Giriş

Dr. Metin Özkan

İçerik

- Giriş (Introduction)
- C'den C++'a Geçiş (C++ as a better C)
- Sınıflar ve Nesneler (Classes and Objects)
- Operatör Yükleme (Operator Overloading)
- Miras, Kalıtım (Inheritance)
- Çok İşlev, Çok Biçimlilik (Polymorphism)
- Kural Dışı Durum Yönetimi (Exception Handling)
- Şablonlar (Templates)
- Standart Şablon Kütüphanesi (Standard Template Library, STL) {vectors and lists}

Amaç

- Nesne tabanlı programlamanın temel kavramlarını öğrenmek,
- Sınıf ve nesneleri geliştirebilmek,
- C++ programlama dilini kullanarak yüksek kalitede* yazılım geliştirmek için bu programlama yapılarının kullanımını öğrenmek,
  - **Yüksek kalitede yazılım*
    - *Hatalara karşı azartılmış duyarlılık: Bir nesne kendisine ait veriye erişimi kontrol eder. Böylece, bir nesne hatalı erişim teşebbüslerini reddedebilir.*
    - *Kolay tekrar kullanım: Nesneler kendilerini idame etmektedir ve bundan dolayı başka programların yapı blokları olarak kullanılabilirler.*
    - *Düşük bakım gereksinimleri: Bir nesne tipi, kullanıldıkları uygulamalarda bir değişiklik gerektirmeden kendi iç verilerini değiştirebilirler.*

Yazılım Kalitesi

- Yazılım kalitesinin 5 önemli özelliği;
  - *Kullanılabilirlik (Usability): Yüksek kullanılabilirliğe sahip bir yazılım ile, kullanıcılar için işlerini yapmak daha kolaydır. Acemiler için kolay öğrenme, uzmanlar için etkin kullanım.*
  - *Etkinlik (Efficiency): En etkin yazılım, işlemci zamanı, bellek, disk alanı, ağ bant genişliği ve diğer sistem kaynaklarını en az kullanandır.*
  - *Güvenilirlik (Reliability): En az hataya sahip yazılım en güvenilir yazılımdır. Sistemin, mevcut hataları da saptaması veya üstesinden gelebilmesi beklenir.*
  - *Bakım yapılabilirlik (Maintainability): Yazılımda değişiklik yapabilmenin kolay olması beklenir. Yazılım tasarımı, gelecekteki değişiklikler tahmin edilerek ve esneklik ekleyerek yapılmalıdır.*
  - *Tekrar kullanılabilirlik (Reusability): Yazılım bileşenleri, farklı yazılımlarda az ya da hiç değişiklik yapmadan kullanılabilmelidir.*

