NESNEYE DAYALI PROGRAMLAMA VE C++

İstanbul Teknik Üniversitesi

©1999-2003 Dr. Feza BUZLUCA

GİRİŞ

Dersin Amacı:

- **Nesneye Dayalı Programlama** (*Object-Oriented Programming*) ve **Üretken Programlama** (*Generic Programming*) yöntemlerini tanıtmak.
- Bu yöntemlerin iyi bir program yazmak için etkin bir biçimde nasıl kullanılacağını C++ ile göstermek.

Neden C++:

- Kaliteli program yazmaya yönelik özellikleri
- Sahip olduğu program arşivleri
- Bu konuda yazılmış yüzlerce kitap ve dergi
- Yaygın kullanım

  Sistem Programları: İşletim sistemleri sürücü programları

  Ticari programlar: Bankacılık, sigorta uygulamaları

  Grafik programları

  Bilgisayar ağları üzerinde iletişim programları

©1999-2003 Dr. Feza BUZLUCA

C++ ile Programlama Öğrenirken Dikkat Edilmesi Gereken Noktalar:

Konuşma dillerinin gramer kurallarına benzer şekilde bilgisayar programlama dilleri de çeşitli yapısal kurallar içerirler.

Programlama dilinin teknik ayrıntılarını ve kurallarını iyi bilmek iyi program yazmak için yeterli değildir.

Bir programlama dili öğrenilirken dilin kuralları ve teknik ayrıntılarından önce o dilin desteklediği programlama yöntemi ve düşünce biçimi iyice anlaşılmalıdır.

Özellikle C++ konusunda daha zor olan ve daha çok zaman alan aşama programlama yönteminin öğrenilmesidir.

C++ öğrenirken ilk amacınız, program yazmak için yeni sentaks (gramer) kuralları öğrenmek değil, program geliştirmek için yeni ve daha etkin bir yöntem öğrenmek olmalıdır.

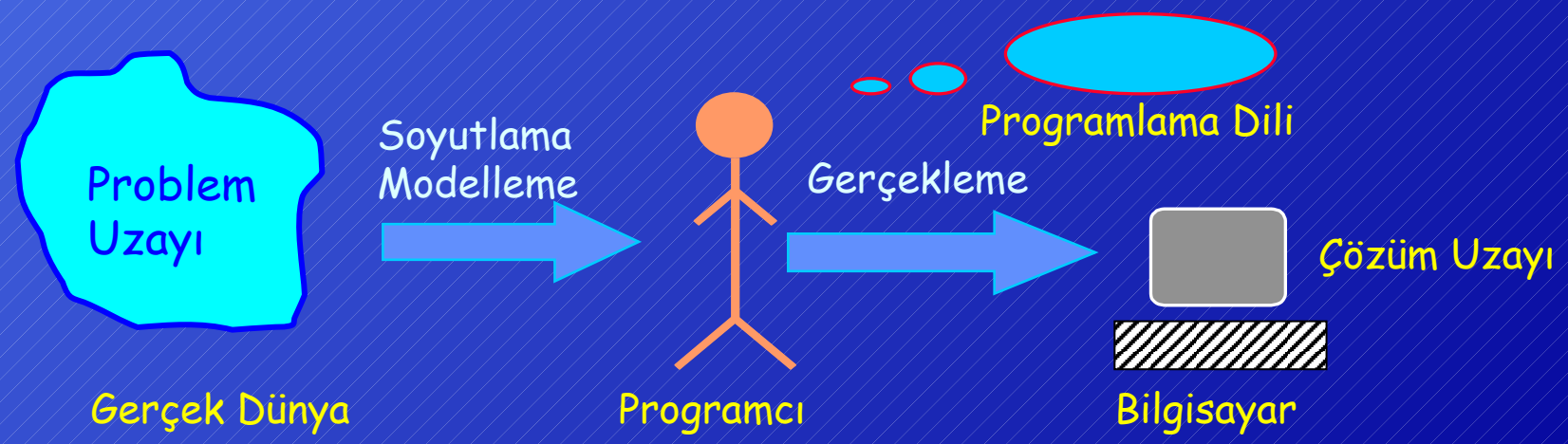
©1999-2003 Dr. Feza BUZLUCA

Programlama Nedir?

