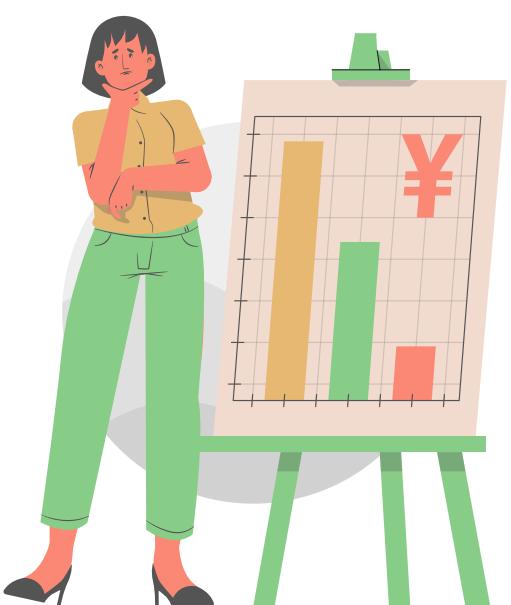


26 Mayıs 2022

# SİSTEM ANALİZİ VE TASARIMI - BIL206

Öğr. Gör. Buse Yaren TEKİN





# İçerikler

- UML
- Nesne Tabanlı Modelleme
- Sınıf
- Miras
- Çok İşlevlilik
- Diyagramlar
- Kapsülleme