Yazılım Kalitesi (Software Quality) ve Paydaşları

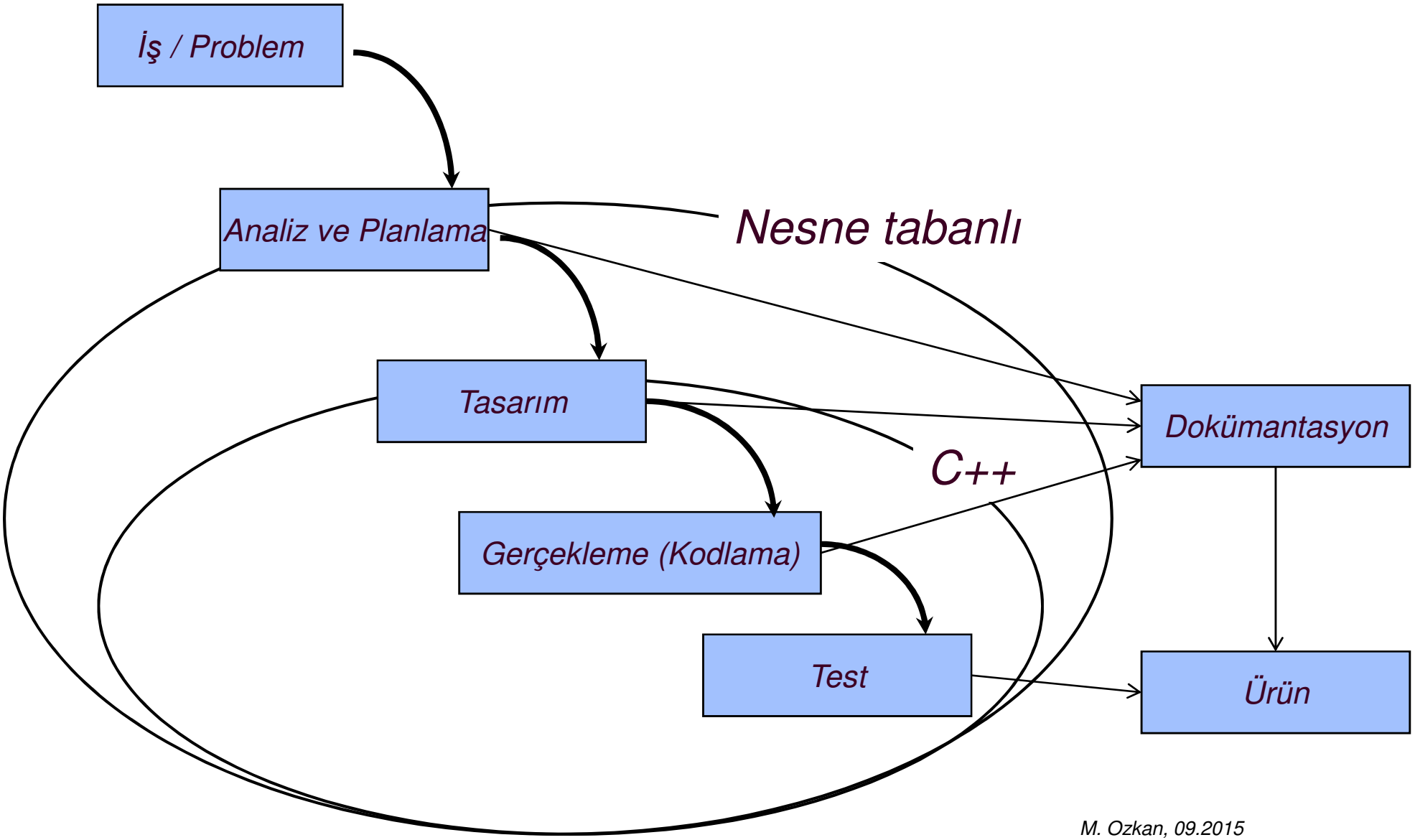
Müşteri:
Para ve kullanılan
kaynaklar anlamında
kabul edilebilir bir maliyete
problemlerini çözmek ister.

Kullanıcı:
Kolay öğrenmek,
Etkin kullanmak ister.

Geliştirici:
Kolay tasarlamak,
Kolay sürdürmek,
Kolay kodları tekrar
kullanmak ister.

Geliştirme Yöneticisi:
Daha düşük maliyete g
geliştirerek ve sürdürerek
müşterileri memnun edip,
daha fazla satmak ister.

Yazılım Geliştirme Süreci



Neden C++

- C++, diğer diller ile karşılaştırdığında belirli üstün özelliklere sahiptir:
  - Nesne tabanlı programlama dilidir
  - Taşınabilirlik (Portability)
    - Aynı C++ program kodunu, hemen hemen bütün bilgisayar tipinde ve işletim sisteminde, değişiklik yapmadan derleyebilirsin.
    - C++, dünyadaki en çok kullanılan ve taşınan dillerden biridir.
  - Kısalık (Brevity)
    - C++ ile kodlar çok kısa yazılabilir. Çünkü, anahtar kelimeler olarak özel karakterler kullanılır.
  - Modüler programlama
    - Bir C++ uygulaması, bir çok kaynak kod dosyasından oluşur. Kodlar ayrı ayrı derlenir (compile) ve birlikte bağlanır (link).
    - Sadece değişiklik yapılan kaynak dosyalar tekrar derlemeye tabi tutulduğu için, geliştirme sürecinde zaman kazandırır.

Neden C++

- C Uyumluluğu
  - C++, C dili ile geriye uyumludur.
  - C dilinde yazılmış kodlar, C++ dilinde yazılmış program içinde derlenebilir.
- Hız
  - C++ dilinde yazılmış program derleme sonucu yüksek hızda çalışabilir.
- C++ programcıları, Java ve C# gibi diğer nesne tabanlı dillere kolayca uyum sağlayabilir.
- C++, birçok uygulama alanında, bir çok programcı tarafından kullanılmaktadır.
  - Sistem programlama: işletim sistemleri, cihaz sürücüleri,
  - Bankacılık, ticaret,
  - Grafik ve kullanıcı arayüz programları,
  - Bilgisayar haberleşme programları.