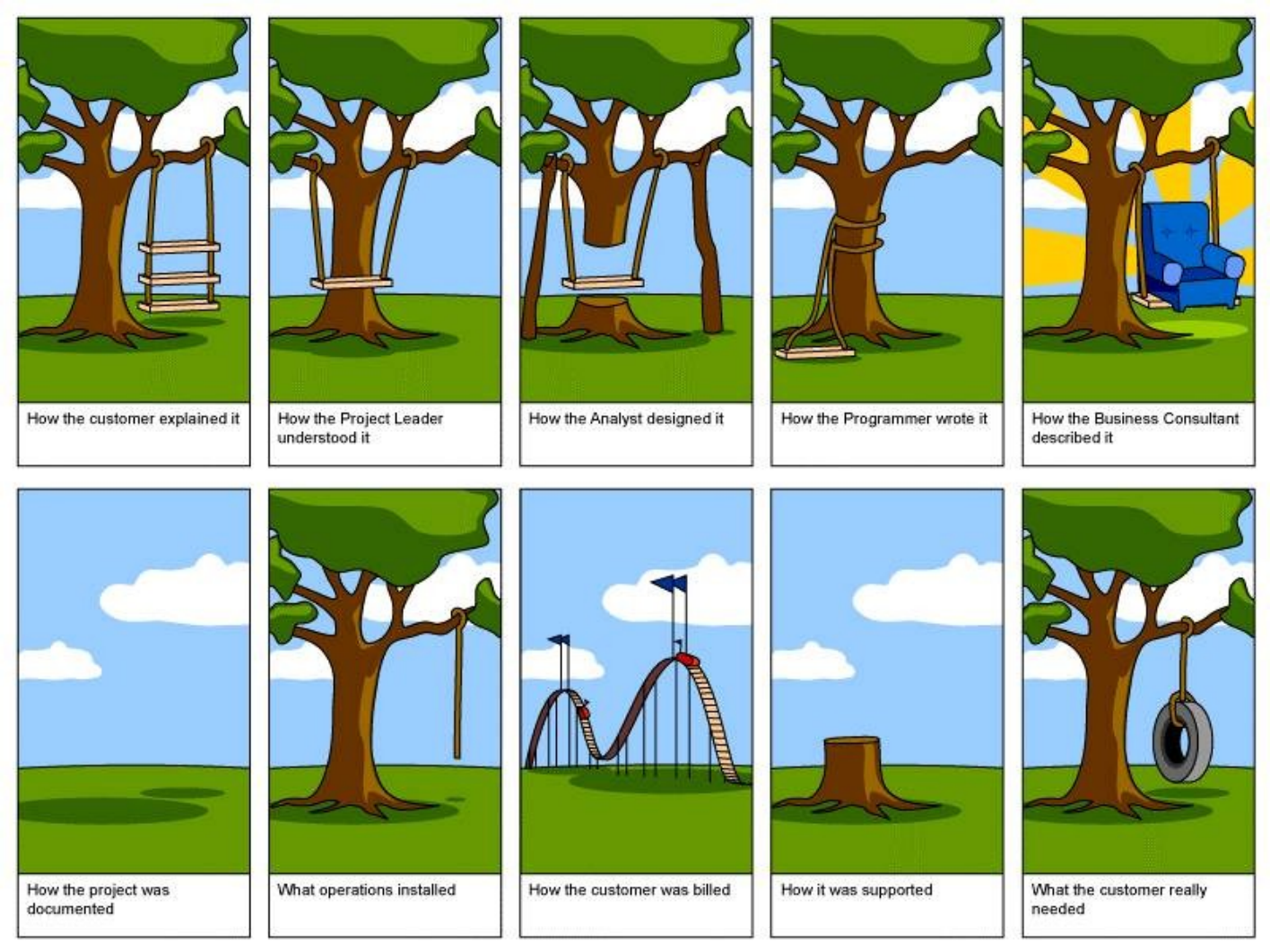
İnsanlar günlük hayatta kullandıkları konuşma dilleri ile çeşitli kavramları birbirlerine anlatmaya çalışırlar.
Benzer şekilde bilgisayar programcıları da çözülmesi gereken problemlerle ilgili kavram ve varlıkları, kullandıkları programlama dili ile bilgisayarda ifade etmeye çalışırlar.

