



# **Bölüm 1: Bilgisayarlar**

## **JAVA ile Nesne Yönelimli Programlama**



# Temel Parçalar

- Bir bilgisayar, temelde birkaç önemli parçadan oluşur.
  - **İşlemci (CPU):** Bilgisayarın beyni olarak tüm aritmetiksel ve mantıksal hesaplamaları yapar.
  - **Bellek (RAM):** Geçici verileri saklar, dolayısıyla kapasite ve veriye erişim hızı gibi özellikleri bilgisayarın çalışma hızını etkiler.
  - **Sabit Disk:** Verileri kalıcı olarak saklar.



# Bilgisayarın İşleyişi

- Temel adımlar:
  - İşlemciye talimatlar gönderilir.
  - İşlemci, talimatları sırayla işler.
  - İşlem sonuçları kullanıcıya veya diğer yazılımlara sunulur.
- Programlama,
  - İşlemcinin anlayacağı dilde talimatlar yazma sürecidir.
  - Bilgisayarın istenen görevleri yerine getirmesi için kullanılır.



# Bilgisayar Nasıl İnşa Edilir?

- **Sayılar (Numbers)**
- Harf ve Dizgeler (Letters and Strings)
- Yapılandırılmış Bilgi (Structured Information)
- Saklanan Program Kavramı (Stored Program Concept)
- von Neumann Mimarisi (von Neumann Architecture)



# Sayılar

- Bilgisayarın temel işlevi elektrik sinyallerini belirli kurallara göre işlemektir.
  - Bu kurallar sayesinde bilgisayar birçok işlevi gerçekleştirir.
- Elektrik sinyallerini kullanıcıların anlayabileceği sayı ve sembollerle ilişkilendirmek önemlidir.
  - Bu sayede bilgisayar, bize anlamlı bilgiler sunabilir.
- Elektrik sinyalleri, ikili (binary) temsili kullanılarak işlenirler.



# Sayılar

- Tam sayıları temsil için 2'nin üssü olan sayıların kombinasyonları kullanılır.
- Bu sistem "ikili" veya "taban 2" olarak adlandırılır.
- Her bir basamak (bit), 0 veya 1'i temsil eder. Örneğin:
  - 0 veya 1
  - Yanlış veya Doğru
  - Kapalı veya Açık
  - Düşük voltaj veya Yüksek voltaj
- Tüm işlemler ikili sistem temeli üzerine inşa edilmiştir.



# Sayılar

- 4 ardışık üs kullanılarak ( $2^0$ ,  $2^1$ ,  $2^2$ ,  $2^3$ ), 0 ile 15 arasındaki tam sayılar temsil edilebilir.
- Bu, her bir üssü 0 veya 1 kez kullanarak yapılır. Örneğin:
  - $13_{10} = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1101_2$
  - İkili tabanda 1101 sayısı, ondalık tabanda 13 sayısına denk gelir.
- Sayıları temsil etmek için farklı sayı sistemleri kullanılabilir.
  - 603 ondalık tabanda,  $603_{10} = 6 \cdot 10^2 + 0 \cdot 10^1 + 3 \cdot 10^0$  olur.
  - 207 sekizlik tabanda,  $207_8 = 2 \cdot 8^2 + 0 \cdot 8^1 + 7 \cdot 8^0$  olur.



# Sayılar

- Herhangi bir taban ( $b \geq 2$ ) seçildiğinde,
  - 0 ile  $b^d - 1$  arasındaki her pozitif tam sayı,
  - $d$  adet basamak kullanılarak temsil edilebilir.
  - Bu basamakların katsayıları 0'dan  $b - 1$ 'e kadar değerler alabilir.
- 64-bit bir bilgisayar,  $2^{64} - 1$ 'e kadar olan tam sayıları temsil edebilir.





# Sayılar

- Tüm sayı sistemleri, aritmetik işlemleri gerçekleştirirken benzer temel kuralları takip eder.
- Hangi sayı tabanı kullanılırsa kullanılsın, toplama işleminde geçerli ilkeler:
  - basamak değerleri, taşıma işlemleri ve sonucun doğru hesaplanması.
- Toplama İşlemi: 10 Tabanı (Ondalık):
  - $57 + 28 = 85$
- Toplama İşlemi: 2 Tabanı (İkili):
  - $00111001 + 00011100 = 01010101$



# Sayılar

- Negatif tam sayıları temsil etmek için genellikle «ikiye tümler» (two's complement) yaklaşımı kullanılır.
- Bu yaklaşım, pozitif sayının bitlerini tersine çevirip sonra 1 eklemeyi içerir.
- Sonuç, negatif sayının temsili olur.
- Örneğin;
  - 8-bitlik bir bilgisayarda 3 sayısı şu şekilde gösterilir: 00000011.
  - Negatif 3 sayısını temsil etmek için,
    - Bitler tersine çevrilir.  $00000011 \rightarrow 11111100$ .
    - Ardından 1 eklenir,  $11111100 + 1 \rightarrow 11111101$ .



