

BAB 5

MATRIKS

5.1 Membuat Matriks dengan Fungsi *matrix()*

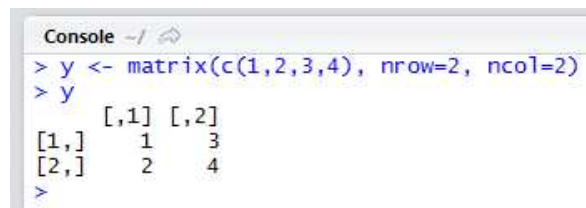
Norman Matloff (2009) menyatakan sebagai berikut.

“Matrix is a vector with two additional attributes, the number of rows and number of columns.

Multidimensional vectors in R are called arrays. A two-dimensional array is also called a matrix, and is eligible for the usual matrix mathematical operations.

*Matrix row and column subscripts begin with 1, so for instance the upper-left corner of the matrix **a** is denoted **a[1,1]**. The internal linear storage of a matrix is in column-major-order, meaning that first all of column 1 is stored, then all of column 2, etc.”*

Berdasarkan uraian di atas dapat ditarik informasi bahwa suatu matriks merupakan suatu vektor dengan dua atribut tambahan (*two additional attributes*), yakni jumlah baris dan jumlah kolom. Vektor multidimensi (*multidimensional vectors*) dalam R disebut *arrays*. Array dua dimensi (*two-dimensional array*) juga disebut matriks. Perhatikan Gambar 5.1.



```
Console -/ ^
> y <- matrix(c(1,2,3,4), nrow=2, ncol=2)
> y
      [,1] [,2]
[1,]    1    3
[2,]    2    4
>
```

Gambar 5.1

Pada Gambar 5.1 digunakan fungsi *matrix()* untuk membuat matriks. Perintah R `y <- matrix(c(1,2,3,4), nrow=2, ncol=2)` dapat diartikan membuat suatu matriks dengan nama “y”, jumlah baris sebanyak 2, dan jumlah kolom sebanyak 2. Elemen-elemen pada matriks y adalah 1, 3, 2, dan 4. Diketahui:

- ⇒ Nilai 1 menempati posisi baris ke-1 dan kolom ke-1.
- ⇒ Nilai 2 menempati posisi baris ke-2 dan kolom ke-1.
- ⇒ Nilai 3 menempati posisi baris ke-1 dan kolom ke-2.
- ⇒ Nilai 4 menempati posisi baris ke-2 dan kolom ke-2.

Perintah R `y[1,1]` berarti menampilkan nilai pada matriks `y`, pada posisi baris ke-1 dan kolom ke-1, yakni 1. Perintah R `y[1,2]` berarti menampilkan nilai pada matriks `y`, pada posisi baris ke-1 dan kolom ke-2, yakni 3, dan seterusnya. Perhatikan Gambar 5.2.

```
> y <- matrix(c(1,2,3,4), nrow=2, ncol=2)
> y
      [,1] [,2]
[1,]    1    3
[2,]    2    4
> y[1,1]
[1] 1
> y[1,2]
[1] 3
>
```

Gambar 5.2

Pada Gambar 5.3, perintah R `z <- matrix(c(1,2,3,4,5,6), nrow=2)` dapat diartikan membuat suatu matriks bernama `z`, yang jumlah barisnya sebanyak 2. Elemen-elemen pada matriks `z` adalah 1, 2, 3, 4, 5, dan 6. Perhatikan bahwa:

- ⇒ Nilai 1 menempati posisi baris ke-1 dan kolom ke-1.
- ⇒ Nilai 2 menempati posisi baris ke-2 dan kolom ke-1.
- ⇒ Nilai 3 menempati posisi baris ke-1 dan kolom ke-2.
- ⇒ Nilai 4 menempati posisi baris ke-2 dan kolom ke-2.
- ⇒ Nilai 5 menempati posisi baris ke-1 dan kolom ke-3.
- ⇒ Nilai 6 menempati posisi baris ke-2 dan kolom ke-3.

```
Console ~/
> z <- matrix(c(1,2,3,4,5,6), nrow=2)
> z
      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
>
```

