



# Sistem Programlama

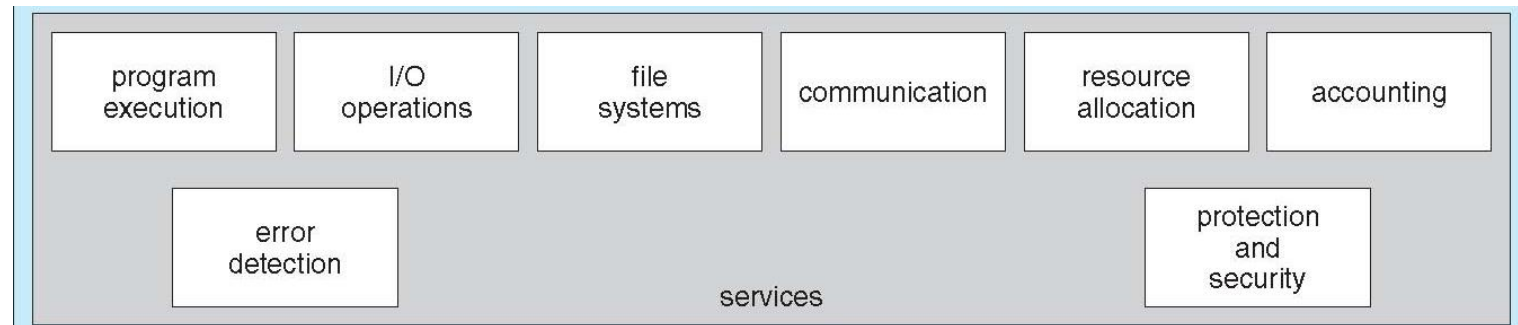
---

DR. ÖĞR. ÜYESİ ABDULLAH SEVİN

# Sistem Programlama nedir?

---

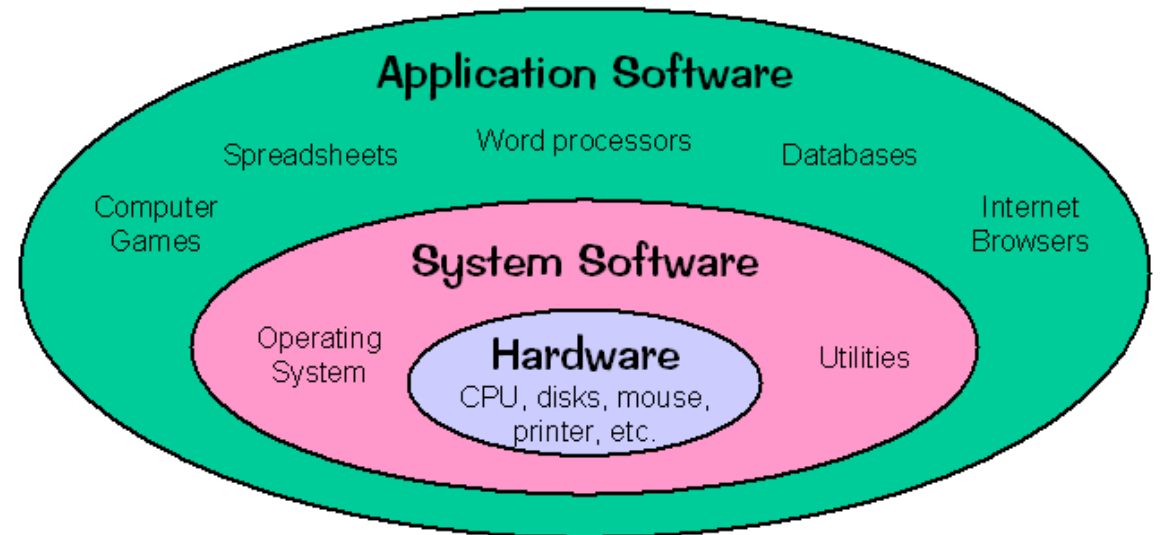
- **Sistem programlama:** program geliştirme için sistem araçlarının kullanılmasıdır.
- Programlama bilgilerine ek olarak, *bilgisayar mimarisi ve işletim sistemi* bilgisi
- **Uygulama programları** ile direkt olarak kullanıcıya servisler sağlanırken,  
**Sistem programları** ile *uygulama programlarına servisler sağlanır.*
- Sistem programlamada, işletim sistemi servisleri ile etkileşim kuran programlar yazılır.