Nesne Tabanlı Yaklaşım

- Dünya nesnelerden (objects) oluşmaktadır.
  - *Bilgisayar, lamba, saat, araba ,….*
- Dünya nesnelerden, nesneler daha küçük nesnelerden, böylece devam eder. Nesneler, bir araya gelerek daha büyük farklı nesneleri oluştururlar.
- Bu durumda, nesne tabanlı bir dünyada yaşadığımızı söyleyebiliriz.
- Nesneler iki kısımdan oluşmaktadır:
  - *Özelliklere sahiptir* (nitelikler/üye veri - attributes / member data)
    - *Nesnenin halihazırdaki durumunu belirtir. Örneğin, arabanın rengi, hızı, konumu, v.b.*
  - *Davranışlara sahiptir* (işlem/metot/üye fonksiyon - operations / methods / member functions)
    - *Nesnenin nasıl davranacağını ve tepki vereceğini denetler. Örneğin, gaz pedalı ile aracın hızının değiştirilmesi, direksiyon ile yününün değiştirilmesi, v.b.*

Nesne Tabanlı Yaklaşım

- Örneğin, bir öğrenci isim ve öğrenci numarasına (nitelikler - attributes) sahiptir; bir derse kaydolur, derslerden notlar alır (işlemler - operations).
- Nesne tabanlı paradigması, bütün hesaplamaların nesnelerin içerisinde yürütüldüğü bir problem çözümüne yaklaşımdır.
- Bu durumda, çalışan bir program, bir işin yapılması için işbirliği yapan nesneler topluluğu olarak görülebilir.
- Örneğin,
  - *Grafik programında: Nokta, çizgi, daire, v.b.*
  - *Matematik programında: karmaşık sayılar, matris,v.b.*
  - *Bilgisayar kullanıcı ortamında: Pencereler, menüler, butonlar, v.b.*
  - *Veri depolama yapılarında: diziler(arrays), yığınlar (stacks), link-lists,v.b*