# Sayılar

- Sınırlı sayıda rakam (0-9) kullanılarak sınırlı sayı temsil edilebilir.
- Çok büyük/küçük sayıları temsil için kesir ve üs bölümleri kullanılır.
- Kesir ve üs kısımları ayrı olarak ele alınır.
- Örneğin, 31415901 sayısı,
  - İlk altı rakam (314159) kesirli kısmı temsil eder.
  - Son iki rakam (01), sayının üssünü temsil eder.
  - 01, 10'un kuvveti olan  $10^1$ 'i ifade eder.
  - Sonuç  $0.314159 \times 10^1 = 3.14159$  olur.



# Bir Bilgisayar Nasıl İnşa Edilir?

- Sayılar (Numbers)
- **Harf ve Dizgeler (Letters and Strings)**
- Yapılandırılmış Bilgi (Structured Information)
- Saklanan Program Kavramı (Stored Program Concept)
- von Neumann Mimarisi (von Neumann Architecture)



# Harf ve Dizgeler - ASCII

- Metin tabanlı veri işlemek için harfleri sayısal olarak temsil etmek gerekir.
- ASCII, Amerikan Ulusal Standartlar Enstitüsü (ANSI) tarafından belirlenen bir kodlama standardıdır.
- ASCII, büyük ve küçük harfleri, rakamları ve özel karakterleri sayısal olarak temsil etmek için kullanılır.
  - 8-bit bir kod olduğundan 256 farklı sembolü temsil edebilir.
- Farklı dillerde kullanılan karakterlerin temsili için yetersizdir.



# Harf ve Dizgeler - Unicode

- Unicode, Uluslararası Standartlar Organizasyonu (ISO) tarafından oluşturulan bir kodlama sistemidir.
- 16-bit kodlama ile tüm dillerdeki harf, sembol, ve işareti temsil edebilir.
- Unicode İngilizce gibi dillerde gereğinden fazla bellek alanı kullanabilir.
- Bu nedenle, ISO, "Unicode Dönüşüm Formatları" (UTF) adı verilen alternatif kodlama yöntemi tanımlamıştır.
- UTF-8, Unicode karakterlerini temsil için dinamik bir yaklaşım kullanır.
- Sık kullanılan karakterler daha az bellek kullanırken, nadiren kullanılan karakterler daha fazla bellek kullanır.



# Harf ve Dizgeler - Emoji

- İletişimi zenginleştiren ve duyguları, ifadeleri veya nesnelere hızla aktarmaya yarayan simgelerdir.
- Bilgisayarlar tarafından metin olarak işlenir ve karakterler gibi temsil edilir.
- Karmaşık ve renkli olduklarından temsil için özel bir standart geliştirilmiştir.
- Standartlar, Unicode Consortium tarafından belirlenmiştir.
- Bu standartlar, her bir emoji'yi benzersiz bir kodla temsil eder.
- Böylece, emojiler her cihazda ve uygulamada aynı şekilde görünür.



# Harf ve Dizgeler - String

- Dizge (string), metin veya karakterlerden oluşan bir veri türüdür.
- Örneğin, "Merhaba Dünya!" bir dizgedir.
- Metin işleme ve iletişimde sıkça kullanılır.
- Dizge içindeki her karakterin (harf, rakam, sembol) bir sayı değeri vardır.
- Uzunluğunu belirlemek için bir "uzunluk alanı" (length field) kullanılır.
- Örneğin, «çikolata» dizgesi
  - ASCII kodlamasına göre: 231, 105, 107, 111, 108, 97, 116, 97 sayıları ile temsil edilir.





# Bir Bilgisayar Nasıl İnşa Edilir?

- Sayılar (Numbers)
- Harf ve Dizgeler (Letters and Strings)
- **Yapılandırılmış Bilgi (Structured Information)**
- Saklanan Program Kavramı (Stored Program Concept)
- von Neumann Mimarisi (von Neumann Architecture)



# Yapılandırılmış Bilgi

- Sayı dizileri, bilgiyi anlamak ve bilgisayarla etkileşimde bulunmaya yarar.
- Her türlü bilgi sayılarla temsil edilebilir.
  - **Resimler:** Her piksel, kırmızı, yeşil ve mavi bileşenlerin miktarını temsil eden üç sayıdan oluşur. Örneğin, bir mavi piksel için (0, 0, 255) gibi.
  - **Sesler:** Ses, havadaki "ses basınç seviyeleri" olarak temsil edilir. Bu, bir sese ait titreşimlerin zaman içindeki kaydını ifade eder.
  - **Filmler:** Bir film, saniyede 24 veya 30 kare gibi hızlarda ardışık resimlerin dizisiyle temsil edilir. Her kare, resimlere benzer şekilde sayı dizileriyle temsil edilir.



