# 1. Hafta: Giriş

AUTHOR
Hakan Mehmetcik

# 1. R ve R'ın Temel Yapısı

## R Dilinin Genel Özellikleri ve Kullanımı

R, veri analizi, istatistiksel hesaplamalar ve grafiksel görselleştirme için tasarlanmış güçlü ve açık kaynaklı bir programlama dilidir. R'ın temel avantajlarından biri, akademik ve profesyonel dünyada geniş bir kullanım alanına sahip olmasıdır. R, özellikle büyük veri setleri üzerinde hızlı ve etkili analizler yapmayı sağlar.

## R ile neler yapabilirsiniz?

- Sayısal veriler üzerinde işlem yapabilir, istatistiksel analizler gerçekleştirebilirsiniz.
- Verilerinizi görselleştirmek için grafikler ve çizelgeler oluşturabilirsiniz.
- Gelişmiş istatistiksel modeller kurabilir, veriler üzerinde tahminlerde bulunabilirsiniz.

## R'ın Veri Analizi ve İstatistikteki Yeri

R, özellikle veri bilimciler, istatistikçiler ve araştırmacılar tarafından veri analizi için yaygın olarak kullanılır. Akademik dünyada, bilimsel araştırmalarda ve iş dünyasında veri analizi için standart bir araçtır. Bunun nedeni R'ın geniş bir paket ekosistemine sahip olmasıdır; yani, özel amaçlar için kullanabileceğiniz binlerce hazır fonksiyon ve paket bulunur.

## R'ın Avantajları:

- Esneklik: R, çok farklı veri türleri ve analizlerle çalışmanıza olanak sağlar.
- Geniş Paket Desteği: R, geniş bir topluluğa sahiptir ve hemen hemen her türlü veri analizi için uygun bir paket bulabilirsiniz (örneğin: `ggplot2`, `dplyr`, `tidyverse`). - Veri Görselleştirme: Grafiksel görselleştirme için R, güçlü araçlar sunar. Karmaşık veri setlerini kolayca anlaşılır grafiklere dönüştürebilirsiniz.

## R'ı GitHub Codespaces Üzerinde Kullanmaya Başlama

Bu derste, GitHub Codespaces kullanarak R ile çalışacağız. Codespaces, bulut tabanlı bir geliştirme ortamıdır. Yani, bilgisayarınıza herhangi bir yazılım yüklemeden GitHub üzerinde R projeleri geliştirebilirsiniz. GitHub Codespaces, R dili için hazır bir çalışma ortamı sağlar ve GitHub repository'nizle entegre bir şekilde çalışmanıza olanak tanır.

### R ile GitHub Codespaces'te Çalışmaya Başlama Adımları:

1. Repository'ye Erişin: GitHub'da ders için oluşturulan repository'ye gidin (örneğin: IST2083 GitHub Repository).
2. Codespace'i Açın: Repository sayfasında "Code" butonuna tıklayın ve "Open with Codespaces" seçeneğini seçin.
3. Ortamı Başlatın: Codespace, sizin için bir geliştirme ortamı oluşturacak. Bu ortamda R kodları yazabilir ve çalıştırabilirsiniz.
4. Dosyalarla Çalışma: R dilinde yazılmış `.qmd` (Quarto) dosyalarını açarak ders notlarını inceleyebilir ve R konsolu üzerinde örnek kodları çalıştırabilirsiniz.

### GitHub Codespaces'in Avantajları:

- Kurulum Gerektirmez: Hiçbir yazılım indirip kurmanıza gerek kalmaz, her şey tarayıcı üzerinde çalışır.
- Bulut Tabanlı: Tüm verileriniz ve projeleriniz bulutta saklanır, böylece her yerden erişim sağlayabilirsiniz.
- Anında Kullanıma Hazır: R dili ve ilgili paketler önceden yüklenmiştir; hemen kullanmaya başlayabilirsiniz.

**Sonuç Olarak:** R, veri analizi ve istatistik için en güçlü araçlardan biridir. GitHub Codespaces ise bu dili kullanmayı kolaylaştıran bir platform sunar. Bu derste, GitHub Codespaces kullanarak R ile nasıl çalışılacağını ve temel veri analizi kavramlarını öğrenmeye başlayacaksınız.