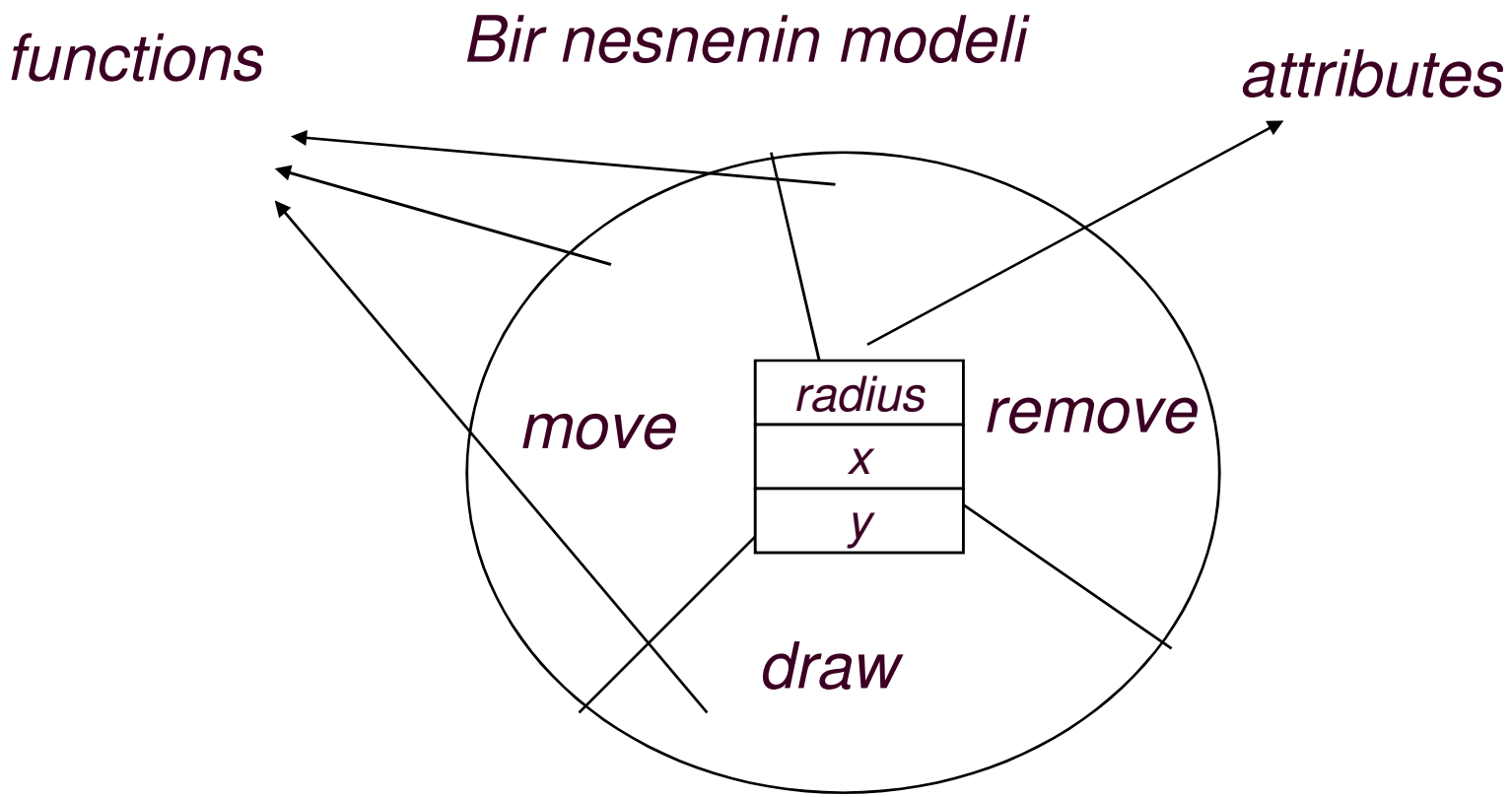
Nesne Tabanlı Yaklaşım

- Kapsülleme (Encapsulation/data hiding): Veri ve bu veri ile işlem yürüten fonksiyonların, tek bir varlık (class) içerisinde tutulmasını ifade eder. Nesnenin iç yapısının, nesne dışından saklanması şeklinde de açıklanabilir. Sadece nesnenin kendi metotları, nesnenin kendi alanını doğrudan denetleyebilir ve idare edebilir.
- Nesne tabanlı programlama yaklaşımında kapsülleme, sınıf (class) olarak adlandırılan yapılandırılmış veri tipi ile sağlanmaktadır.

Örnek bir nesne

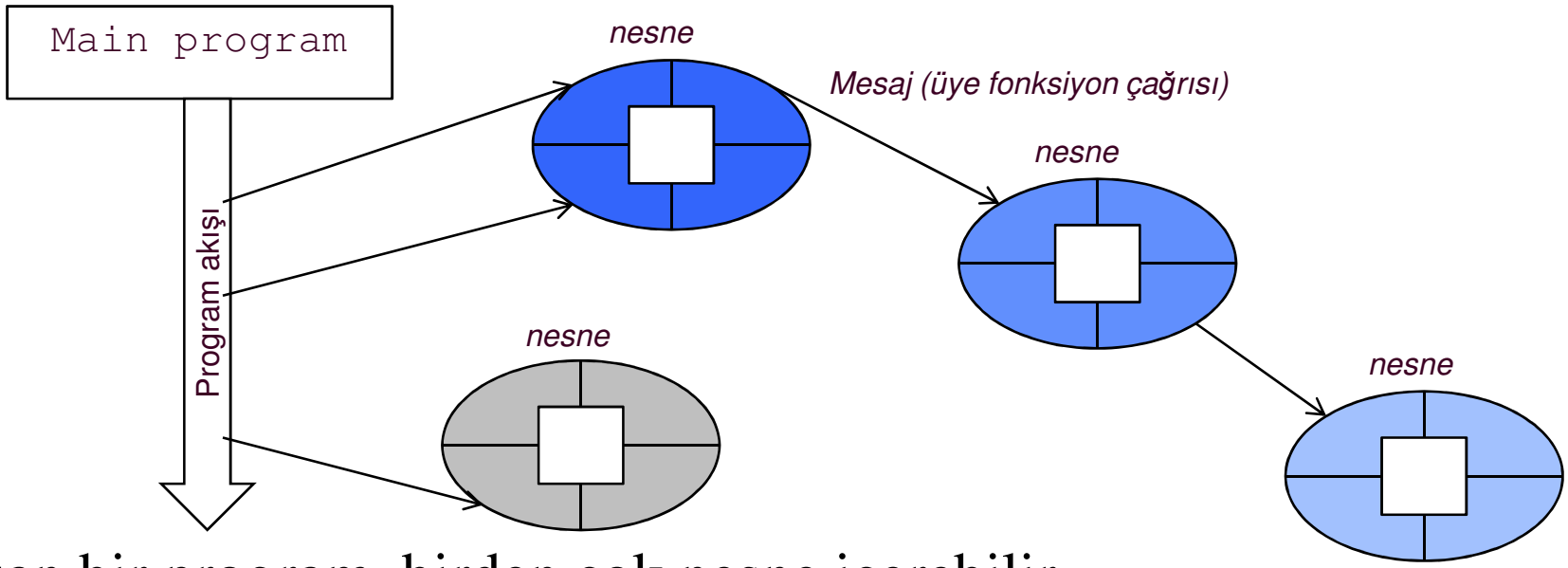
- Bir grafik programda kullanılan, daireler (Circle) bir nesne olarak oluşturulsun.
- Bir daire, iki-boyutlu uzayda 3 özelliğe (attributes) sahiptir.
  - *Yarıçap (Radius)*
  - *Merkezinin X-Y koordinat değerleri (X-Y coordinates of the center)*
- Bir dairenin, programdaki fonksiyonları (operations) şunlar olabilir.
  - *draw: Daireyi ekranda görünür yapar.*
  - *remove: Daireyi ekranda görünmez yapar.*
  - *move: Daireyi yeni koordinatlara taşır.*
- Nesnenin özellikleri ve fonksiyonları oluşturulduktan sonra, bu tarife uygun bir çok nesne yaratılabilir:
  - *Circle circle1, circle2;*
  - *circle1.move(10,20);*
  - *circle2.draw();*