Gambar 5.3

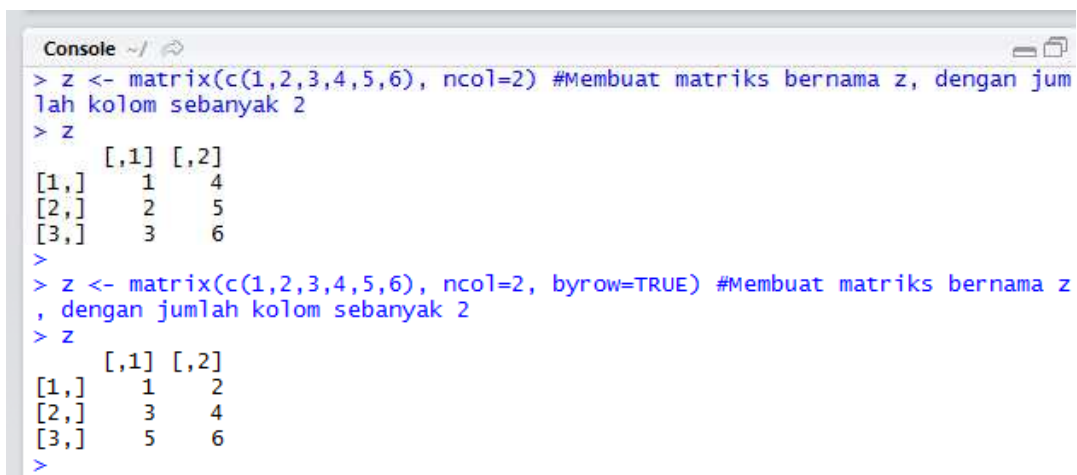
```
Console ~/
> z <- matrix(c(1,2,3,4,5,6), ncol=2)
> z
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> |
```

Gambar 5.4

Pada Gambar 5.4, perintah R `z <- matrix(c(1,2,3,4,5,6), ncol=2)` dapat diartikan membentuk suatu matriks bernama **z**, yang jumlah kolomnya sebanyak 2. Elemen-elemen pada matriks **z** adalah 1, 2, 3, 4, 5, dan 6. Perhatikan bahwa:

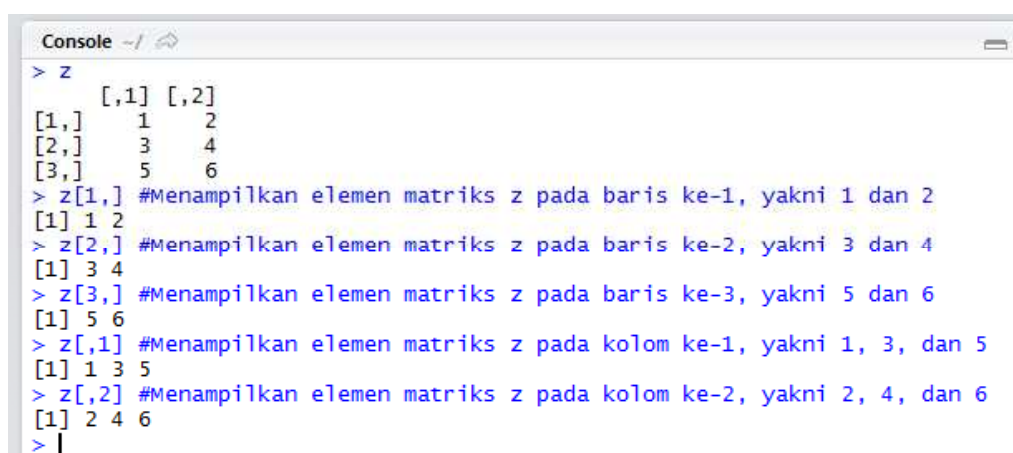
- ⇒ Nilai 1 menempati posisi baris ke-1 dan kolom ke-1.
- ⇒ Nilai 2 menempati posisi baris ke-2 dan kolom ke-1.
- ⇒ Nilai 3 menempati posisi baris ke-3 dan kolom ke-1.
- ⇒ Nilai 4 menempati posisi baris ke-1 dan kolom ke-2.
- ⇒ Nilai 5 menempati posisi baris ke-2 dan kolom ke-2.
- ⇒ Nilai 6 menempati posisi baris ke-3 dan kolom ke-2.