---

# 2. R Konsolu ve Temel İşlemler

R, bir programlama dili olmasının yanı sıra, aynı zamanda bir hesap makinesi gibi basit matematiksel işlemler yapabileceğiniz bir ortam sağlar. R konsolunu kullanarak basit komutları hemen çalıştırabilirsiniz. Aşağıda R konsolunda nasıl çalışacağınızı ve temel matematiksel işlemleri nasıl gerçekleştireceğinizi öğreneceksiniz.

## R Konsolunu Kullanarak Basit Komutlar Çalıştırma

GitHub Codespaces'te R konsolunu açarak komutlar yazabilirsiniz. R konsolu, yazdığınız her komutu anında çalıştırır ve size sonucu gösterir. Örneğin, R konsoluna aşağıdaki gibi bir toplama işlemi yazabilirsiniz:

```
3+5
```

```
[1] 8
```

Bu komutu çalıştırdığınızda, R size işlemin sonucunu gösterecektir:

```
[1] 8
```

## Temel Matematiksel İşlemler

R, toplama, çıkarma, çarpma ve bölme gibi temel matematiksel işlemleri doğrudan konsol üzerinde yapabilir.

> **Note**
>
> Bu tür işlemler aynı zamanda temel matematiksel işlemleri gerçekleştirmek için R'da sunulan çeşitli **aritmetik operatörler** olarak adlandırılırlar.

Aşağıda bazı temel matematiksel işlemlerin örnekleri verilmiştir:

```
# Toplama
toplam <- 5 + 3
print(toplam)  # 8
```

```
[1] 8
```

```
# Çıkarma
cikarma <- 5 - 3
print(cikarma)  # 2
```

```
[1] 2
```

```
# Çarpma
carpma <- 5 * 3
print(carpma)  # 15
```

```
[1] 15
```

```
# Bölme
bolme <- 5 / 3
print(bolme)  # 1.666667
```

```
[1] 1.666667
```

```
# Üs Alma
us_alma <- 2^3
print(us_alma)  # 8
```

```
[1] 8
```

```
# Modülüs
modulus <- 5 %% 3
print(modulus)  # 2
```

```
[1] 2
```

```
# Tam Bölme
tam_bolme <- 5 %/% 3
print(tam_bolme)  # 1
```

```
[1] 1
```

## Karşılaştırma Operatörleri

Karşılaştırma operatörleri, iki değeri karşılaştırarak TRUE ya da FALSE değerleri döndürür. R dilinde yaygın olarak kullanılan karşılaştırma operatörleri şunlardır:

```
# Eşittir
print(5 == 3)  # FALSE
```

```
[1] FALSE
```

```
print(5 == 5)  # TRUE
```

```
[1] TRUE
```

```
# Eşit Değildir
print(5 != 3)  # TRUE
```

```
[1] TRUE
```

```
print(5 != 5)  # FALSE
```

```
[1] FALSE
```

```
# Büyüktür
print(5 > 3)  # TRUE
```

```
[1] TRUE
```

```
print(3 > 5)  # FALSE
```

```
[1] FALSE
```

```
# Küçüktür
print(5 < 3)  # FALSE
```

```
[1] FALSE
```

```
print(3 < 5)  # TRUE
```

```
[1] TRUE
```

```
# Büyük Eşittir
print(5 >= 3)  # TRUE
```

```
[1] TRUE
```

```
print(5 >= 5)  # TRUE
```

```
[1] TRUE
```

```
print(3 >= 5)  # FALSE
```

```
[1] FALSE
```

```
# Küçük Eşittir
print(5 <= 3)  # FALSE
```

```
[1] FALSE
```

```
print(5 <= 5)  # TRUE
```

```
[1] TRUE
```

```
print(3 <= 5)  # TRUE
```

```
[1] TRUE
```

## Mantıksal Operatörler

Mantıksal operatörler, R dilinde mantıksal ifadeleri değerlendirmek ve birleştirmek için kullanılır. Bu operatörler, genellikle karşılaştırma operatörleri ile birlikte kullanılarak daha karmaşık koşullar oluşturur. İşte R dilinde yaygın olarak kullanılan mantıksal operatörler: **1. Ve (& ve &&)**

İki mantıksal ifadenin her ikisi de doğruysa TRUE döner. & operatörü vektörler üzerinde eleman bazında çalışır. && operatörü yalnızca ilk elemanları karşılaştırır.

```
TRUE & TRUE   # TRUE
```

```
[1] TRUE
```

```
TRUE & FALSE  # FALSE
```

```
[1] FALSE
```

```
TRUE && TRUE  # TRUE
```

```
[1] TRUE
```

```
TRUE && FALSE # FALSE
```

```
[1] FALSE
```