Örnek bir nesne



- Gösterimde görüldüğü gibi, nesnenin sahip olduğu veriye dışardan erişim, ancak üye fonksiyonlar vasıtasıyla, fonksiyonların işleyişine uygun olarak gerçekleşmektedir. Nesnenin sahip olduğu veri, dışardan saklanmakta ve yalıtılmaktadır.

Örnek bir nesne



- Çalışan bir program, birden çok nesne içerebilir.
- Program işlemleri, ilgili nesnelerin üye fonksiyonlarını çağırarak, yaptırır.
- Bir nesne, kendi işlemlerini yürütürken, başka nesnelerin üye fonksiyonlarını çağırarak, bazı işlemlerini yaptırtabilir.
- Bir nesnenin üye fonksiyonunun çağrılması, ona mesaj gönderilmesi olarak ifade edilebilir.

Sonuç

- Nesne tabanlı yaklaşım, programcıya problem uzayındaki bileşenleri ifade etmek için araçlar sunar.
- Problem uzayındaki bileşenler (sınıflar-classes), çözüm uzayındaki (programs) ifadeleri (nesneler-objects) ile bulunur.
- Nesne tabanlı yaklaşımın faydaları
  - *İyi bir programın anlaşılması kolaylaşır. Hata nedenleri ve gerekli değişiklikler için programın analizi kolaylaşır.*
  - *Hataların oluşma olasılığı çok düşüktür.*
  - *Yeni yazılım bileşenlerinin (nesneler) eklenmesi ya da mevcut bileşenlerin değiştirilmesi kolaydır.*
  - *Yazılım bileşenleri (nesneler), başka projelerde kullanılabilir.*
  - *Takım çalışmasına uygundur. Bileşenler (nesneler) farklı takım üyelerince yazılabilir ve sonrasında kolayca entegre edilebilir.*

Sonuç

- Program yazımı, eğlencelidir, ancak program yazma, büyük oranda yazılım geliştirme fazlarının uygulanmasına bağlıdır.
- Kaliteli yazılım geliştirme, çok zor bir iştir; program yazma yetenekleri yanında, diğer yazılım geliştirme yetenekleri gerektirir.
- Bir sonraki, satranç oyun analojisini inceleyiniz.