Programlama, yaşadığımız gerçek dünyadaki problemlere ilişkin çözümlerin bilgisayarda ifade edilmesidir.

Bunu yapabilmek için, kodlamaya geçilmeden önce tasarım aşamasında, problemi oluşturan varlıkların bilgisayarda ifade edilebilecek şekilde modellerinin oluşturulması gerekmektedir.



©1999-2003 Dr. Feza BUZLUCA



Dr. Sanem Sarıel'in OOP notlarından alınmıştır.

©1999-2003 Dr. Feza BUZLUCA

Bir Yazılımın Kalitesi

- Bir yazılım istenen işi doğru olarak yapmalıdır.
- Program kolay kullanılabilir olmalıdır.
- Program uygulamanın gerektirdiği kadar hızlı çalışmalıdır.
- Program, sistem kaynaklarını (işlemci, bellek, disk, iletişim ağı vb.) gerektiğinden daha fazla kullanmamalıdır.
- Yazılım sağlam olmalıdır.
- Yazılım yeterli dokümantasyona (kullanım klavuzu) sahip olmalıdır.
- Yazılımı güncellemek kolay olmalıdır.

kullanıcı

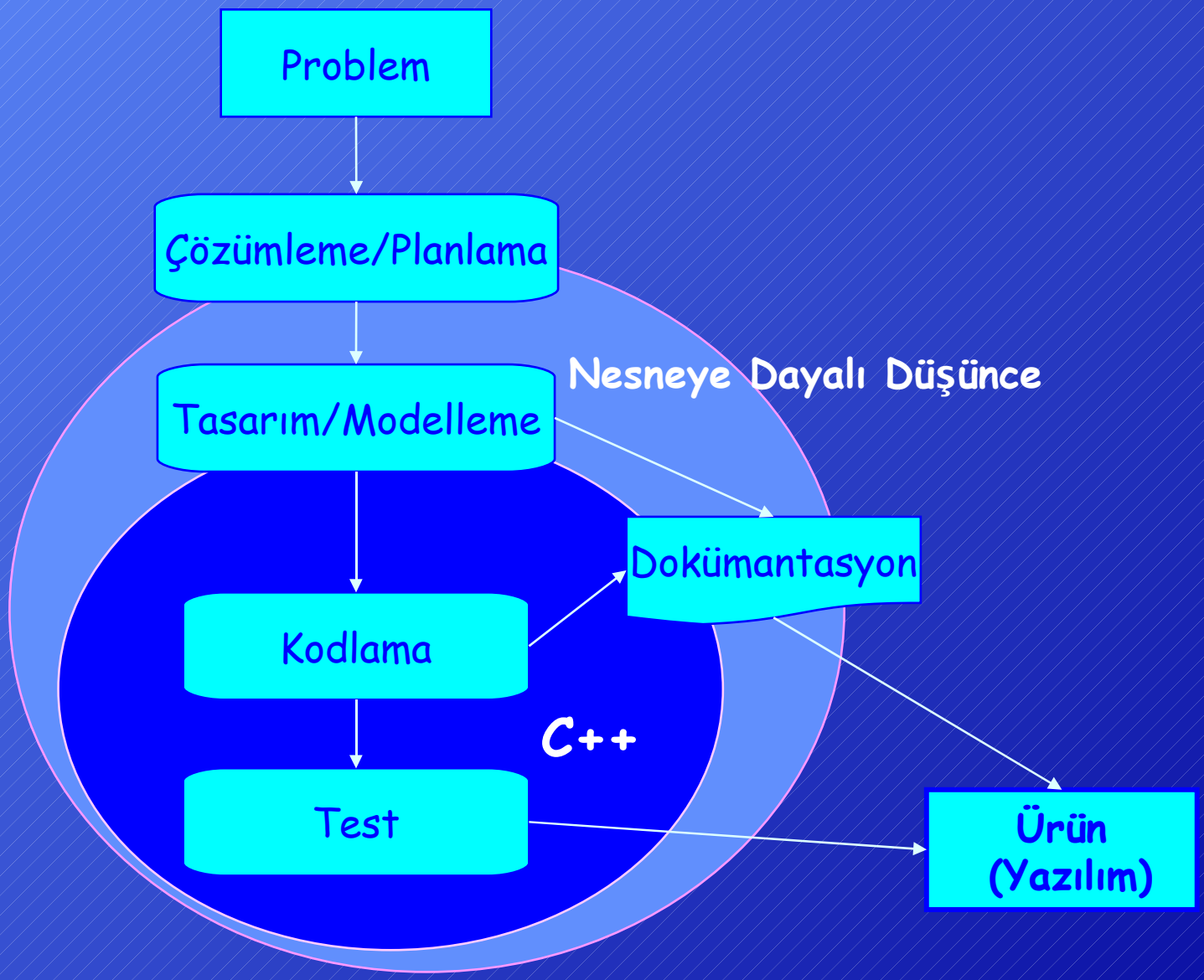
- Programın kaynak kodu kolay okunabilir ve anlaşılır olmalıdır.
- Programın yapısı bağımsız modüllerden oluşmalı, modüller arasındaki iletişim ve etkileşim mümkün olduğu kadar az olmalıdır.
- Yazılımın bakımı kolay olmalıdır.
- Yazılım projesi yeterli bir süre içinde tamamlanmalıdır.
- Yazılımın modülleri yeni projelerde tekrar kullanılabilmeli.
- Yazılımı geliştirme maliyeti düşük tutulabilmelidir.
- Yazılım, tasarım aşamalarını açıklayan dokümantasyona sahip olmalıdır.