**2. Veya (| ve ||)**

İki mantıksal ifadeden en az biri doğruysa TRUE döner. | operatörü vektörler üzerinde eleman bazında çalışır. || operatörü yalnızca ilk elemanları karşılaştırır.

```
TRUE | TRUE   # TRUE
```

```
[1] TRUE
```

```
TRUE | FALSE  # TRUE
```

```
[1] TRUE
```

```
FALSE | FALSE # FALSE
```

```
[1] FALSE
```

```
TRUE || TRUE  # TRUE
```

```
[1] TRUE
```

```
TRUE || FALSE # TRUE
```

```
[1] TRUE
```

```
FALSE || FALSE # FALSE
```

```
[1] FALSE
```

**3. Değil (!)**

Bir mantıksal ifadenin tersini döner.

```
!TRUE  # FALSE
```

```
[1] FALSE
```

```
!FALSE # TRUE
```

```
[1] TRUE
```

## Diğer Önemli Operatörler

R dilinde çeşitli operatörler, veri manipülasyonu ve analizinde sıkça kullanılır. İşte R dilinde yaygın olarak kullanılan bazı önemli operatörler:

### Kolon Operatörü ( : )

Kolon operatörü, belirli bir aralıktaki sayıları oluşturmak için kullanılır.

```r
# 1'den 10'a kadar olan sayıları oluşturma
sequence <- 1:10
print(sequence)  # 1 2 3 4 5 6 7 8 9 10
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

## Atama Operatörü ( `<-` ve `=` )

Atama operatörleri, bir değeri bir değişkene atamak için kullanılır.

```r
# Atama operatörleri
x <- 10
y = 20
print(x)  # 10
```

```
[1] 10
```

```r
print(y)  # 20
```

```
[1] 20
```

## Üye Olma Operatörü ( `%in%` )

Üye olma operatörü, bir elemanın bir vektörde olup olmadığını kontrol eder.

```r
# Üye olma operatörü
vec <- c(1, 2, 3, 4, 5)
print(3 %in% vec)  # TRUE
```

```
[1] TRUE
```

```r
print(6 %in% vec)  # FALSE
```

```
[1] FALSE
```

## Dizi Operatörü ( `seq()` )

Dizi operatörü, belirli bir aralıkta ve adım büyüklüğünde sayılar oluşturmak için kullanılır.

```r
# Dizi operatörü
sequence <- seq(1, 10, by = 2)
print(sequence)  # 1 3 5 7 9
```

```
[1] 1 3 5 7 9
```

## Tekrar Operatörü ( `rep()` )

Tekrar operatörü, belirli bir değeri veya vektörü tekrar etmek için kullanılır.

```r
# Tekrar operatörü
repeated <- rep(1:3, times = 3)
```

```
print(repeated)  # 1 2 3 1 2 3 1 2 3
```

```
[1] 1 2 3 1 2 3 1 2 3
```

---

# 3. R'da Mesajlarını Anlama ve Çözme ve Yardım Alma

R ile çalışırken bazen hatalarla karşılaşabilirsiniz. Hatalar, kodunuzda bir sorun olduğunu ve R'ın bunu çalıştıramadığını belirtir. Ancak endişelenmenize gerek yok, bu hatalar çoğunlukla küçük yazım veya kullanım hatalarından kaynaklanır.

**Örnek Bir Hata:

Aşağıdaki komutta `*` yerine `x` yazıldığı için bir hata oluşacaktır:

```
3 x 5
```

Çıktı:

```
Error: unexpected symbol in "3 x"
```

Bu hata, R'ın `x` sembolünü tanımadığını ve bu yüzden işlemi gerçekleştiremediğini belirtir. Bu hatayı düzeltmek için `x` sembolünü `*` ile değiştirebilirsiniz:

```
3 * 5
```

Bu sefer doğru sonuç olan `15` çıktısını elde edeceksiniz.

## Genel Hata Çözme İpuçları:

1. Sembolleri Doğru Kullanın: R'da toplama için `+`, çarpma için `*`, bölme için `/` gibi semboller kullanılır. Yanlış bir sembol kullanırsanız hata alırsınız.
2. Kodunuzda Yazım Hataları Olup Olmadığını Kontrol Edin: R büyük/küçük harfe duyarlıdır, yani `A` ve `a` R için farklı şeyler ifade eder. Yazım hatalarına dikkat edin.
3. Hata Mesajını Dikkatlice Okuyun: R'ın verdiği hata mesajları, sorunun ne olduğunu anlamanıza yardımcı olabilir. Örneğin, yukarıdaki hata mesajı, "unexpected symbol" ifadesiyle bir sembol hatası olduğunu belirtmiştir.

---

# 4. Değişkenler ve Veri Türleri

R dilinde, verileri saklamak ve işlemek için değişkenler kullanılır. Değişkenler, belirli bir değeri tutar ve bu değer üzerinde işlemler yapmanıza olanak tanır.

Değişken Tanımlama (<- Operatörü) R'da bir değişken tanımlamak için <- operatörü kullanılır. Bu operatör, sağdaki değeri soldaki değişkene atar.

```r
x <- 10   # x değişkenine 10 değerini atar
y <- "Merhaba"   # y değişkenine bir karakter dizisi atar
```

## R'da Veri Türleri

R'da değişkenlerin farklı veri türleri olabilir. R'ın sık kullanılan veri türleri şunlardır:

- **Sayısal (Numeric)**: Ondalıklı veya tam sayı olarak saklanan sayılar.

  ```r
  a <- 5.7 # Ondalıklı bir sayısal veri
  b <- 10 # Tam sayı olarak bir sayısal veri
  ```

- **Karakter (Character)**: Metin veya harf dizisi olarak saklanan veriler.

  ```r
  c <- "R programlama dili" # Karakter veri türünde bir veri
  ```

- **Mantıksal (Logical)**: TRUE veya FALSE değerlerini saklayan değişkenler.

  ```r
  durum <- TRUE   # Logical
  ```

## Değişkenlerle Matematiksel İşlemler

Sayısal veri türüne sahip değişkenlerde matematiksel işlemler yapabilirsiniz. Örneğin:

```r
a <- 5
b <- 3
```

### Toplama

```r
toplam <- a + b
print(toplam) # Çıktı: 8
```

```
[1] 8
```

### Çıkarma

```r
cikarma <- a - b
print(cikarma) # Çıktı: 2
```

```
[1] 2
```

### Çarpma

```r
a*b # Çıktı: 15
```

```
[1] 15
```

**Bölme**

```
bolme <- a / b
print(bolme) # Çıktı: 1.666667
```

```
[1] 1.666667
```

**Üs Alma**

```
us <- a^b
print(us) # Çıktı: 125
```

```
[1] 125
```

---

# 5. R'da Temel Fonksiyonlar

R programlama dilinde temel fonksiyonlar, veri analizi ve istatistiksel hesaplamalar için yaygın olarak kullanılır. İşte bazı temel fonksiyonlar ve örnek kullanımları:

R'ın Yerleşik Fonksiyonları Yerleşik fonksiyonlar, R dilinin varsayılan olarak sunduğu işlevlerdir. Örneğin, mean() fonksiyonu bir vektörün ortalamasını hesaplar, sum() toplamını verir ve length() bir vektörün kaç eleman içerdiğini gösterir.

Örnekler: 1. print(): Bir nesneyi konsola yazdırır.

````
::: {.cell}

```{.r .cell-code}
print("Merhaba Dünya")
```

::: {.cell-output .cell-output-stdout}

```
[1] "Merhaba Dünya"
```


:::
:::
````

2. sum(): Bir vektörün elemanlarının toplamını hesaplar.

```
toplam <- sum(c(1, 2, 3, 4, 5))
```

3. mean(): Bir vektörün elemanlarının ortalamasını hesaplar.

```
ortalama <- mean(c(1, 2, 3, 4, 5))
```

4. sd(): Bir vektörün standart sapmasını hesaplar.

```r
standart_sapma <- sd(c(1, 2, 3, 4, 5))
```

5. length(): Bir vektörün uzunluğunu döndürür.

```r
uzunluk <- length(c(1, 2, 3, 4, 5))
```

6. seq(): Belirli bir aralıkta ardışık sayılar oluşturur.

```r
dizi <- seq(1, 10, by=2)
```

7. rep(): Bir değeri belirli sayıda tekrar eder

```r
tekrar <- rep(1, times=5)
```

8. c(): Vektör oluşturur.

```r
vektor <- c(1, 2, 3, 4, 5)
```

9. data.frame(): Veri çerçevesi oluşturur.