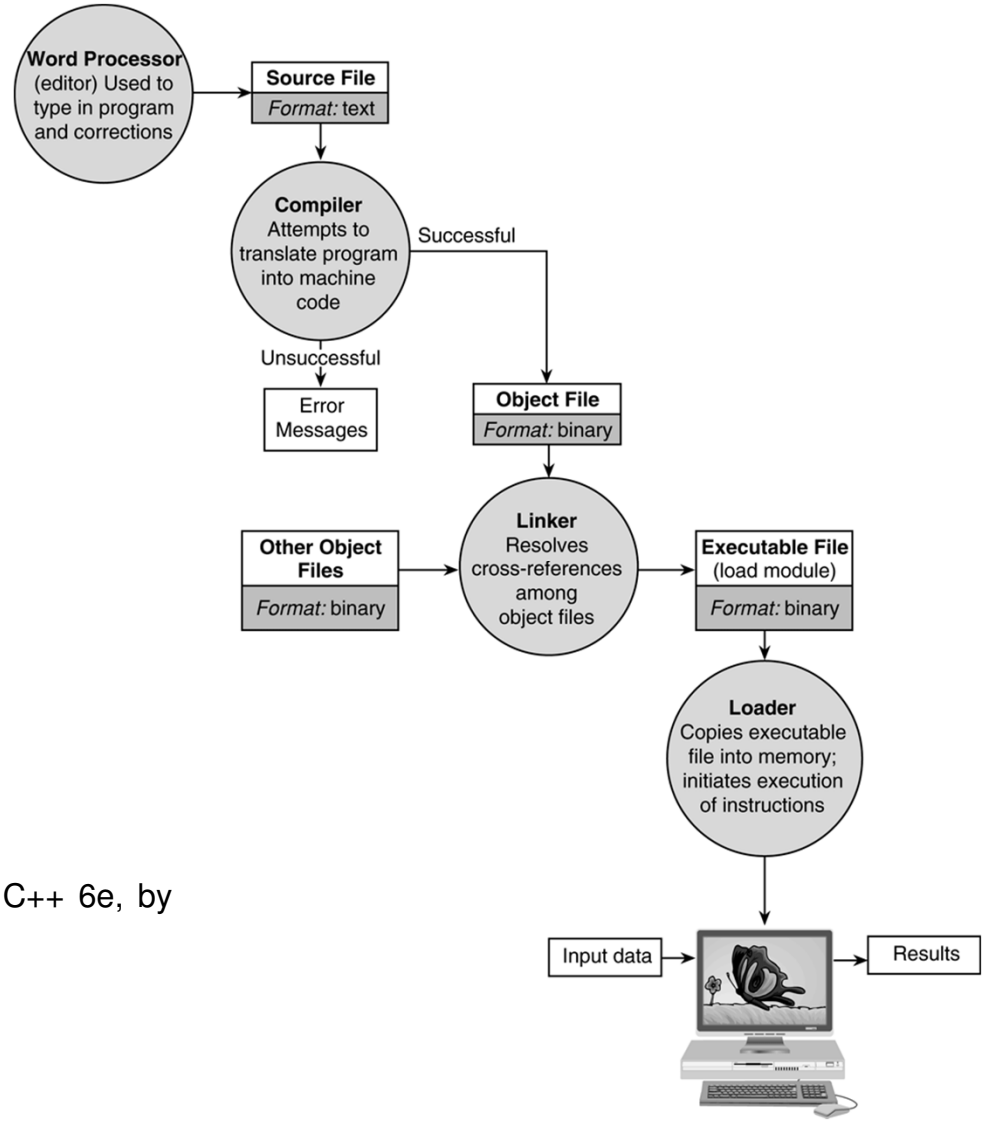
Satranç oynamayı öğrenme – Yazılım tasarımını öğrenme

- Satranç
  - *1. Temelleri: Oyunun kuralları ve fiziksel gereksinimleri öğrenilir. Oyun taşlarının isimleri ve hareket tipleri öğrenilir.*
    - *Bu durumda, oyuncu satranç oynayabilir ancak, iyi bir oyuncu olduğu söylenemez.*
  - *2. Prensipleri: Taşları korumanın değeri, taşların birbirine göre bağıl değeri, oyun tahtasının merkezinin stratejik değeri öğrenilir.*
    - *Bu durumda, oyuncu iyi bir satranç oyunu çıkarabilir. Oyunda nasıl karar vereceğini ve aptalca hatalar yapmamayı becerir.*
  - *3. Diğer ustaların oyunlarını (patterns) çalışma: Usta oyun hamleleri öğrenilir, hafızada tutulur, hamlelerin kullanılacağı durum geldiğinde uygulanır.*
    - *Bu durumda, oyuncu, usta bir satranç oyuncusu olabilir.*

Satranç oynamayı öğrenme – Yazılım tasarımını öğrenme

- Yazılım
  - *1. Temelleri: Programlama dilinin kuralları, veri yapıları, algoritmalar öğrenilir.*
    - *Bu durumda, programcı program yazabilir ancak ortaya çıkan iyi bir program değildir.*
  - *2. Prensipleri: Nesne tabanlı programlama öğrenilir. Soyutlamanın önemi, bilgi saklama, v.b.*
    - *Bu durumda, programcı programı daha iyi analiz eder ve daha az hatalı program yazabilir.*
  - *3. Diğer uzmanların tasarımlarını (patterns) çalışma: Uzmanların çeşitli problemler için kullandıkları tasarım yapıları öğrenilir, hafızada tutulur, benzer problem ile karşılaşıldığında kullanılır.*
    - *Bu durumda, programcı, kaliteli bir yazılım ortaya çıkarabilir.*