# Sistem Programlama

---

- **Bilgisayar;** donanım, sistem yazılımı ve uygulama yazılımı olmak üzere üç ana parçaya ayrılabilir.
- **Donanım ve sistem yazılımı,** uygulama programlarını çalıştırmak için kullanılır.
- **Sistem yazılımı,** donanım ve uygulama yazılımları arasında bir **soyutlama** katmanı oluşturur.



# Sistem Programlama

---

- Sistem yazılımı, bilgisayar donanımının çalıştırılması için kullanılır.
- Sistem yazılımı, bir bilgisayar sisteminin donanım bileşenlerini **kontrol** etmek, **bütünleştirmek** ve **yönetmekten** sorumludur,
- böylece diğer yazılımlar ve sistem kullanıcıları, veri aktarımı gibi düşük seviyeli ayrıntılarla ilgilenmek zorunda kalmazlar. (Bellekten diske kopyalama veya bir ekrana yazı yazdırma vs. )



# Sistem Programlama

---

- Genel olarak, sistem yazılımı bir işletim sisteminden ve disk biçimlendiriciler, dosya yöneticileri, görüntü yöneticileri, metin düzenleyiciler, kullanıcı kimlik doğrulaması (oturum açma) ve yönetim araçları ve ağ ve cihaz kontrol yazılımı gibi bazı temel yardımcı programlardan oluşur.
- Genelde kullanıcılar, arka planda çalıştığı için sistem yazılımı ile etkileşime girmezler.

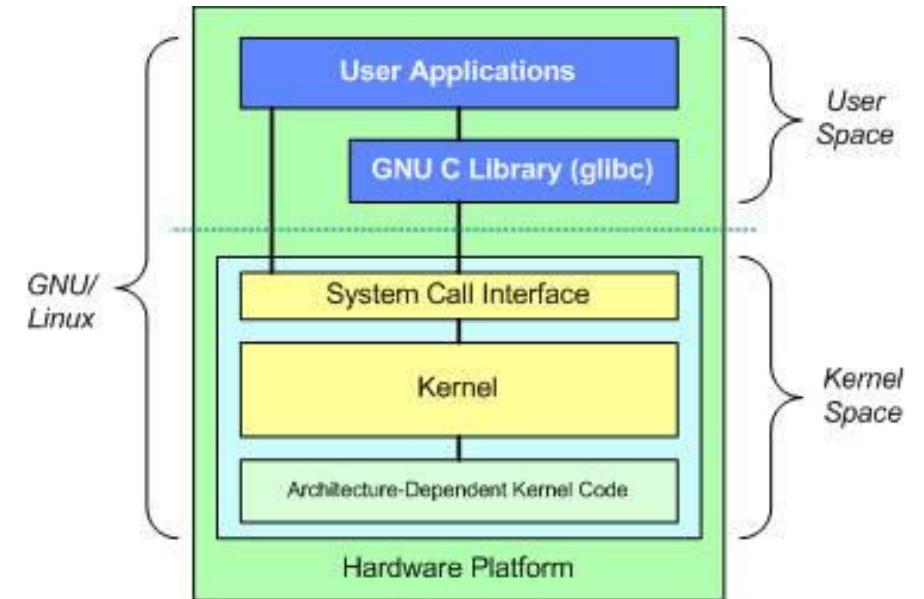
# Hedefler

---

1. Bir işletim sisteminin temellerini kullanıcı açısından anlamak.
2. Neden C'de programlama yaptığımızı anlayacak kadar iyi C programlamak.
3. Sistem programlama hakkında bilgili olmak.
4. Temel tek işlemcili bilgisayar organizasyonunu anlamak.
5. Sistem hakkında donanımsal bilgi sahibi olmak
6. Donanımı gerektiğinde kontrol edebilmek