```r
veri <- data.frame( isim = c("Ali", "Ayşe", "Fatma"), yas = c(25, 30, 35) )
```

10. summary(): Bir nesnenin özet istatistiklerini döndürür.

```r
ozet <- summary(c(1, 2, 3, 4, 5))
```

## Parametrelerle Fonksiyon Kullanımı

Fonksiyonlar, belirli bir işlevi gerçekleştirmek için parametre alabilir. Parametreler, fonksiyonun nasıl çalışacağını belirler. Örneğin, mean() fonksiyonuna na.rm = TRUE parametresi eklenerek eksik değerleri göz ardı edebilirsiniz.

```r
sayilar <- c(10, 20, NA, 40, 50)
ortalama <- mean(sayilar, na.rm = TRUE)   # NA değerini göz ardı ederek ortalama hesap
```

## Fonksiyon Oluşturma

R dilinde kendi fonksiyonlarınızı oluşturmak oldukça basittir. Fonksiyonlar, belirli bir işlevi gerçekleştirmek için bir dizi komut içerir ve gerektiğinde parametreler alabilir. Fonksiyonlar, function anahtar kelimesi kullanılarak tanımlanır.

Fonksiyon Oluşturma Bir fonksiyon oluşturmak için aşağıdaki adımları izleyebilirsiniz:

- function anahtar kelimesini kullanarak fonksiyonunuzu tanımlayın.
- Fonksiyonun alacağı parametreleri parantez içinde belirtin.
- Fonksiyonun gövdesinde, fonksiyonun gerçekleştireceği işlemleri yazın.
- return() fonksiyonunu kullanarak fonksiyonun döndüreceği değeri belirtin (isteğe bağlı).

```r
# Toplama fonksiyonu oluşturma
topla <- function(a, b) {
  sonuc <- a + b
  return(sonuc)
}

# Fonksiyonu çağırma
toplam <- topla(5, 3)
print(toplam)  # 8
```

```
[1] 8
```

## R Fonksiyonları İçin Yardım Alma ( ? ve help() Komutları)

Bir R fonksiyonu hakkında bilgi almak için ? veya help() komutlarını kullanabilirsiniz. Bu komutlar, ilgili fonksiyonun nasıl kullanılacağı ve parametrelerinin ne olduğu hakkında detaylı bilgi verir.

**Örnekler**:

```r
?mean   # mean() fonksiyonu hakkında yardım alır
help(sum)  # sum() fonksiyonu hakkında yardım alır
```

Bu komutlar, fonksiyonun açıklamasını, hangi argümanları kabul ettiğini ve nasıl çalıştırılacağını gösteren bir dokümantasyon penceresi açar.

## Belgeleme ve Fonksiyonların Anlamı

R'ın tüm fonksiyonlarının nasıl çalıştığını, örnekler ve açıklamalarla birlikte görebilirsiniz. R'da her fonksiyonun geniş bir açıklaması bulunur ve bu belgeleme, fonksiyonların nasıl kullanılacağını öğrenmenin etkili bir yoludur.

**Örnek**:

```r
help(mean)  # Ortalama hesaplayan mean() fonksiyonunu anlamak için yardım alabiliriz
```

---

# 6. Basit Veri Yapıları

R dilinde, veriler çeşitli veri yapıları kullanılarak organize edilir. Temel veri yapıları, vektörler, listeler ve matrislerdir.

## Vektörler

R dilinde en temel veri yapılarından biri vektörlerdir. Vektörler, aynı veri türüne sahip bir dizi elemanı saklar. Vektör oluşturmak için c() fonksiyonu kullanılır.

```r
# Sayısal vektör oluşturma
numeric_vector <- c(1, 2, 3, 4, 5)

# Karakter vektör oluşturma
character_vector <- c("a", "b", "c", "d")

# Mantıksal vektör oluşturma
logical_vector <- c(TRUE, FALSE, TRUE)

# Vektör elemanlarına erişim
print(numeric_vector[1])  # 1
```

```
[1] 1
```

```r
print(character_vector[3])  # "c"
```

```
[1] "c"
```

```r
# Vektör elemanlarını güncelleme
numeric_vector[2] <- 10

# Vektör toplama
sum_vector <- numeric_vector + c(1, 1, 1, 1, 1)

# Vektör elemanlarının toplamı
total_sum <- sum(numeric_vector)

# Güncellenmiş vektörleri yazdırma
print(numeric_vector)
```

```
[1]  1 10  3  4  5
```

```r
print(sum_vector)
```

```
[1]  2 11  4  5  6
```

```r
print(total_sum)
```

```
[1] 23
```

## Listeler

Listeler, farklı türdeki verileri bir arada saklayabilen veri yapılarıdır. Bir liste içinde sayılar, karakterler, mantıksal değerler ve hatta başka vektörler bulunabilir.