Namun perhatikan Gambar 5.5 dan Gambar 5.6.



```
Console ~/ /
> z <- matrix(c(1,2,3,4,5,6), ncol=2) #Membuat matriks bernama z, dengan jumlah kolom sebanyak 2
> z
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
>
> z <- matrix(c(1,2,3,4,5,6), ncol=2, byrow=TRUE) #Membuat matriks bernama z, dengan jumlah kolom sebanyak 2
> z
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
>
```

Gambar 5.5



```
Console ~/ /
> z
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
> z[1,] #Menampilkan elemen matriks z pada baris ke-1, yakni 1 dan 2
[1] 1 2
> z[2,] #Menampilkan elemen matriks z pada baris ke-2, yakni 3 dan 4
[1] 3 4
> z[3,] #Menampilkan elemen matriks z pada baris ke-3, yakni 5 dan 6
[1] 5 6
> z[,1] #Menampilkan elemen matriks z pada kolom ke-1, yakni 1, 3, dan 5
[1] 1 3 5
> z[,2] #Menampilkan elemen matriks z pada kolom ke-2, yakni 2, 4, dan 6
[1] 2 4 6
> |
```

Gambar 5.6

Berdasarkan Gambar 5.6, pada perintah R `z <- matrix(x(1,2,3,4,5,6), ncol=2, byrow=TRUE)` berarti membuat matriks bernama **z** dengan jumlah kolom sebanyak 2. Perhatikan bahwa:

- ⇒ Nilai 1 menempati posisi baris ke-1 dan kolom ke-1.
- ⇒ Nilai 2 menempati posisi baris ke-1 dan kolom ke-2.
- ⇒ Nilai 3 menempati posisi baris ke-2 dan kolom ke-1.
- ⇒ Nilai 4 menempati posisi baris ke-2 dan kolom ke-2.
- ⇒ Nilai 5 menempati posisi baris ke-3 dan kolom ke-1.
- ⇒ Nilai 6 menempati posisi baris ke-3 dan kolom ke-2.

Pada Gambar 5.6, perintah `Rz[1,]` berarti menampilkan elemen-elemen dari matriks **z** pada baris ke-1, yakni 1 dan 2. Perintah `R z[2,]` berarti menampilkan elemen-elemen dari matriks **z** pada baris ke-2, yakni 3 dan 4. Perintah `Rz[,1]` berarti menampilkan elemen-elemen dari matriks **z** pada kolom ke-1, yakni 1, 3, dan 5.

5.2 Menugaskan Bilangan ke Matriks

Perhatikan Gambar 5.7. Pada Gambar 5.7 perintah `R a <- matrix(nrow=3, ncol=2)` berarti membentuk matriks bernama **a**, dengan jumlah baris sebanyak 3 dan jumlah kolom sebanyak 2. Pada awalnya, elemen dari matriks **a** adalah NA (*not available*).

```

Console ~/ ↵
> a <- matrix(nrow=3, ncol=2)
> a
      [,1] [,2]
[1,]   NA   NA
[2,]   NA   NA
[3,]   NA   NA
> a[1,1] <- 1
> a[2,1] <- 2
> a[3,1] <- 3
> a[1,2] <- 4
> a[2,2] <- 5
> a[3,2] <- 6
> a
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> |

```

Gambar 5.7

Perhatikan bahwa:

- ⇒ Perintah `R a[1,1] <- 1` berarti menugaskan bilangan 1 ke matriks **a** pada baris ke-1, kolom ke-1.
- ⇒ Perintah `R a[2,1] <- 2` berarti menugaskan bilangan 2 ke matriks **a** pada baris ke-2, kolom ke-1.

- ⇒ Perintah R `a[3,1] <- 3` berarti menugaskan bilangan 3 ke matriks `a` pada baris ke-3, kolom ke-1.
- ⇒ Perintah R `a[1,2] <- 4` berarti menugaskan bilangan 4 ke matriks `a` pada baris ke-1, kolom ke-2.
- ⇒ Perintah R `a[2,2] <- 5` berarti menugaskan bilangan 5 ke matriks `a` pada baris ke-2, kolom ke-2.
- ⇒ Perintah R `a[3,2] <- 6` berarti menugaskan bilangan 6 ke matriks `a` pada baris ke-3, kolom ke-2.