# Sistem Programlama

- Sistem kütüphanelerinin oluşturduğu **soyutlama katmanı (abstraction)** ile bir fonksiyonu (Ör: printf ) donanımın detaylarını bilmeden kullanabiliriz.
- Yanda verilen Linux mimarisinde görülen GNU C kütüphanesi gibi, bir sistem benzer kütüphaneleri içerdiği sürece uygulama o sistem üzerinde kullanılabilir.
- Sistem araçlarının kullanımı standartların oluşmasını sağlar ve böylece geliştirilen programlar diğer bilgisayarlara kolayca transfer edilir.



# Neden C dili

---

- ❑ **Sistem programlamada C kullanılmasıının sebepleri:**
- ❑ C programı; işletim sistemleri, aygıt sürücüler, ağ sunucuları gibi uygulamalarda kullanılarak modern bilgisayarın temelini oluşturur.
- ❑ C programlama **en az soyutlamaya** sahiptir. Bundan dolayı donanıma daha yakındır.
- ❑ Birçok C ifadesi direkt olarak makine koduna dönüştürülebilir.
- ❑ C programlamada hafızaya işaretçiler (pointers) aracılığı ile erişilebilir ve böylece sistemin parçalarına ulaşma imkanı sağlanır.



# Neden C

---

- ❑ Yalnızca dilin veya sistemin mevcut olduğu birçok durumda kullanılır.
  - ❑ Küçük, gömülü sistemler vs.
- ❑ "Üst düzey dilleri" desteği olmayan birçok "düşük düzey" durum var.
  - ❑ İşletim sistemleri, gerçek zamanlı sistemler, sürücüler