Yazılım geliştiren

Nesneye dayalı programlama yukarıdaki maddelerde belirtilen kalite kriterlerini sağlamak üzere ortaya konmuş bir yöntemdir. Tasarım yapılırken ve bu tasarım C++ ile programa dönüştürülürken kalite kriterleri sürekli göz önünde bulundurulmalıdır.

©1999-2003 Dr. Feza BUZLUCA

Yazılım Geliştirme Aşamaları



©1999-2003 Dr. Feza BUZLUCA

Çözümleme (Analiz): Belirsizlik kalmayacak şekilde çözülmesi istenen problem anlaşılır ve yazılımın gerçekleştirilmesine yönelik hazırlıklar yapılır.

Tasarım: Bu aşamada amaç, gerçek dünyadaki problemin bilgisayarda temsil edilebilecek soyut bir modelinin oluşturulmasıdır. Çözümün hangi unsurlardan oluşacağı ve bu parçaların nasıl modelleneceği tamamiyle programlama yöntemine bağlıdır.

Tasarım sonrası ortaya çıkacak olan yazılımın kalitesini doğrudan etkilediğinden bu aşamada iyi ve doğru bir çözümün oluşturulması çok önemlidir. Bu nedenle kodlamaya geçilmeden önce kurulan modelin *sağlama*sının (*verification*) yapılması gerekir.

Kodlama: Tasarım aşamasında oluşturulan model, bir programlama dili ile bilgisayara bu aşamada aktarılır.

Dokümantasyon: Yazılım projesinin her aşamasında yapılan işleri açıklayan dokümanlar hazırlamak gereklidir.