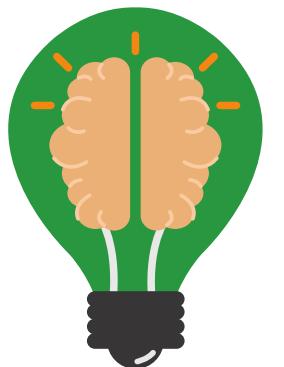


# Nesne Modellemeye İlişkin Temel Kavramlar

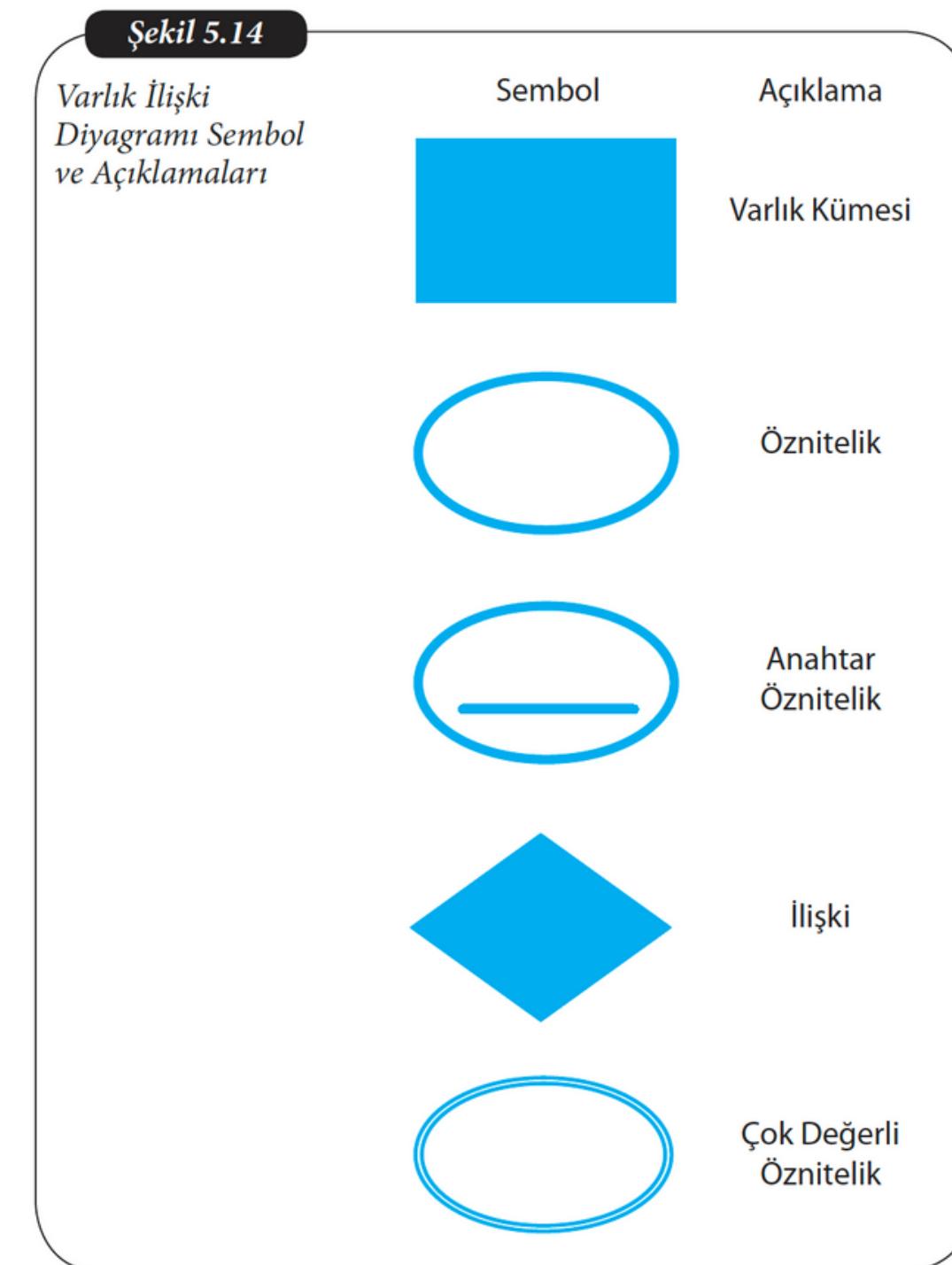
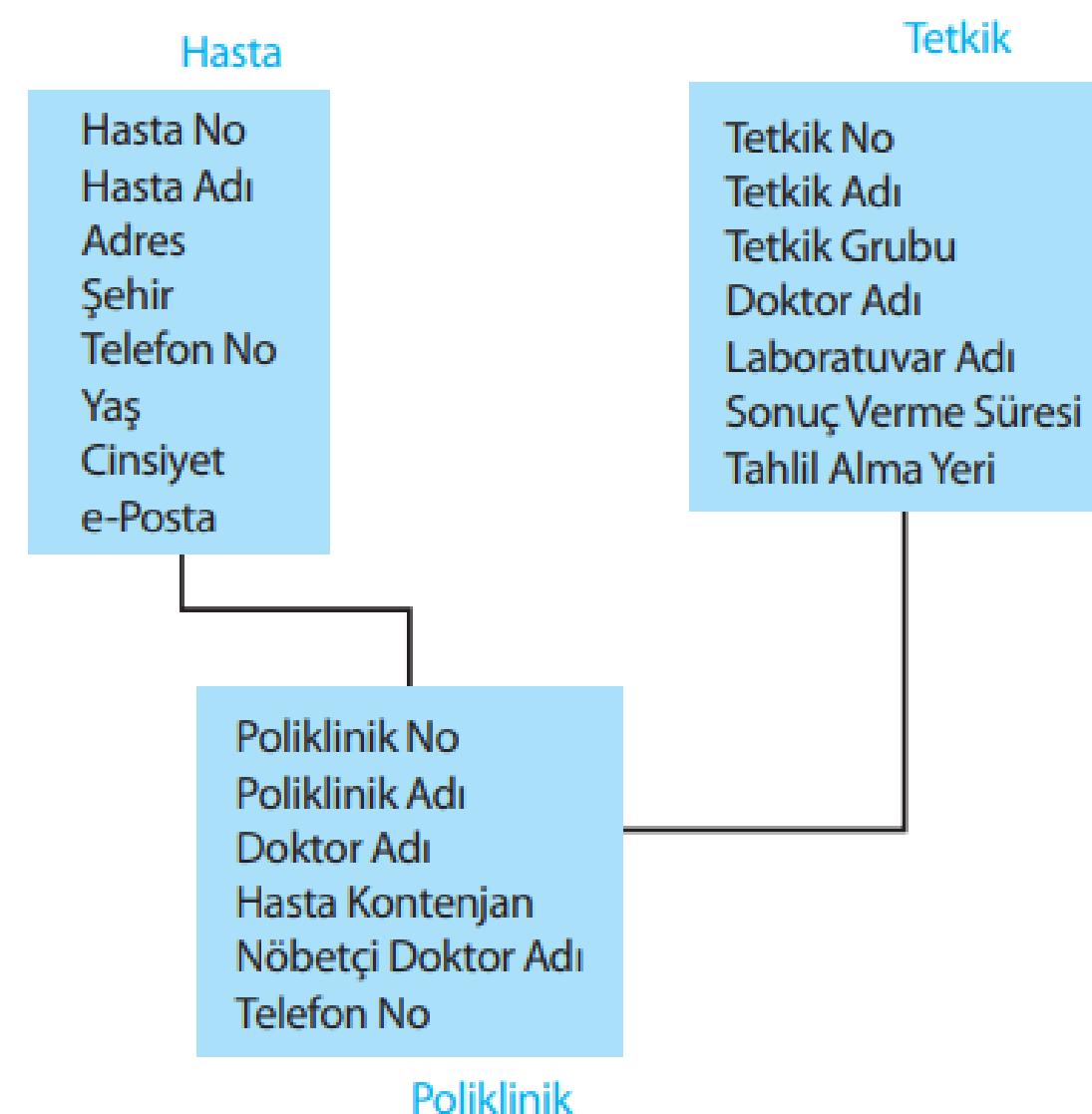
Bölüm 1