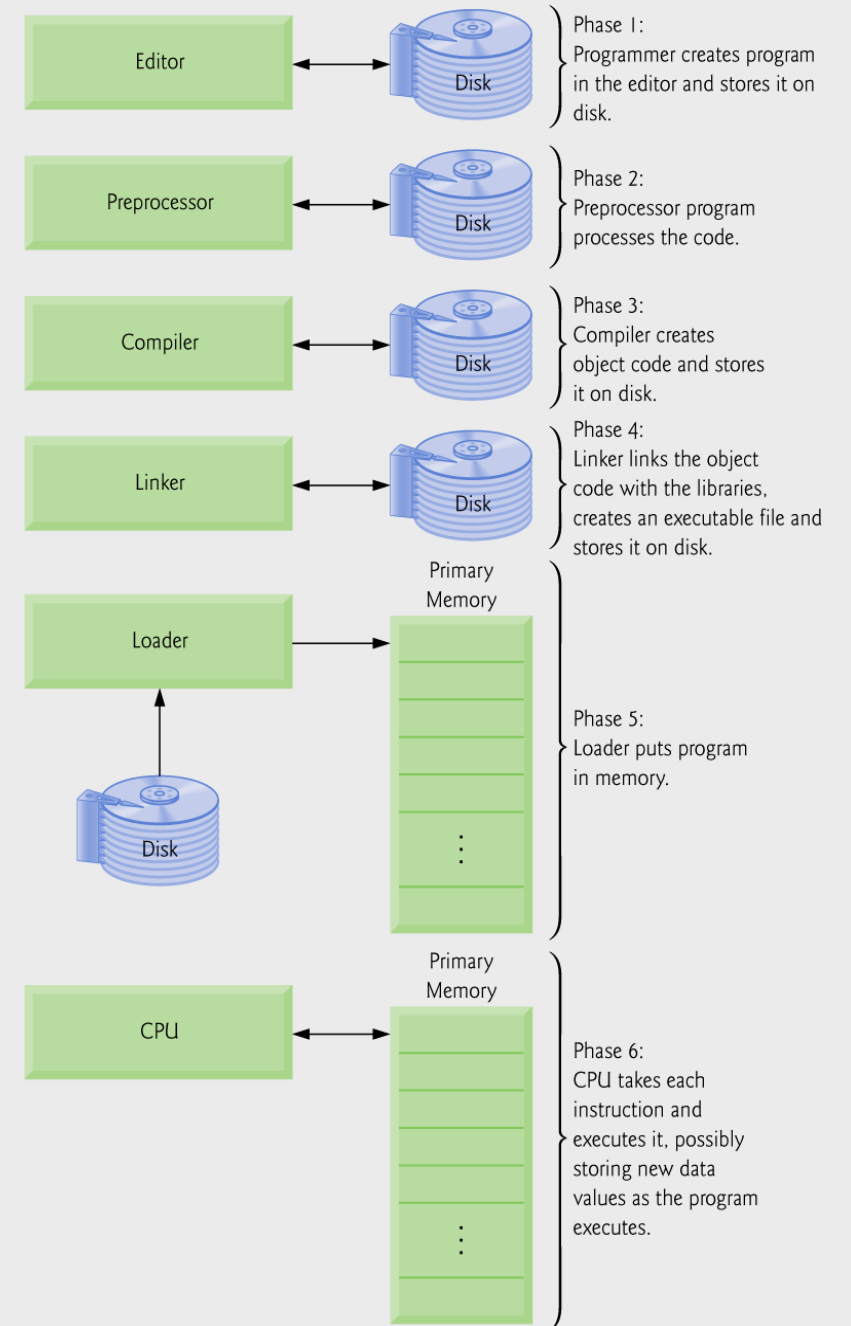
# C prog. Geliştirme ortamı

## Standard Adımlar

- Düzenleme
- Önışlemci
- Derleme
- Bağlama
- Yükleme
- Yürütme

**Deitel & Deitel**

© 2007 Pearson Ed -All rights reserved



# C prog. Geliştirme ortamı

---

- Edit: Düzenleme
- Önışlemci: Derleme sürecinde ayrı bir adımdır. Basit bir ifadeyle, bir C Ön İşlemcisi yalnızca bir **metin değiştirme** aracıdır ve derleyiciye gerçek derlemeden önce gerekli ön işlemeyi yapmasını söyler. Tüm önışlemci komutları bir kare simgesiyle (#) başlar. (#define,..)
- Derleme
- Bağlama
- Yükleme
- Yürütme

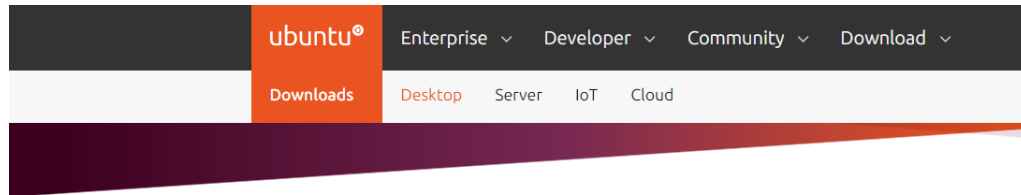
# Neden Linux?

---

- ❑ **Açık kaynak** olması sebebiyle, sistem programlama dersinde Linux tabanlı işletim sistemini tercih edilmektedir.
- ❑ Linux tabanlı işletim sisteminin çalışmasıyla ilgili detaylar incelenebilir. **Parçalar eklenebilir veya değiştirilebilir.**
- ❑ Ticari olmamasından dolayı kapalı bir sistem değildir.