# Bir Bilgisayar Nasıl İnşa Edilir?

- Sayılar (Numbers)
- Harf ve Dizgeler (Letters and Strings)
- Yapılandırılmış Bilgi (Structured Information)
- **Saklanan Program Kavramı (Stored Program Concept)**
- von Neumann Mimarisi (von Neumann Architecture)



# Saklanan Program Kavramı

- Modern bilgisayarların temel ilkesini oluşturur.
- Bilgisayarın belleğinde saklanan komutlar ve veriler programları oluşturur.
- ENIAC'tan farklı olarak, EDVAC'da komutlar ve veriler bellekte sıralı olarak saklanır, bu sayede programlar esnek ve genişletilebilir hale gelmiştir.
- Her komut, bir görevi veya işlemi temsil eder.
- Komutlar ardışık olarak işlenir.
- Bazen bir koşul sağlandığında program farklı bir komuta atlayabilir.
- Bu, programların belirli şartlara göre farklı yollar izlemesini sağlar.



# Saklanan Program Kavramı

- Bellek, bilgisayarın verileri ve programları geçici olarak sakladığı yerdir.
- Bellek hiyerarşisi genellikle üç seviyede incelenir:
  - **İç (L1) Cache:** En hızlı önbellek türü. İşlemciye yakın konumda yer alır.
  - **Orta (L2) Cache:** L1 önbelleğe destek, daha büyük kapasiteye sahiptir.
  - **Ana Bellek (RAM):** Daha büyük depolama alanı, daha yavaş çalışır.
- Bellek hiyerarşisi, bilgisayarın performansını doğrudan etkiler.
- Daha hızlı ve işlemciye yakın bellekler, daha hızlı veri erişimi sağlar.



# Saklanan Program Kavramı

- «Al-Çöz-Çalıştır» (Fetch-Decode-Execute) döngüsü, programların işlemci tarafından nasıl yürütüldüğünü tanımlar.
- Bu döngü üç adımdan oluşur:
  - Al,
  - Çöz ve
  - Çalıştır.
- "Al-Çöz-Çalıştır" döngüsü sürekli olarak tekrarlanır.
- İşlemci sıradaki komutu alır, çözer ve çalıştırır.
- Ardından bir sonraki komutu alır ve bu döngü devam eder.



# Saklanan Program Kavramı

- **Adım 1: Al (Fetch):**
  - İşlemci programın bir sonraki komutunu bellekten alır.
- **Adım 2: Çöz (Decode):**
  - Alınan komut çözülür.
  - Her komutun belirli bir işlevi vardır.
- **Adım 3: Çalıştır (Execute):**
  - Çözülen komut çalıştırılır.
  - İşlemci komutun gerektirdiği işlemi gerçekleştirir.
  - Bu işlem, veri işleme, sonuç hesaplama, bir işlemi başlatma olabilir.



# Saklanan Program Kavramı - CPU

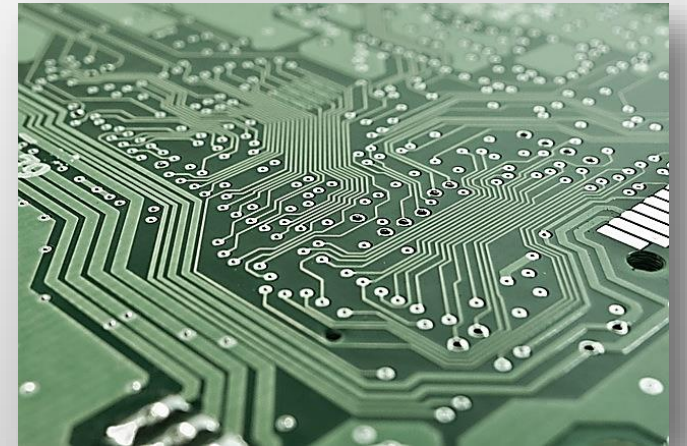
- Merkezi İşlem Ünitesi (CPU), bilgisayarın "beyni" olarak düşünülebilir.
- Tüm hesaplamalar ve işlemler burada gerçekleşir.
- CPU, iki temel bileşenden oluşur:
  - Aritmetik işlem birimi (ALU)
    - Hesaplamaları gerçekleştirir.
  - Kontrol Birimi
    - Komutları yönlendirir ve işlemleri düzenler.
- CPU,
  - Aritmetik işlemleri gerçekleştirmek için parçalara,
  - Geçici verileri saklamak için düşük kapasiteli yazmaç'lara sahiptir.