# Varlık İlişki Diyagramları

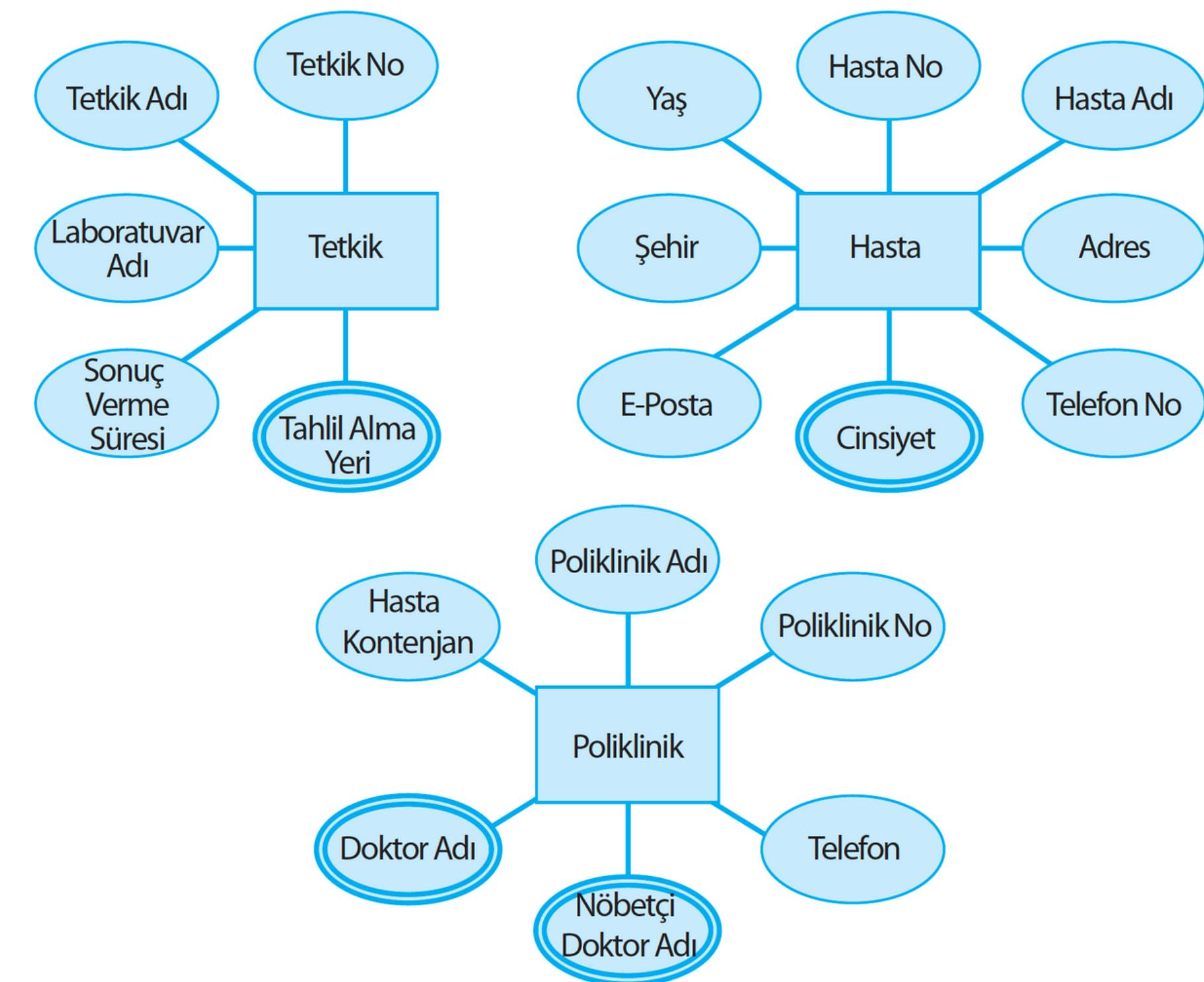
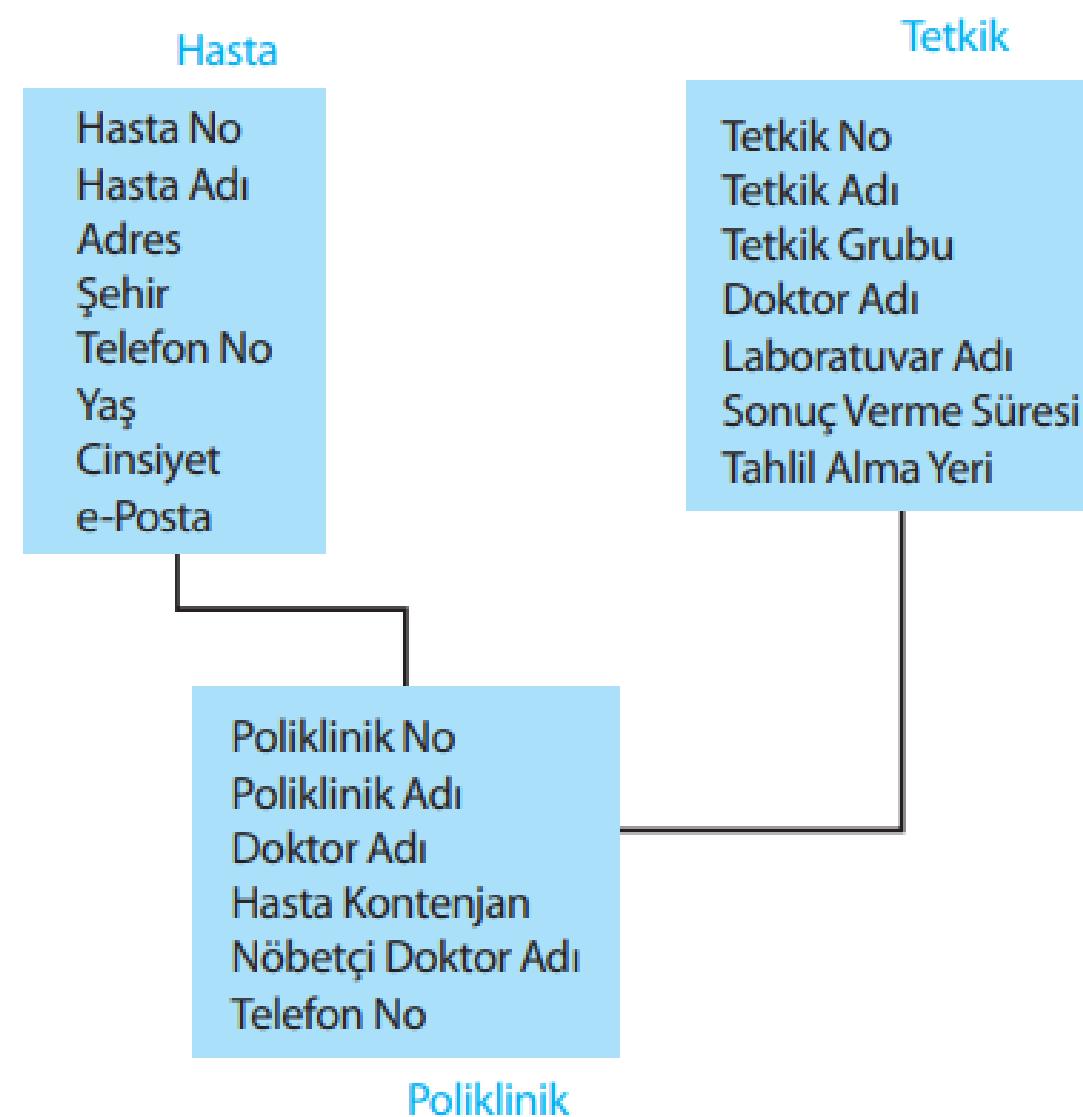
**Çok Değerli Öznitelik:** Bir aktörün cinsiyet özniteliği kadın veya erkek olmak üzere birden fazla değerden birini alabilecektir. Bu şekilde bir kategori ayrımına gidilen sistemlerde hasta varlık kümесinin cinsiyet özniteliği çok değerli bir özniteliktir.