5.3 Mengganti Elemen Matriks

```

Console ~1 ↻
> a
  [,1] [,2]
[1,]  1   4
[2,]  2   5
[3,]  3   6
> a[2,2]
[1] 5
> a[2,2] <- 1000 #Mengubah elemen dari matriks a pada baris ke-2, kolom ke-2 dengan
bilangan 1000
> a[2,2]
[1] 1000
> a
  [,1] [,2]
[1,]  1   4
[2,]  2 1000
[3,]  3   6
> |

```

Gambar 5.8

Pada Gambar 5.8, misalkan elemen matriks `a` pada baris ke-2 dan kolom ke-2, yakni nilai 5, ingin diganti menjadi 1000. Perintah R untuk mengganti elemen matriks `a` pada baris ke-2 dan kolom ke-2, yakni nilai 5, menjadi 1000 adalah `a[2,2] <- 1000`.

5.4 Berbagai Cara Menampilkan Elemen Matriks

Gambar 5.9 menampilkan berbagai cara untuk menampilkan elemen matriks. Berdasarkan Gambar 5.9, perintah `RA[,c(1)]` berarti menampilkan elemen-elemen matriks `A` pada kolom ke-1, yakni 1, 2, dan 3. Perintah `RA[,-c(2:4)]` berarti menampilkan elemen-elemen matriks `A`, selain dari pada kolom ke-2 sampai ke-4. Dengan kata lain, berarti menampilkan elemen-elemen matriks `A` pada kolom ke-1, yakni 1, 2, dan 3. Perintah `RA[,c(1,4)]` berarti menampilkan elemen-elemen matriks `A` pada kolom ke-1 dan ke-4. Perintah `RA[,c(2:4)]` berarti menampilkan elemen-elemen matriks `A` pada kolom ke-2 sampai ke-4. Perintah `RA[c(1,3),]` berarti menampilkan elemen-elemen matriks `A` pada baris ke-1 dan ke-3.

```

Console ~/
> A <- matrix(c(1,2,3,4,5,6,7,8,9,10,11,12), nrow=3)
> A
      [,1] [,2] [,3] [,4]
[1,]  1   4   7  10
[2,]  2   5   8  11
[3,]  3   6   9  12
> A[,c(1)] #Menampilkan elemen-elemen matriks A pada kolom ke-1
[1] 1 2 3
> A[,-c(2:4)] #Menampilkan elemen-elemen matriks A, selain dari kolom ke-2 sampai kolom ke-4
[1] 1 2 3
> A[,c(1,2)] #Menampilkan elemen-elemen matriks A pada kolom ke-1 dan ke-2
      [,1] [,2]
[1,]  1   4
[2,]  2   5
[3,]  3   6
> A[,c(1,4)] #Menampilkan elemen-elemen matriks A pada kolom ke-1 dan ke-4
      [,1] [,2]
[1,]  1  10
[2,]  2  11
[3,]  3  12
> A[,c(2:4)] #Menampilkan elemen-elemen matriks A dari kolom ke-2 sampai ke-4
      [,1] [,2] [,3]
[1,]  4   7  10
[2,]  5   8  11
[3,]  6   9  12
> A[c(1),] #Menampilkan elemen-elemen matriks A pada baris ke-1
[1] 1 4 7 10
> A[c(1,3),] #Menampilkan elemen-elemen matriks A pada baris ke-1 dan ke-3
      [,1] [,2] [,3] [,4]
[1,]  1   4   7  10
[2,]  3   6   9  12
> A[c(2:3),] #Menampilkan elemen-elemen matriks A pada baris ke-2 sampai ke-3
      [,1] [,2] [,3] [,4]
[1,]  2   5   8  11
[2,]  3   6   9  12
> A[-c(2:3),] #Menampilkan elemen-elemen matriks A, selain dari baris ke-2 sampai ke-3
[1] 1 4 7 10
>