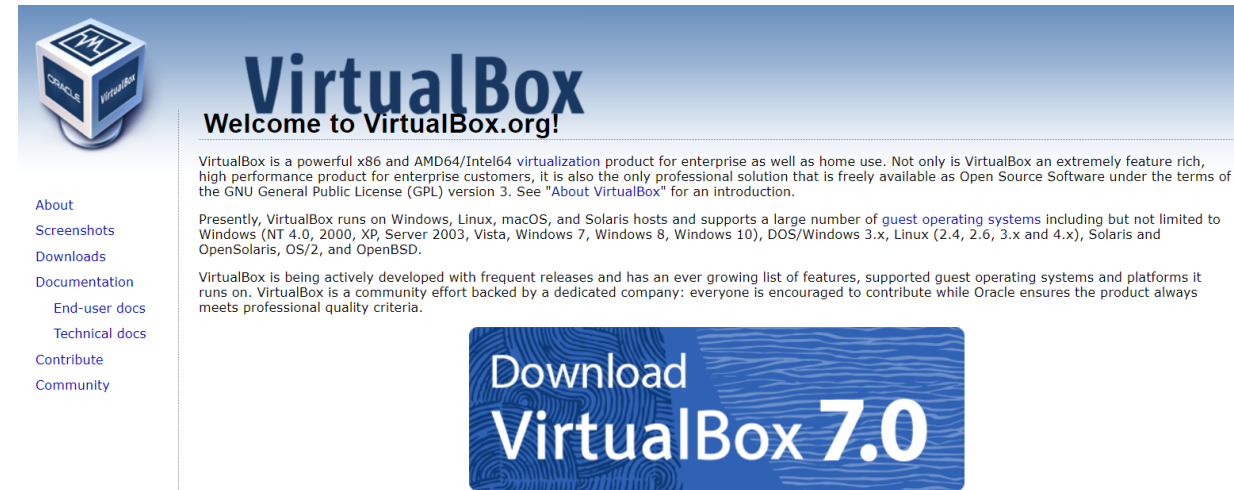
# İşletim Sistemi

- ❑ Ders ile ilgili uygulamaları gerçekleştirmek için Linux tabanlı Ubuntu İşletim sistemini kullanabilirsiniz.
- ❑ Farklı İşletim Sistemi üzerine kuracak iseniz Sanal Makine kullanmalısınız. (Örn: Windows üzerinde Oracle VM VirtualBox)



## Download Ubuntu Desktop

The open-source desktop operating system that powers millions of PCs and laptops around the world. Find out more about Ubuntu's features and how we support developers and organisations below.

[Ubuntu Desktop homepage](#)[Visit the Ubuntu Desktop blog ›](#)

# Konular

---

## **Derse ait temel başlıklar aşağıdaki gibi sıralanabilir:**

- C programlama dili hakkında temel bilgiler
- C işaretçiler, malloc, stringler, vb...
- Dosyalar ve izin dosyaları, Sinyaller
- Bağlantılar (linkler), Shell yönlendirme (Shell Redirection)
- Dosyadan okuma/yazma (File I/O), sistem çağrıları ve tampon bellek (buffer) kullanımı
- Sistem çağrıları ve giriş çıkış
- Simgesel dil (Assembly) (yerel değişkenler, fonksiyonlar, dallanma)
- Prosesler (süreçler) ve ilgili sistem çağrıları (fork, exec, dup, pipe)
- Prosesler arası iletişim, sinyaller



# Değerlendirme

---

1. Ara Sınav	%60
1. Kısa Sınav	%10
1. Proje / Tasarım	%30
1. Final	%50

# Kaynaklar

---

<http://web.eecs.utk.edu/~huangj/cs360/index.html>

## **CS360 -- Systems Programming**

**Jian Huang --- Fall 2022**

---

### **CS360 LINKS**

- [General Information](#)
- [Syllabus](#)
- Homeworks
- [Labs](#)
- [Lecture Notes](#)
- [The Lab Home Page](#) (maintained by the TA's)

---

### **INTRODUCTION AND CLASS GOALS**



THE UNIVERSITY OF  
**TENNESSEE**  
KNOXVILLE

# Kaynaklar

---

[http://web.eecs.utk.edu/~jplank/plank/classes/cs360/lecture\\_notes.html](http://web.eecs.utk.edu/~jplank/plank/classes/cs360/lecture_notes.html)

## CS360 -- Systems Programming

### James S. Plank

#### Lecture Notes

Although you can read the notes here, I have put them on bitbucket, and you can grab them to compile and use on your own machine. I strongly suggest that you do so, so that you have all of the programs on your own machine, do:

```
UNIX> git clone https://jimplank@bitbucket.org/jimplank/cs360-lecture-notes.git
```

The lecture notes will be in the directory **cs360-lecture-notes**. You should occasionally do a "git pull" in that directory, to make sure that your notes are up to date.