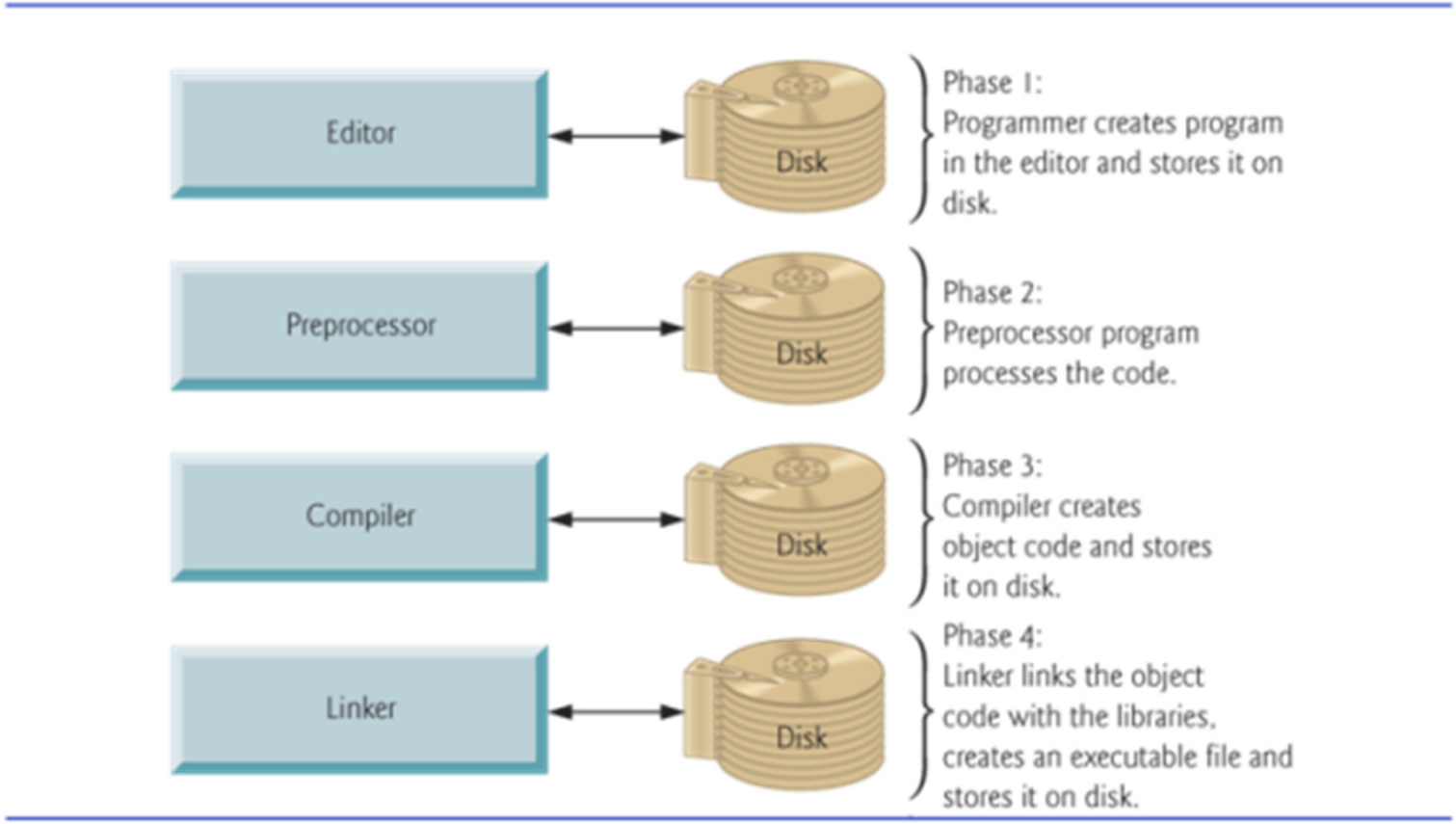
Karakteristik C++ Geliştirme Ortamı



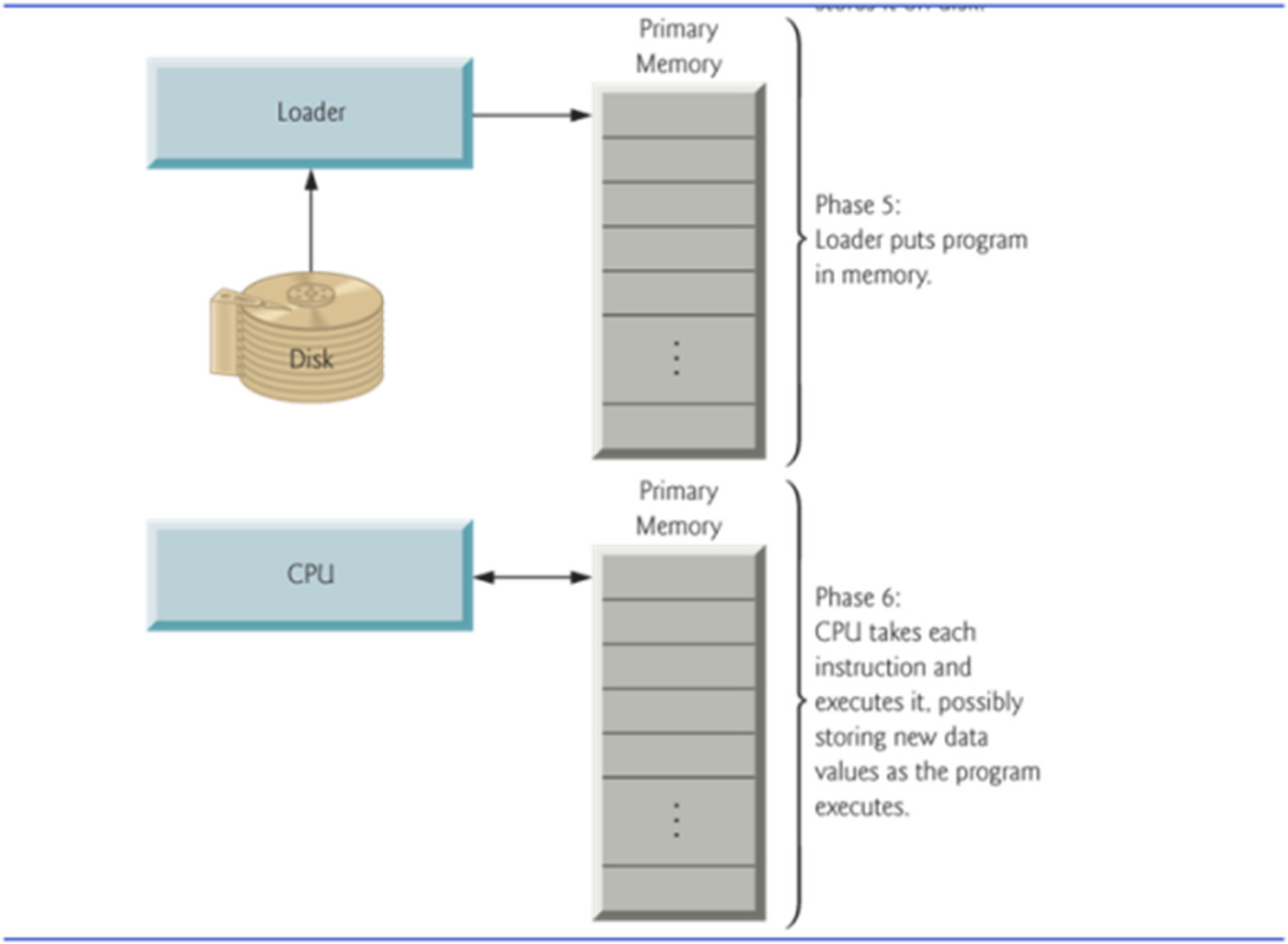
Ref: Problem Solving, Abstraction, and Design using C++ 6e, by Frank L. Friedman and Elliot B. Koffman

Karakteristik C++ Geliştirme Ortamı

- C++ programları, 6 fazdan geçer: edit, preprocess, compile, link, load ve execute.



Karakteristik C++ Geliştirme Ortamı



Kaynaklar

- T.C. Lethbridge and R. Laganiere, Object-Oriented Software Engineering - Practical software development using UML and Java, McGraw Hill, Second Edition, 2005.
- H.M.Deitel and P.J.Deitel, C++ How To Program, 9E, Pearson Press, 2014.
- B. Stroustrup, The C++ Programming Language, 3rd Edition, Special Edition, Addison Wesley, 2000.
- Dr. Feza Buzluca, Ders Notları.
- Ç. Turhan ve F.C. Serçe, C++ Dersi: Nesne Tabanlı Programlama, 2nci Baskı, 2014.