# Varlık İlişki Diyagramı Ödevi



# Varlık İlişki Diyagramı Ödevi



# UML ile Nesne Yönelimli Analiz ve Modelleme

Nesne tabanlı programlama dillerinin kullanımı giderek yaygınlaşmaktadır. Bunun nedeni, nesne tabanlı programlama dillerinin, **kodların yeniden kullanımına olanak sağlaması ve programlama maliyetlerini azaltmasıdır**. Ayrıca, nesne tabanlı programlama yaklaşımı sayesinde, farklı programlama takımlarının bir yazılımın ortaya çıkmasında iş birliği içinde çalışması mümkün olmaktadır. Bu programlama grupları coğrafi olarak farklı konumlarda ikamet edebilmektedir. Her programlama grubu, belirlenmiş bir arayüze bağlı kalarak, bir ya da daha çok nesnenin gerçekleştirilmesi için gereken programlama kod parçalarını bağımsız olarak geliştirmek ile sorumlu olabilmektedir.

# UML ile Nesne Yönelimli Analiz ve Modelleme

Programlamada nesne tabanlı yaklaşım, nesne tabanlı analiz (NTA) ve nesne tabanlı tasarım (NTT) için birtakım tekniklerin kullanılmasını gerektirir. **Bu tekniklerde nesne tabanlı diyagramlar kullanılır.** Nesne tabanlı analiz ve tasarım için geliştirilen diğer diyagramlar, herhangi bir ortam içinde kullanılabilir. Örneğin, kullanım durum (use case) diyagramları, nesne tabanlı ya da geleneksel yapısal analizler için kullanılabilir.



# Nesne Modellemeye İlişkin Temel Kavramlar

Nesne tabanlı analizi gerçekleştirmek için, bazı kavramları bilmek gereklidir. Bu kavramlar, sistemler ve sistem geliştirme süreçleri üzerinde yeni bir düşünme yolunu gerektirir. Geleneksel yöntemlerde yaptığı gibi veri ve veri üzerinde işlem modelleri tanımlamak yerine; nesne tabanlı analiz statik veri yapıları ve dinamik davranış modelleri oluşturmaktadır. Nesne modellemeye ilişkin temel kavramlar izleyen kesimde verilmiştir.



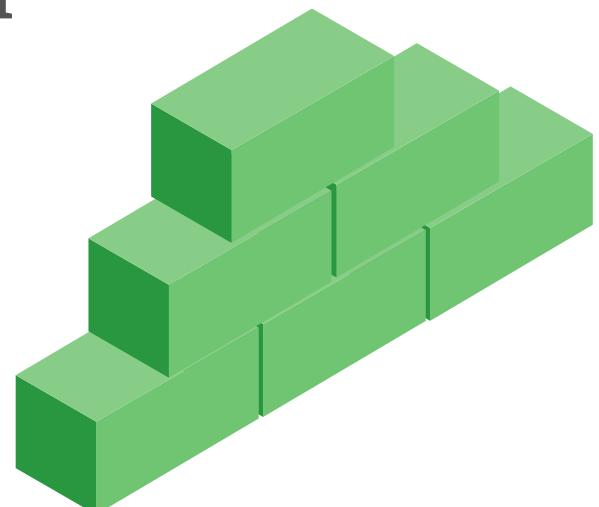
# Nesneler, Öznitelikler, Davranışlar ve Kapsülleme

Sistem geliştirmede nesne tabanlı yaklaşım, sistem ortamının nesnelerden oluşması üzerine kuruludur. Yaşadığımız dünya nesnelerden oluşmaktadır. Gerçekten de çevremizde, kapılar, masalar, dolaplar, ağaçlar gibi nesneler ve her nesnenin kendine özgü nitelikleri ve işlevleri bulunmaktadır. Ayrıca, nesneler birbirleri ile etkileşim içindedir.



# Nesneler, Öznitelikler, Davranışlar ve Kapsülleme

Nesneler, sadece görülebilen ve dokunulabilen somut şeyler olmayabilir. Örneğin, arkadaşınızdan telefon çağrısı bekliyorsunuz. Gelen çağrı, görülemez ve dokunulamaz. Ancak, algılayabildiğimiz telefon çağrısı soyut bir nesnedir. Buna göre, içinde bulunduğuumuz çevre somut ya da soyut nesnelerden oluşmaktadır. Sistem geliştirmede nesne tabanlı yaklaşım kullanıldığında, sistemi oluşturacak nesnelerin belirlenmesi önemlidir.



