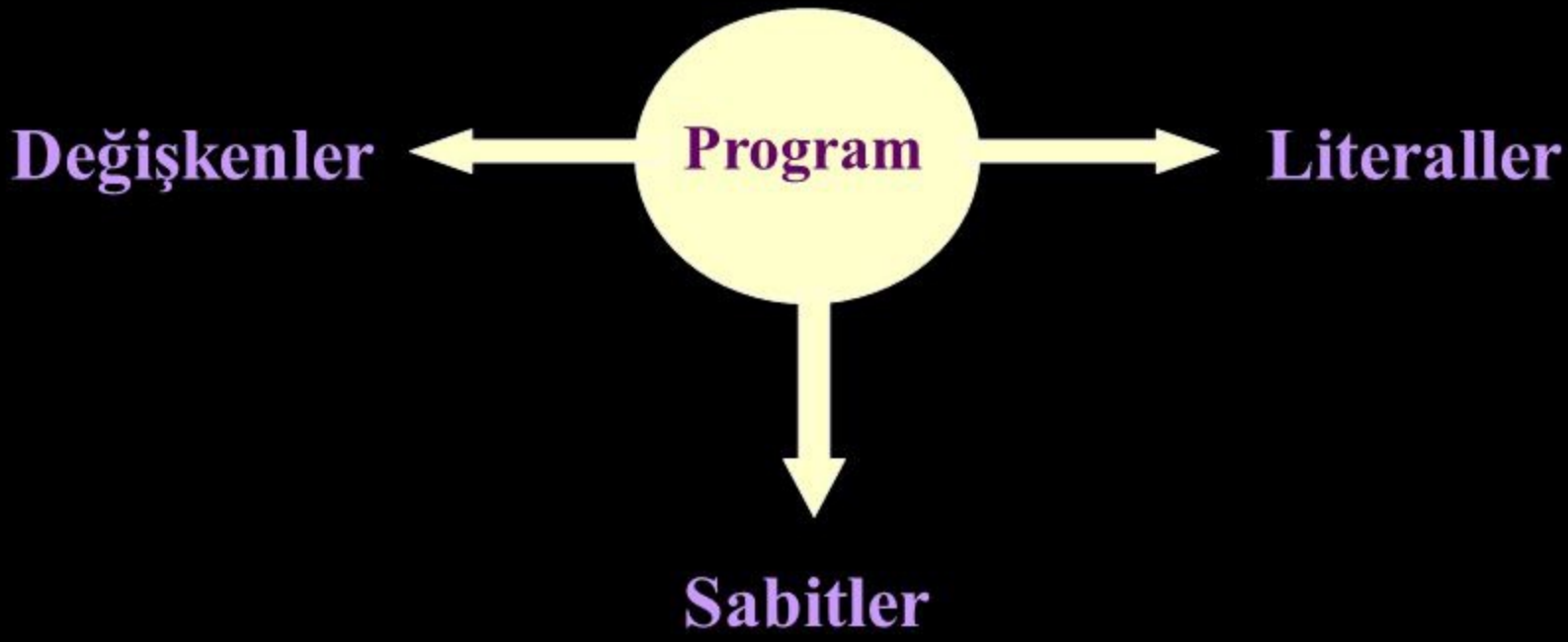
- Nesneye dayalı dillerde veriler, komutların veya fonksiyonların üzerlerinde işlem yaptığı pasif elemanlar olmaktan çıkıp, çevreleri ile ilişki kurabilen nesneler olarak tanımlanırlar.

- C++ en yaygın bilinen nesneye dayalı dildir.

https://github.com/themanoftalent

3-PROGRAMLAMA KAVRAMLARI

3-1- Programlarda Kullanılan Araçlar



https://github.com/themanoftalent

Değişkenler:

- Program içinde programcı tarafından tanımlanan nesnelere verilen isimdir.