```r
# Liste oluşturma
my_list <- list(number = 5, greeting = "Hello", flag = TRUE, numbers = c(1, 2, 3))
```

```r
# Liste elemanlarına erişim
print(my_list$number)  # 5
```

```
[1] 5
```

```r
print(my_list$greeting)  # "Hello"
```

```
[1] "Hello"
```

```r
print(my_list$flag)  # TRUE
```

```
[1] TRUE
```

```r
print(my_list$numbers)  # c(1, 2, 3)
```

```
[1] 1 2 3
```

```r
# Liste elemanlarını güncelleme
my_list$number <- 10
my_list$greeting <- "Hi"

# Listeye yeni eleman ekleme
my_list$new_element <- "New Element"

# Liste elemanlarını silme
my_list$new_element <- NULL

# Güncellenmiş listeyi yazdırma
print(my_list)
```

```
$number
[1] 10

$greeting
[1] "Hi"

$flag
[1] TRUE

$numbers
[1] 1 2 3
```

## Matrisler

Matrisler, satırlar ve sütunlar şeklinde organize edilmiş sayısal veri yapılarıdır. Tüm elemanlar aynı türde olmalıdır.

```r
# 2x3 boyutunda bir matris oluşturma
my_matrix <- matrix(1:6, nrow = 2, ncol = 3)
```

```
# Matris elemanlarına erişim
print(my_matrix[1, 2])  # 3
```

```
[1] 3
```

```
print(my_matrix[2, ])  # c(4, 5, 6)
```

```
[1] 2 4 6
```

```
print(my_matrix[, 3])  # c(5, 6)
```

```
[1] 5 6
```

```
# Matris elemanlarını güncelleme
my_matrix[1, 2] <- 10

# Matris toplama
matrix1 <- matrix(1:4, nrow = 2)
matrix2 <- matrix(5:8, nrow = 2)
sum_matrix <- matrix1 + matrix2

# Matris çarpma
product_matrix <- matrix1 %*% matrix2

# Güncellenmiş matrisleri yazdırma
print(my_matrix)
```

```
     [,1] [,2] [,3]
[1,]    1   10    5
[2,]    2    4    6
```

```
print(sum_matrix)
```

```
     [,1] [,2]
[1,]    6   10
[2,]    8   12
```

```
print(product_matrix)
```

```
     [,1] [,2]
[1,]   23   31
[2,]   34   46
```

## Data Frameler

Data frameler, R dilinde kullanılan ve farklı veri türlerine sahip sütunları içerebilen iki boyutlu veri yapılarıdır. Data frameler, genellikle tablo şeklinde verileri saklamak için kullanılır ve her sütun bir vektör olarak temsil edilir. Data frameler, data.frame() fonksiyonu kullanılarak oluşturulur.

```r
# Data frame oluşturma
my_data_frame <- data.frame(
  numbers = c(1, 2, 3, 4, 5),
  letters = c("a", "b", "c", "d", "e"),
  logicals = c(TRUE, FALSE, TRUE, FALSE, TRUE)
)