# Nesneler, Öznitelikler, Davranışlar ve Kapsülleme

Her nesne, öznitelik olarak adlandırılan ve sahip olduğu özellikleri belirten verilere sahiptir. Örneğin, “müşteri” olarak tanımlanan bir nesne, müşteri numarası, isim, soyisim, ev adresi, iş adresi, ev telefon numarası, iş telefon numarası gibi özniteliklere sahip olabilir.

Ali ve Mehmet iki hasta nesnesidir. Her iki nesne aynı tipte özniteliklere sahip olmalarına karşın özniteliklerin aldığı değerler farklıdır.

## Ali: Hasta

soyad = Can  
ad = Ali  
adres = Batıkent Mah  
telefon = 532-777 1111  
doğum tarihi = 01/01/76  
sigorta kurumu = SG

## Mehmet: Hasta

soyad = Yılmaz  
ad = Mehmet  
adres = Şirintepe Mah  
telefon = 533-123 1111  
doğum tarihi = 25/05/90  
sigorta kurumu = SGK

# Nesneler, Öznitelikler, Davranışlar ve Kapsülleme

Her nesne, davranışlara sahiptir. Bu davranışlar, nesnenin neler yapabileceğini belirtir. Örneğin, bir nesne olarak kapı ile eşleştirilen davranışlar vardır. Kapı açılabilir, kapanabilir, kilitlenebilir ve kilidi açılabilir. Tüm davranışlar, kapı ile ilişkili davranışlardır. Başka bir nesne ile ilgisi yoktur. **Nesne tabanlı programlamada, davranışlar fonksiyonlar olarak oluşturulur.**

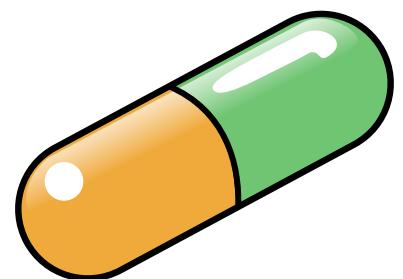
**Davranışlar (behaviors),**  
bir nesnenin yapabildiği  
şeyleri belirtir ve nesnenin  
verileri üzerinde işlem yürüten  
fonksiyonlar olarak gerçekleşir.

**Fonksiyon,** bilgisayar  
programdaki belirli bir  
işlevi yerine getirmek üzere  
oluşturulmuş alt programdır.

# Nesneler, Öznitelikler, Davranışlar ve Kapsülleme

Kapsülleme kavramı, nesne tabanlı sistemler ile ilişkilidir. Kapsülleme, nesneleri, nesnelerin özniteliklerini belirten verileri ve kendi verileri üzerinde işlem gerçekleştiren fonksiyonları bünyesinde barındırır.

**Kapsülleme (encapsulation),** verilerin ve bu veriler ile ilişkili işlemlerin bir varlık içerisinde tutulmasıdır.



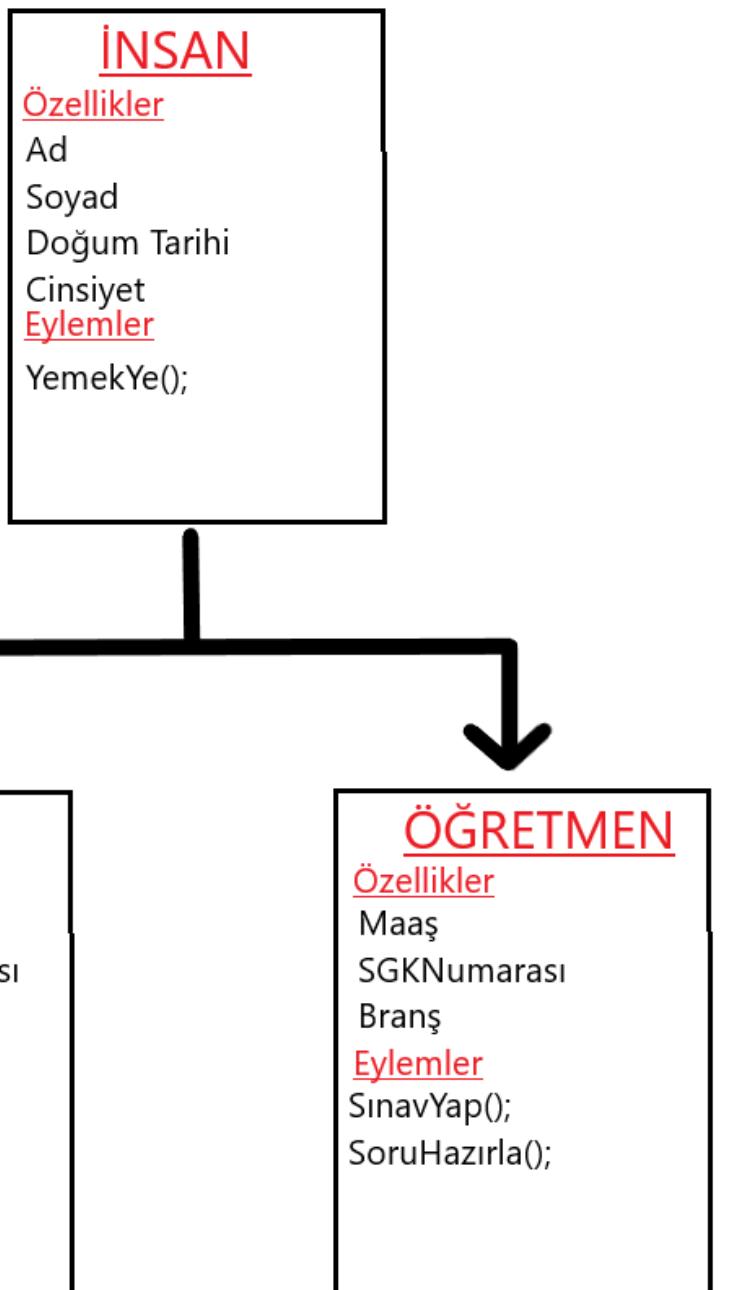
# Nesneler, Öznitelikler, Davranışlar ve Kapsülleme

Kapsülemeyi kısaca tanımlarsak bir nesnenin iç yapısını (verilerini ve özelliklerini) dış dünyadan doğrudan erişime kapatılması anlamına gelir. Bu sayede nesneye ait veriler değer ataması yapılırken yanlış kullanımdan korunmuş olur.