- Pascal kodları kullanarak

```
var N: integer;
N:= 35
```

şeklinde değişken tanımlayabiliriz

Sabitler

- Programın bir yerinde bir değer atanan ve programın çalışması süresince aynı değerde kalan nesnelerdir.

- Örneğin;

  constant *M =50;*

Literaller

- Bir değerin karakter olarak gösterimidir. M sabitine 50 değeri atamıştık ve'50' bu değerin literal olarak karşılığıdır. Literallerin kullanılması programcılar arasında zaman zaman kötü bir programlama yaklaşımı olarak nitelendirilmektedir.

3-2- Veri Tipleri:

- Programlama dilleri çeşitli işleri yapabilmek için kullanılacak veri türlerini ve bu türler üzerinde yapılabilecek işlemleri tanımlar. En yaygın olarak kullanılan veri türleri ;

  - Tamsayılar (integer),
  - Gerçek sayılar (real) ve
  - Kayan noktalı sayılar (floating point) olarak sayılabilir.

https://github.com/themanoftalent

JAVA programlama dilinde kullanılan veri türleri

Değişken Türü	Türkçe karşılığı	Bit büyüklüğü	Sınır Değerleri
boolean	Mantık değişkeni	1	True / false
byte	Tamsayı değişkeni	8	-128 …+127
float	Gerçeksayı değişkeni	32	-3.40292347e+38 ‘den 3.40292347e+38 e

https://github.com/themanoftalent

3-3- Veri Yapıları:

- Veri yapısı, verinin hafizada saklanma şekli ve organizasyonu olarak tanımlanabilir. Veriler, veri tiplerine göre hafizada farklı şekillerde tutulabilirler.

- Veriye farklı yollardan ulaşma isteği farklı organizasyonların yapılmasını beraberinde getirmektedir.

https://github.com/themanoftalent

Örneğin hafızada her biri 4 byte yer kaplayan 100 adet tam sayı

- sırasız
- sıralı (örn küçükten büyüğe doğru)
- ağaç yapısı

olmak üzere üç farklı şekilde organize edilmiş olabilir.

- Ortalamanın bulunması gerekiyorsa; bilgilerin sıralı olması önemli değildir.
- Arama yapılmak istendiğinde; verilerin sıralı olması gereklidir.
- Eğer mevcut verilere ekleme veya çıkartma yapılmak istenirse; ağaç yapısının kullanılması daha uygundur.

Görüldüğü gibi veriye ulaşma şekline göre organizasyon değişmektedir.

https://github.com/themanoftalent

3-4 Atama İfadeleri:

- Atama ifadeleri, bir değişkene bir değer atamak için kullanılan ifadelerdir. En yaygın olarak kullanılan programlama araçlarıdır. Özellikle *imperatif ve nesne tabanlı* dillerde, atama ifadeleri çok yaygın olarak kullanılır.
- Genel yapı;

değişken adı <atama ifadesi> (değer veya işlem)

https://github.com/themanoftalent

Atama ifadesi değişik dillerde değişik şekilde olabilir.

- Pascal'da atama ifadesi "**:=**"
- C 'de atama ifadesi "**=**"

3-5- Kontrol İfadeleri

- Kontrol ifadeleri programın akışını etkileyen ve normal akış sırasını değiştirebilen ifadelerdir.
- Bu ifadeler kullanılarak program değişik noktalara aktarılabilir.
- *Go to* ve *if then else* örnek verilebilir.

https://github.com/themanoftalent

```
Var
not1,not2:integer
begin
write('yüzlük sistemde bir sayı giriniz');
readln(not1);
if not1>84 then not2:=5
else  if not1>69 then not2:=4
        else  if not1>54 then not2:=3
                else if not1>44 then not2:)=2
                        else if not1>24 then not2:=1
                                else not2:=0;
writelen('Notun % lik sistemdeki karşılığı = ' not2);
end.
```

https://github.com/themanoftalent