# Data frame elemanlarına erişim
print(my_data_frame[1, 2])  # "a"
```

```
[1] a
Levels: a b c d e
```

```r
print(my_data_frame[2, ])  # c(2, "b", FALSE)
```

```
  numbers letters logicals
2       2       b    FALSE
```

```r
print(my_data_frame[, 3])  # c(TRUE, FALSE, TRUE, FALSE, TRUE)
```

```
[1]  TRUE FALSE  TRUE FALSE  TRUE
```

```r
print(my_data_frame$numbers)  # c(1, 2, 3, 4, 5)
```

```
[1] 1 2 3 4 5
```

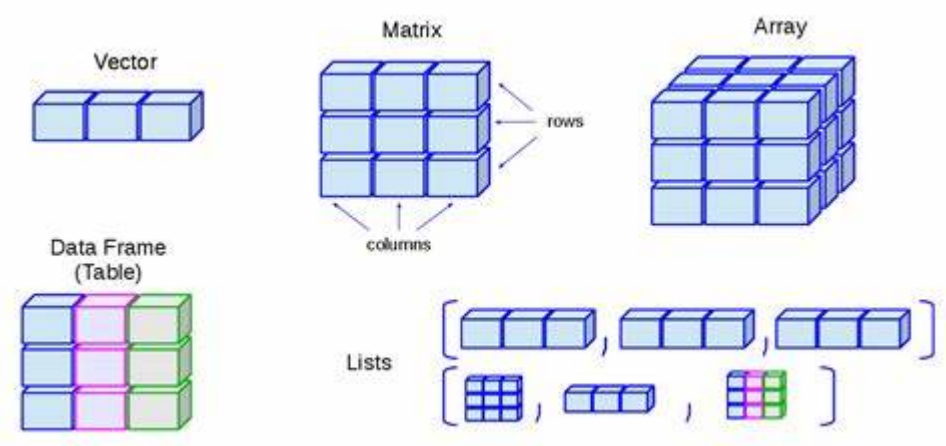
```r
# Data frame elemanlarını güncelleme
my_data_frame[1, 2] <- "z"
```

```
Warning in `[<-.factor`(`*tmp*`, iseq, value = "z"): invalid factor level, NA
generated
```

```r
my_data_frame$numbers[2] <- 10

# Yeni bir sütun ekleme
my_data_frame$new_column <- c(10, 20, 30, 40, 50)

# Satır ekleme
new_row <- data.frame(numbers = 6, letters = "f", logicals = FALSE, new_column = 60)
my_data_frame <- rbind(my_data_frame, new_row)

# Sütun ekleme
new_column <- c(100, 200, 300, 400, 500, 600)
my_data_frame <- cbind(my_data_frame, new_column)

# Güncellenmiş data frame'i yazdırma
print(my_data_frame)
```

```
  numbers letters logicals new_column new_column
1       1    <NA>     TRUE         10        100
2      10       b    FALSE         20        200
```

```
3       3       c      TRUE          30          300
4       4       d     FALSE          40          400
5       5       e      TRUE          50          500
6       6       f     FALSE          60          600
```




# 7. Veri İçe Aktarma ve Kaydetme

Veri içe aktarma ve temel manipülasyon, veri analizi sürecinin önemli adımlarıdır. R dilinde veri içe aktarma ve temel manipülasyon işlemleri için çeşitli fonksiyonlar ve paketler kullanılır.

## Veri İçe Aktarma

R dilinde veri içe aktarmak için read.csv(), read.table(), read_excel() gibi fonksiyonlar kullanılır.

```
# CSV dosyasını içe aktarma
data <- read.csv("https://raw.githubusercontent.com/datasciencedojo/datasets/refs/head
```

```
# readxl paketini yükleme
install.packages("readxl")
```

```
Installing package into '/home/codespace/R/x86_64-pc-linux-gnu-library/3.6'
(as 'lib' is unspecified)
```

```
library(readxl)

# Excel dosyasını içe aktarma
# data <- read_excel("https://github.com/HakanMehmetcik/IST2083/blob/main/Data/sample_
```

## Veri Kaydetme

R dilinde veri kaydetmek için çeşitli fonksiyonlar kullanabilirsiniz. En yaygın kullanılan dosya formatları arasında CSV, Excel ve RDS dosyaları bulunur.

**CSV Dosyasına Veri Kaydetme** Veri çerçevesini CSV dosyasına kaydetmek için write.csv() fonksiyonunu kullanabilirsiniz:

```
# Veri çerçevesini CSV dosyasına kaydetme
# write.csv(data, "path/to/your/file.csv", row.names = FALSE)
```

**Excel Dosyasına Veri Kaydetme** Excel dosyasına veri kaydetmek için writexl paketini kullanabilirsiniz:

```
# writexl paketini yükleme
install.packages("writexl")
```

```
Installing package into '/home/codespace/R/x86_64-pc-linux-gnu-library/3.6'
(as 'lib' is unspecified)
```

```
library(writexl)

# Veri çerçevesini Excel dosyasına kaydetme
# write_xlsx(data, "path/to/your/file.xlsx")
```

## Veri Yapısını İnceleme

Veri inceleme, veri analizi sürecinin önemli bir parçasıdır. R dilinde veri incelemek için çeşitli fonksiyonlar ve paketler kullanabilirsiniz. Bu işlemler arasında veri yapısını kontrol etme, özet istatistikler oluşturma ve veri görselleştirme gibi işlemler bulunur. Veri yapısını incelemek için str(), summary(), head(), ve tail() fonksiyonlarını kullanabilirsiniz.