# Sınıflar, Genelleşme ve Özelleşme

Nesne modellemede bir başka önemli kavram, nesnelerin gruplandırılması için kullanılan sınıflardır. Benzer öznitelik tiplerine ve davranışlara sahip nesneler, sınıf olarak tanımlanır. Belirli bir sınıfa ait nesne örneklediğinde, o sınıf için önceden tanımlanan özniteliklere ve davranışlara sahip bir nesne var olur. **Sınıflar sayesinde programlar parçalara bölünür ve karmaşıklığı azalır.**



# Sınıflar, Genelleşme ve Özelleşme

Nesne tabanlı programlamada örnek bir iş akışı;

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace _01_Sınıflar
{
    public class Personel
    {
        public string Isim;
        public string Soyisim;
        public int Yas;
        public int Maas;
    }
}
```

**Projemizin İsmi**

**Sınıfımızın İsmi**

# Sınıflar, Genelleşme ve Özelleşme

Şekilde sınıf ve nesneler gösterilmektedir. “Hasta” sınıfı, bir hasta nesnesinin sahip olması gereken öznitelikleri ve fonksiyonları tanımlamaktadır. “Hasta” sınıfından Ali ve Mehmet olarak örneklenen nesneler, aynı tip özniteliklere ve fonksiyonlara sahiptir. Ancak, her nesnenin öznitelik değerleri kendine özgüdür.

## Hasta

- soyad
- ad
- adres
- telefon
- doğum tarihi
- sigorta kurumu

- 
- + randevu al ()
  - + teşhis koy ()
  - + tıbbi geçmiş sorula ()

## Ali: Hasta

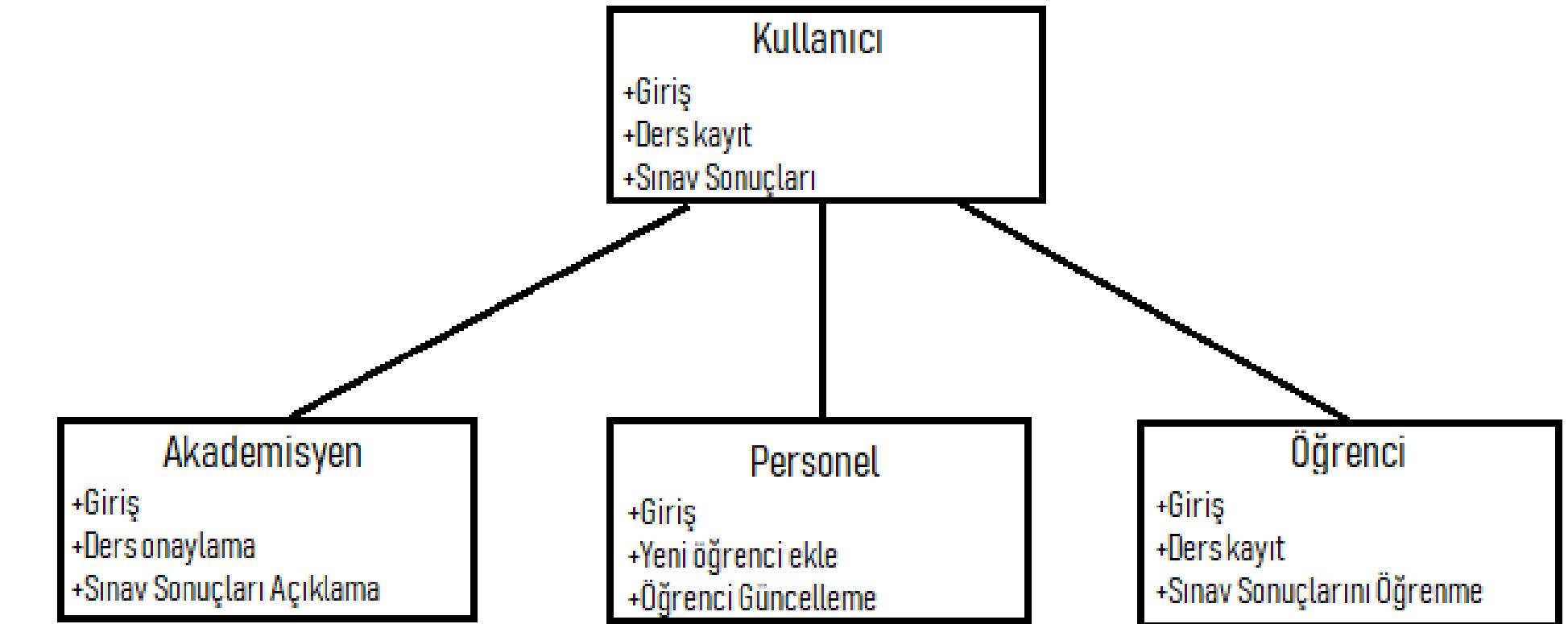
- soyad = Can
- ad = Ali
- adres = Batıkent Mah
- telefon = 532-777 1111
- doğum tarihi = 01/01/76
- sigorta kurumu = SGK

## Mehmet: Hasta

- soyad = Yılmaz
- ad = Mehmet
- adres = Şirintepe Mah
- telefon = 533-123 1111
- doğum tarihi = 25/05/90
- sigorta kurumu = SGK

# Sınıflar, Genelleşme ve Özelleşme

Sınıflar, üst sınıf ve alt sınıf olarak düzenlenebilmektedir. Bu düzenleme içinde, üst ve alt sınıflar arası ilişki miras (**inheritance**) ilişkisi olarak tanımlanmaktadır. Miras kullanılarak tasarlanan sınıfların ortak veri ve fonksiyonları üst sınıf altında toplanmaktadır. Bu teknik, genelleşme olarak tanımlanabilir.



# Sınıflar, Genelleşme ve Özelleşme

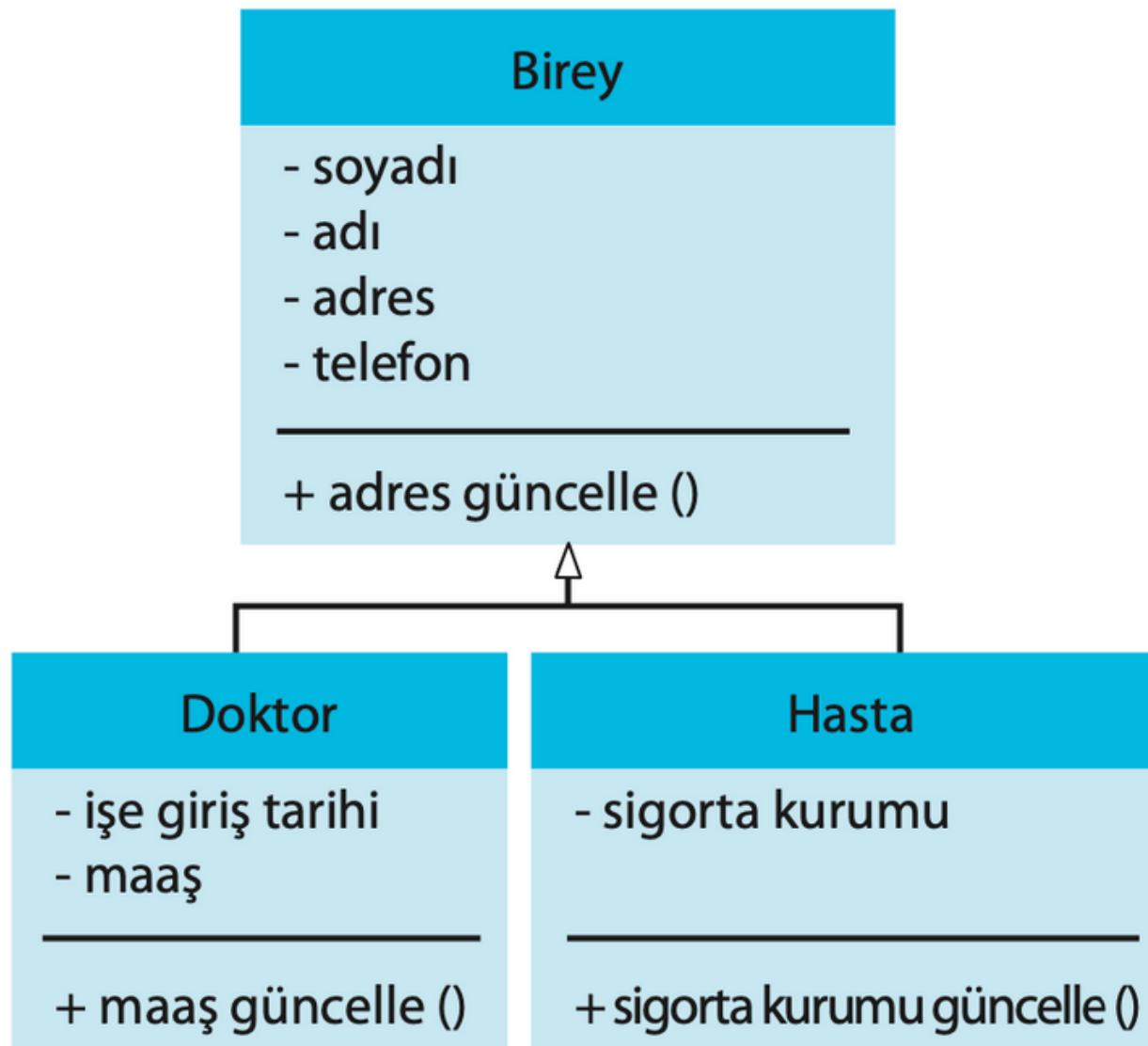