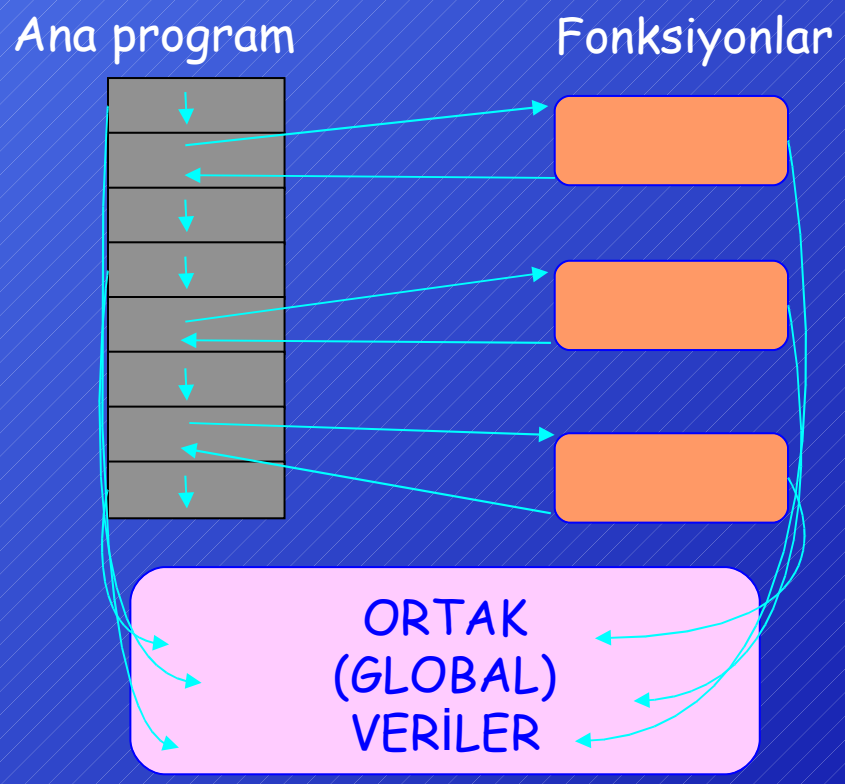
Sınama: En son aşamada programın olası giriş değerleri için nasıl davrandığı incelenir.

©1999-2003 Dr. Feza BUZLUCA

İşleve Dayalı (Procedural) Programlama Yöntemi

Basic, Fortran, Pascal, C gibi programlama dillerinin desteklediği bu yöntemde öncelikle gerçeklenmek istenen sistemin yapması gereken iş belirlenir.

Büyük boyutlu ve karmaşık işler, daha küçük ve basit işlevlere (fonksiyon) bölünerek gerçeklenirler.



©1999-2003 Dr. Feza BUZLUCA

İşleve Dayalı Programlama Yönteminin Değerlendirmesi

- "Böl ve yönet" prensibine dayanır. Amaç büyük programları küçük parçalara bölerek yazılım geliştirme işini kolaylaştırmaktır.
- Ancak yazılımların karmaşıklıkları sadece boyutlarından kaynaklanmaz. Küçük problemler de karmaşık olabilir.
- Gerçek dünyadaki sistemler sadece fonksiyonlardan oluşmaz. Sistemin gerçeğe yakın bir modelini bilgisayarda oluşturmak zordur.
- Tasarım aşamasında verilerin göz ardı edilip fonksiyonlara ağırlık verilmesi hatalar nedeniyle verilerin bozulma olasılığını arttırır.
- Programcılar kendi veri tiplerini yaratamazlar
- Programı güncellemek gerektiğinde, yeni öğeler eklemek ve eski fonksiyonları yeni eklenen unsurlar için de kullanmak zordur.

İşleve dayalı yöntemi de kullanarak kaliteli programlar yazmak mümkündür. Ancak nesneye dayalı yöntem kaliteli programların oluşturulması için programcılara daha çok olanak sağlamaktadır ve yukarıda açıklanan sakıncaları önleyecek mekanizmalara sahiptir.