# Saklanan Program Kavramı - Bus

- Veri yolu (BUS), birçok paralel teller içeren bir iletişim yolu sistemidir.
- Veri yolu, üç ana türde sinyali taşır:
  - **Adres Sinyalleri:** Belirli bir bellek veya cihazı hedeflemek için kullanılır.
  - **Veri Sinyalleri:** Gerçek verileri taşır, bu veriler işlenir veya depolanır.
  - **Kontrol Sinyalleri:** Veri yolu üzerindeki işlemi yönlendiren sinyallerdir.
    - Örneğin, yazma veya okuma işlemlerini kontrol eder.





# Saklanan Program Kavramı - Bellek

- Bellek, bilgisayarda verilerin ve talimatların saklandığı depolama alanıdır.
- Bir program, uzun bir talimat listesi içerir.
- Bu talimatlar, bilgisayarın ne yapması gerektiğini belirler.
- Bir programdaki talimatlar, bellekten tek tek alınır ve işlemci içindeki özel bir alana «yazmaçlara» yüklenir.
- İşlemci bu yazmaçlardaki talimatları sırayla çalıştırır.
- Bu yazmaçlar, aritmetik ve mantıksal işlemler için kullanılır.
- En önemli yazmaçlardan biri, **program sayacıdır**. İşlemciye sırada hangi talimatın olduğunu ve hangi talimatın işlenmesi gerektiğini söyler.



# Saklanan Program Kavramı

- Bellek adresi, bir bellek hücreğine işaret eder.
- Burada, bir talimatın veya verinin sayı ile kodlanmış şekli bulunur.
- Veriler, işlemci tarafından işlenirken, talimatlar sırayla çalıştırılır.
- Örneğin, 200 bellek adresinde "ADD to R1" talimatı, 201 bellek adresinde "102" gibi bir veri değeri olabilir.
- İşlemci talimatları adres sırasına göre çalıştırır.
- Bazı talimatlar işlemcinin sırayı değiştirmesini sağlar.
- Örneğin, 204 bellek adresindeki "JUMP 7 bytes" komutu, işlemcinin 7 bayt sonrasındaki talimata atlamasını sağlar.



# Bir Bilgisayar Nasıl İnşa Edilir?

- Sayılar (Numbers)
- Harf ve Dizgeler (Letters and Strings)
- Yapılandırılmış Bilgi (Structured Information)
- Saklanan Program Kavramı (Stored Program Concept)
- **von Neumann Mimarisi (von Neumann Architecture)**



# von Neumann Mimarisi

- Her komut, bir işlem kodu (opcode) ile temsil edilir.
- İşlem kodları, hangi işlemin yapılacağını belirtir.
- 8-bit, 4 yazmaç, 256 adet 8-bit bellek hücresi ve 4 komut (toplama, çıkarma, çarpma ve bölme) kümesine sahip bir bilgisayar olsun.
- Her komut, belirli bir şekilde kodlanır. Örneğin; İlk iki bit işlem kodunu, sonraki iki bit hedef yazmacı (destination register) ve son dört bit iki işlenen yazmacı temsil eder.
- Örneğin, "0 3 0 2" anlamına gelen "*register 2 ve register 0 içeriğini topla ve sonucu register 3'e kaydet*" komutu, 00110010 şeklinde kodlanabilir.



# von Neumann Mimarisi

- Programlama, farklı soyutlama seviyelerinde gerçekleşebilir.
  - **Yüksek Seviye Dil (High-level language):** Problem alanına daha yakındır. Programcılara daha fazla üretkenlik sağlar. Daha anlaşılabilir ve okunabilir yazılımlar oluşturmaya yardımcı olur.
  - **Çevirici Dili (Assembly language):** Programları metin temelli komutlarla ifade eder. Donanım temsiline çevrilmek üzere kullanılır. Daha düşük seviyeli bir soyutlama sunar.
  - **Donanım Temsili (Hardware Representation):** Sadece ikili sayılar (bitler) ile temsil edilir. Komutlar ve veriler, doğrudan donanım tarafından anlaşılır.



# von Neumann Mimarisi - Assembly

- Çevirici dili, belirli bir bilgisayar mimarisi için özelleştirilmiştir.
- Bir bilgisayarın çevirici dili, başka bir bilgisayarın dilinden farklı olabilir.
- Örnek komutlar:
  - `MOV AL, 61h`: AL yazmacına 61h değerini taşı
  - `MOV AX, BX`: AX yazmacına BX yazmacındaki değeri taşı
  - `ADD EAX, 10`: EAX yazmacına 10 değerini ekle
  - `XOR EAX, EAX`: EAX yazmacını sıfırla
- Bilgisayarın alt seviyede nasıl çalıştığını anlamak için önemlidir.
- Performans optimizasyonu ve donanım ile etkileşim için kullanılır.



SON