Üst sınıfındaki veri ve fonksiyonları kullanmak üzere oluşturulan sınıflar alt sınıf olarak adlandırılır. Alt sınıflar, üst sınıfın kullandıkları veri ve fonksiyonlara ilave olarak yeni veri ve fonksiyonlar da içerir. Buna da **özelleşme** denir. Örneğin, "insan" sınıfı, "öğrenci" ve "öğretmen" sınıfları için bir üst sınıfı oluşturur. "öğrenci" ve "öğretmen" sınıfları, "insan" sınıfının sahip olduğu veri ve fonksiyonları kapsar. Ayrıca, bu sınıflar ilave veriler ve fonksiyonlar içерerek özelleşirler.

# Sınıflar, Genelleşme ve Özelleşme

Miras İlişkisi Kullanmadan

Doktor	Hasta
- soyadı - adı - adres - telefon - işe giriş tarihi - maaş	- soyadı - adı - adres - telefon - sigorta kurumu
+ adres güncelle () + maaş güncelle ()	+ adres güncelle () + sigorta kurumu güncelle ()

Miras İlişkisi Kullanarak



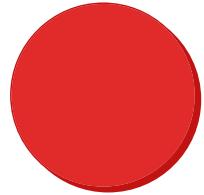
# Kendimizi Sınayalım

1. Aşağıdakilerden hangisi, nesne tabanlı programmanın avantajlarından biri **değildir**?

- a. Kodların yeniden kullanımına olanak sağlaması
- b. Programlama maliyetlerini azaltması
- c. Ayrı yazılım gruplarının coğrafi olarak farklı alanlarda bulunabilmesi
- d. Farklı programcılar aynı yazılımın kod parçalarını bağımsız yazabilmesi
- e. Küçük ölçekli yazılımların geliştirilme sürecini hızlandırması

# Kendimizi Sınayalım

1. Aşağıdakilerden hangisi, nesne tabanlı programmanın avantajlarından biri **değildir**?
- a. Kodların yeniden kullanımına olanak sağlama
  - b. Programlama maliyetlerini azaltma
  - c. Ayrı yazılım gruplarının coğrafi olarak farklı alanlarda bulunabilmesi
  - d. Farklı programcılar aynı yazılımın kod parçalarını bağımsız yazabilmesi

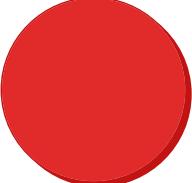


Küçük ölçekli yazılımların geliştirilme sürecini hızlandırması

# Kendimizi Sınayalım

- 2.** Nesne kavramı ile ilgili olarak aşağıda verilen ifadelerden hangisi yanlıştır?
- a. Özniteliklere sahip olabilir.
  - b. Sadece somut varlıklar olabilir.
  - c. Davranışlara sahip olabilir.
  - d. Diğer nesneler ile ilişkisi olabilir.
  - e. Başka nesnelere mesaj gönderebilir.

# Kendimizi Sınayalım

- 2.** Nesne kavramı ile ilgili olarak aşağıda verilen ifadelerden hangisi yanlıştır?
- a. Özniteliklere sahip olabilir.
  - b. Sadece somut varlıklar olabilir.
  - c. Davranışlara sahip olabilir.
  -  d. Diğer nesneler ile ilişkisi olabilir.
  - e. Başka nesnelere mesaj gönderebilir.