©1999-2003 Dr. Feza BUZLUCA

Nesneye Dayalı (Object-Oriented) Programlama Yöntemi

- Gerçek dünya nesnelerden oluşmaktadır.
- Çözülmek istenen problemi oluşturan nesneler, gerçek dünyadaki yapılarına benzer bir şekilde bilgisayarda modellenmelidir.
- Nesnelerin yapıları iki bölümden oluşmaktadır: **1.** *Nitelikler* (özellikler ya da durum bilgileri), **2.** *Davranışlar* (yetenekler)

Tasarım yapılırken sistemin işlevi değil, sistemi oluşturan **veriler** esas alınır. Bu nedenle tasarım yapılırken sorulması gereken soru,
"*Bu sistem ne iş yapar?*" değil,
"***Bu sistem hangi nesnelerden oluşur?***" olmalıdır.

Hangi unsurların nesne olarak modellenebilir:

- İnsan kaynakları ile ilgili bir programda; memur, işçi, müdür, genel müdür.
- Grafik programında; nokta, çizgi, çember, silindir.
- Matematiksel işlemler yapan programda; karmaşık sayılar, matris.
- Kullanıcı arayüzü programında; pencere, menü, çerçeve.

©1999-2003 Dr. Feza BUZLUCA

Nesne Örneği: Grafik Programındaki nokta

Düzlemdeki bir noktanın özellikleri; x-y koordinatlarıdır.

Davranışları ise, noktanın düzlemde yer değiştirmesi, renginin değişmesi, ekranda görünmesi ve ekranda kaybolmasıdır.

Buna göre, örnek olarak düşünülen Nokta modeli şu bölümlerden oluşacaktır:

x ve y koordinatları için iki adet tamsayı değişken: **x** , **y**

Noktanın koordinatlarını değiştirerek düzlemde yer değiştirmesini sağlayan fonksiyon: **hareket** ,

Noktanın ekranda görünmesini sağlayan bir fonksiyon: **gorun** ,

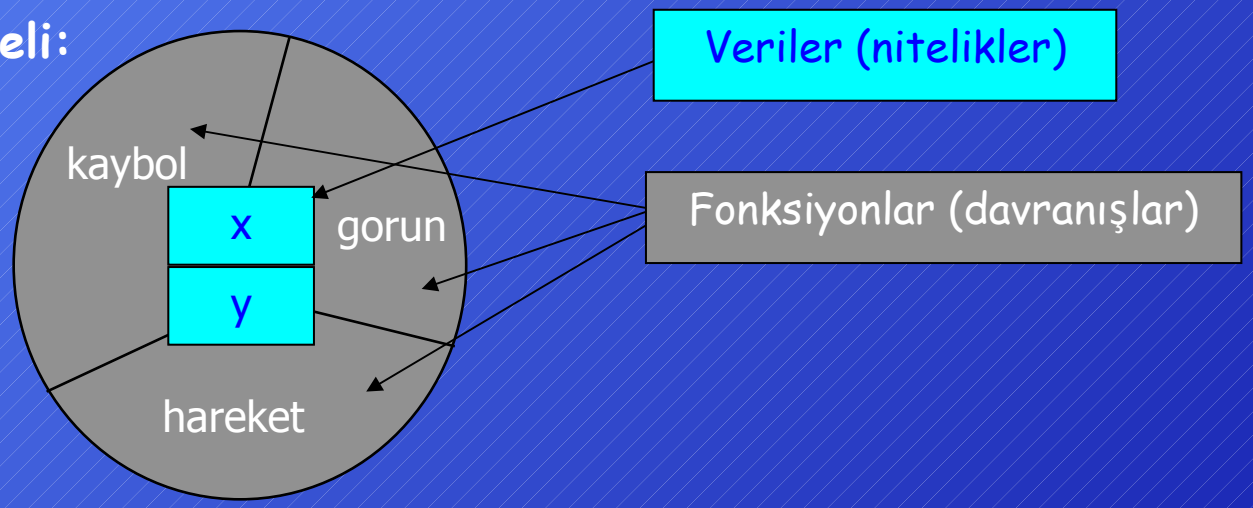
Noktanın ekrandan silinmesini sağlayan bir fonksiyon: **kaybol** .