```
# Veri yapısını kontrol etme
str(data)
```

```
'data.frame':   891 obs. of  12 variables:
 $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
 $ Name       : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559
520 629 417 581 ...
 $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
 $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
 $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
 $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
 $ Ticket     : Factor w/ 681 levels "110152","110413",..: 524 597 670 50 473 276 86
396 345 133 ...
 $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
 $ Cabin      : Factor w/ 148 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...
 $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```
# Özet istatistikler
summary(data)
```

```
  PassengerId       Survived          Pclass
 Min.   :  1.0   Min.   :0.0000   Min.   :1.000
 1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000
```

```
 Median :446.0   Median :0.0000   Median :3.000
 Mean   :446.0   Mean   :0.3838   Mean   :2.309
 3rd Qu.:668.5   3rd Qu.:1.0000   3rd Qu.:3.000
 Max.   :891.0   Max.   :1.0000   Max.   :3.000

                                     Name         Sex           Age
 Abbing, Mr. Anthony                   :  1   female:314   Min.   : 0.42
 Abbott, Mr. Rossmore Edward           :  1   male  :577   1st Qu.:20.12
 Abbott, Mrs. Stanton (Rosa Hunt)      :  1                Median :28.00
 Abelson, Mr. Samuel                   :  1                Mean   :29.70
 Abelson, Mrs. Samuel (Hannah Wizosky):  1                3rd Qu.:38.00
 Adahl, Mr. Mauritz Nils Martin        :  1                Max.   :80.00
 (Other)                               :885                NA's   :177
     SibSp           Parch             Ticket         Fare
 Min.   :0.000   Min.   :0.0000   1601    :  7   Min.   :  0.00
 1st Qu.:0.000   1st Qu.:0.0000   347082  :  7   1st Qu.:  7.91
 Median :0.000   Median :0.0000   CA. 2343:  7   Median : 14.45
 Mean   :0.523   Mean   :0.3816   3101295 :  6   Mean   : 32.20
 3rd Qu.:1.000   3rd Qu.:0.0000   347088  :  6   3rd Qu.: 31.00
 Max.   :8.000   Max.   :6.0000   CA 2144 :  6   Max.   :512.33
                                  (Other) :852
         Cabin     Embarked
            :687    :  2
 B96 B98    :  4   C:168
 C23 C25 C27:  4   Q: 77
 G6         :  4   S:644
 C22 C26    :  3
 D          :  3
 (Other)    :186
```

```r
# İlk birkaç satırı görüntüleme
head(data)
```

```
  PassengerId Survived Pclass
1           1        0      3
2           2        1      1
3           3        1      3
4           4        1      1
5           5        0      3
6           6        0      3
                                                 Name    Sex Age SibSp Parch
1                             Braund, Mr. Owen Harris   male  22     1     0
2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
3                              Heikkinen, Miss. Laina female  26     0     0
4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
5                            Allen, Mr. William Henry   male  35     0     0
6                                    Moran, Mr. James   male  NA     0     0
            Ticket    Fare Cabin Embarked
1        A/5 21171  7.2500              S
2         PC 17599 71.2833   C85        C
3 STON/O2. 3101282  7.9250              S
4           113803 53.1000  C123        S
```

```
5           373450  8.0500              S
6           330877  8.4583              Q
```

```r
# Son birkaç satırı görüntüleme
tail(data)
```

```
    PassengerId Survived Pclass                                      Name    Sex
886         886        0      3     Rice, Mrs. William (Margaret Norton) female
887         887        0      2                     Montvila, Rev. Juozas   male
888         888        1      1              Graham, Miss. Margaret Edith female
889         889        0      3 Johnston, Miss. Catherine Helen "Carrie" female
890         890        1      1                     Behr, Mr. Karl Howell   male
891         891        0      3                       Dooley, Mr. Patrick   male
    Age SibSp Parch     Ticket   Fare Cabin Embarked
886  39     0     5     382652 29.125              Q
887  27     0     0     211536 13.000              S
888  19     0     0     112053 30.000   B42        S
889  NA     1     2 W./C. 6607 23.450              S
890  26     0     0     111369 30.000  C148        C
891  32     0     0     370376  7.750              Q
```

---