# Kendimizi Sınayalım

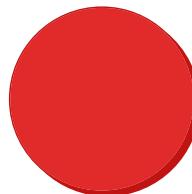
**3.** Sınıflar arası tanımlanan ilişkiler için aşağıdakilerden hangisi doğrudur?

- a. Her sınıf, sadece bir sınıfın tek bir nesnesi ile ilişkili olabilir.
- b. Toplanma ilişkisinde, bütün yok olduğunda parça da yok olur.
- c. Bileşim ilişkisinde, bütün olmadan da parça var olmaya devam edebilir.
- d. Miras ilişkisinde, alt sınıf üst sınıfın üye verileri ve fonksiyonlarına da sahiptir.
- e. Çok biçimlilik kullanımında, miras hiyerarşisini kullanmak gerekmez.

# Kendimizi Sınayalım

**3.** Sınıflar arası tanımlanan ilişkiler için aşağıdakilerden hangisi doğrudur?

- a. Her sınıf, sadece bir sınıfın tek bir nesnesi ile ilişkili olabilir.
- b. Toplanma ilişkisinde, bütün yok olduğunda parça da yok olur.
- c. Bileşim ilişkisinde, bütün olmadan da parça var olmaya devam edebilir.



Miras ilişkisinde, alt sınıf üst sınıfın üye verileri ve fonksiyonlarına da sahiptir.

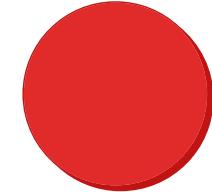
- e. Çok biçimlilik kullanımında, miras hiyerarşisini kullanmak gerekmek.

# Kendimizi Sınayalım

- 5.** UML diyagramları ile ilgili olarak aşağıda verilen ifadelerden hangisi **yanlıştır**?
- a. UML diyagramları, sadece nesne tabanlı programlama için kullanılabilir.
  - b. UML diyagramları sistem modellemesinde kullanılır.
  - c. UML diyagramları, analizden tasarıma kadar kullanabilecek kadar zengindir.
  - d. UML diyagramları, iki grupta toplamak mümkündür.
  - e. Yapısal diyagramlar, bilgi sistemindeki verileri ve static ilişkileri temsil eder.

# Kendimizi Sınayalım

**5.** UML diyagramları ile ilgili olarak aşağıda verilen ifadelerden hangisi **yanlıştır**?



a. UML diyagramları, sadece nesne tabanlı programlama için kullanılabilir.

b. UML diyagramları sistem modellemesinde kullanılır.

c. UML diyagramları, analizden tasarıma kadar kullanabilecek kadar zengindir.

d. UML diyagramları, iki grupta toplamak mümkündür.

e. Yapısal diyagramlar, bilgi sistemindeki verileri ve static ilişkileri temsil eder.

# Kaynaklar

1. Öğr. Gör. Aslı Birol, Sistem Analizi ve Tasarımı, BIL3403 Ders Notları
2. Anadolu Üniversitesi, BIL206, Sistem Analizi ve Tasarımı Ders Notları
3. Medium, Kişisel Blog Yazısı.
4. Sistem Analizi ve Tasarımı Final Notları, Web Sitesi.