Model bir defa oluşturulduktan sonra, ana programda bu modelden bir çok nesne yaratılabilir.

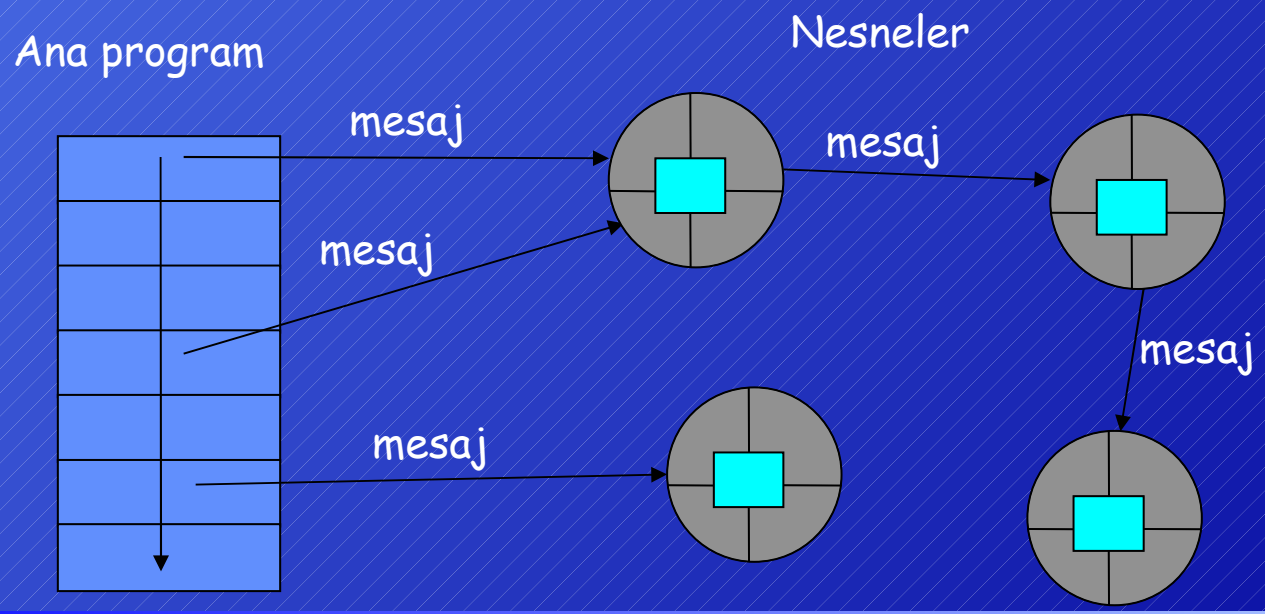
```
Nokta nokta1, nokta2, nokta3;
:
nokta1.hareket(50,30);
nokta1.gorun();
```


©1999-2003 Dr. Feza BUZLUCA

Bir Nesne Modeli:



Nesneye Dayalı Bir Programın Yapısı:



©1999-2003 Dr. Feza BUZLUCA

Yöntemin Değerlendirmesi:

- Gerçek dünya nesnelerden oluştuğundan bu yöntem ile sistemin daha gerçekçi bir modeli oluşturulabilir. Program daha anlaşılır olur.
- Nesne modellerinin içindeki veriler sadece üye fonksiyonların erişebileceği şekilde düzenlenebilirler. Veri saklama (*data hiding*) adı verilen bu özellik sayesinde verilerin herhangi bir fonksiyon tarafından bozulması önlenir.
- Programcılar kendi veri tiplerini yaratabilirler.
- Bir nesne modeli oluşturduktan sonra bu modeli çeşitli şekillerde defalarca kullanmak mümkündür (*reusability*).
- Programları güncellemek daha kolaydır.
- Nesneye dayalı yöntem takım çalışmaları için uygundur.

©1999-2003 Dr. Feza BUZLUCA