#### Lecture notes on bitbucket

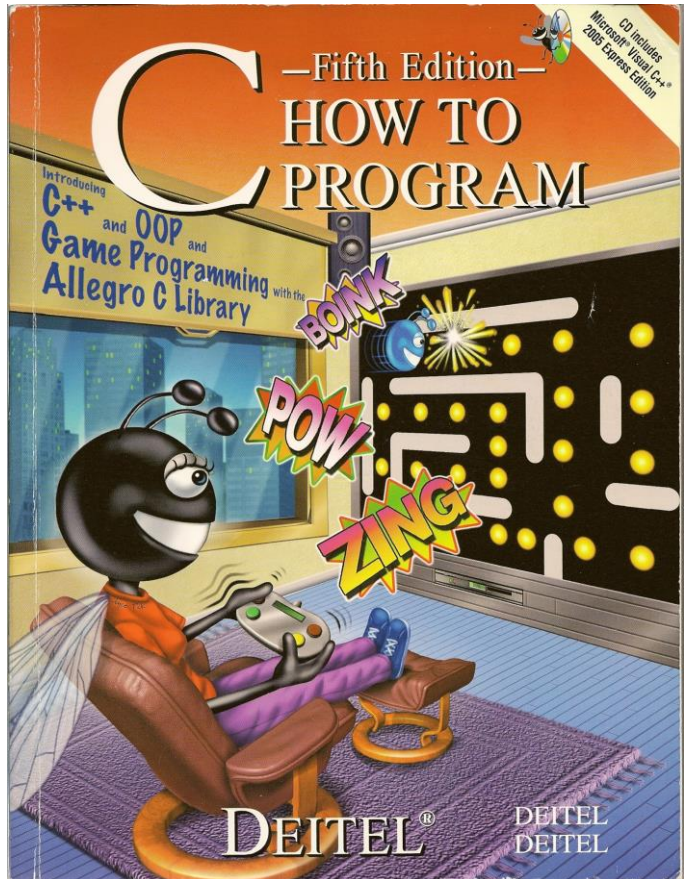
- [Lecture 1:](#) "C Stuff 1:" Getting Started with C, Scalar Types and Aggregate Types
- [Lecture 2:](#) "C Stuff 2:" Pointers, Casting, Malloc, Segmentation Violations and Bus Errors
- [Lecture 3:](#) Pointer Arithmetic (Small Lecture)
- [Lecture 4:](#) Strings in C
- [Lecture 5a:](#) Libfdr -- The Code
- [Lecture 5b:](#) Libfdr -- Fields
- [Lecture 5c:](#) Libfdr -- Jvals
- [Lecture 5d:](#) Libfdr -- Dllist: Doubly-Linked Lists
- [Lecture 5e:](#) Libfdr -- JRB: Red-Black Trees
- [Lecture 6:](#) Some Basic Terminology
- [Lecture 7:](#) Introduction to System Calls and I/O
- [Lecture 8:](#) Cat and Buffering
- [Lecture 9:](#) Links
- [Lecture 10:](#) Sh Redirection
- [Lecture 11:](#) Stat and Opendir/Readdir/Closedir



THE UNIVERSITY OF  
TENNESSEE  
KNOXVILLE

# Ek Kaynak

---



Deitel & Deitel, 5<sup>th</sup> ed.  
*C & C++*

[Paul Deitel](#) (Author), [Harvey Deitel](#) (Author)

# Ek Kaynak

---

Deitel & Deitel, 6<sup>th</sup> ed.

*C* & *C++*

Similar to 5<sup>th</sup> edition.