```

Gambar 5.9

5.5 Menghapus Baris dan Kolom pada Matriks

Perhatikan Gambar 5.10. Pada perintah R `A <- A[-c(2),]` bertujuan untuk menghapus baris ke-2 dari matriks A. Sementara perintah R `A <- A[, -c(4)]` bertujuan menghapus kolom ke-4 dari matriks A.

```

> A <- matrix(c(1,2,3,4,5,6,7,8,9,10,11,12), nrow=3)
> A
      [,1] [,2] [,3] [,4]
[1,]  1   4   7  10
[2,]  2   5   8  11
[3,]  3   6   9  12
> A <- A[-c(2),] #Menghapus baris ke-2 dari matriks A
> A
      [,1] [,2] [,3] [,4]
[1,]  1   4   7  10
[2,]  3   6   9  12
> A <- A[, -c(4)] #Menghapus kolom ke-4 dari matriks A
> A
      [,1] [,2] [,3]
[1,]  1   4   7
[2,]  3   6   9
>

```

Gambar 5.10

5.6 Mengganti Elemen Matriks (Bagian 2)

Pada Gambar 5.11 dan Gambar 5.12 diberikan berbagai contoh perintah R untuk mengganti elemen matriks. Berdasarkan Gambar 5.11, diberikan matriks **B**, **C**, dan **D**. Diketahui digunakan fungsi *rbind*(

) untuk membuat matriks **C**, sementara fungsi *cbind()* digunakan untuk membuat matriks **D**. Berdasarkan Gambar 5.11, Perintah `B[,1] <- C[,2]` berarti mengganti elemen-elemen matriks **B** pada kolom ke-1, dengan elemen-elemen matriks **C** pada kolom ke-2. Perintah `B[,1] <- D[,1]` berarti mengganti elemen-elemen matriks **B** pada kolom ke-1, dengan elemen-elemen matriks **D** pada kolom ke-1. Perintah `B[,c(2,3)] <- C` berarti mengganti elemen-elemen matriks **B** pada kolom ke-2 dan ke-3, dengan elemen-elemen pada matriks **C**. Perintah `B[c(1),] <- D[c(2),]` berarti mengganti elemen-elemen matriks **B** pada baris ke-1, dengan elemen-elemen pada matriks **D**, baris ke-2. Namun **memberikan informasi kesalahan**

*Error in B[c(1),] = D[c(2),] :
number of items to replace is not a multiple of replacement length*

Hal ini karena jumlah elemen dari matriks **B** pada baris ke-1 sebanyak 3, sementara jumlah elemen dari matriks **D** pada baris ke-2 sebanyak 2. Selanjutnya, perintah `B[c(1),] <- D[c(2),]` berarti mengganti elemen-elemen matriks **C** pada baris ke-1, dengan elemen-elemen pada matriks **D**, baris ke-2.

```

Console ~/1 ↗
> B <- matrix(c(1,2,3,4,5,6), nrow=2)
> B
      [,1] [,2] [,3]
[1,]  1   3   5
[2,]  2   4   6
> C <- rbind(c(5,2), c(1,4))
> C
      [,1] [,2]
[1,]  5   2
[2,]  1   4
> D <- cbind(c(5,2), c(1,4))
> D
      [,1] [,2]
[1,]  5   1
[2,]  2   4
> B[,1] <- C[,2] #Mengganti elemen-elemen matriks B pada kolom ke-1, dengan elemen-elemen matriks C pada kolom ke-2
> B
      [,1] [,2] [,3]
[1,]  2   3   5
[2,]  4   4   6
> B[,1] <- D[,1] #Mengganti elemen-elemen matriks B pada kolom ke-1, dengan elemen-elemen matriks D pada kolom ke-1
> B
      [,1] [,2] [,3]
[1,]  5   3   5
[2,]  2   4   6
> B[,c(2,3)] <- C #Mengganti elemen-elemen matriks B pada kolom ke-2 dan ke-3, dengan elemen-elemen pada matriks C
> B
      [,1] [,2] [,3]
[1,]  5   5   2
[2,]  2   1   4

```

Gambar 5.11

```

[1,] 5 3 5
[2,] 2 4 6
> B[,c(2,3)] <- c #Mengganti elemen-elemen matriks B pada kolom ke-2 dan ke-3, dengan elemen-elemen pada matriks c
> B
  [,1] [,2] [,3]
[1,] 5 5 2
[2,] 2 1 4
> B[c(1),] <- D[c(2), ] #Mengganti elemen-elemen matriks B pada baris ke-1, dengan elemen-elemen pada matriks D, baris ke-2
Error in B[c(1), ] <- D[c(2), ] :
  number of items to replace is not a multiple of replacement length
> B
  [,1] [,2] [,3]
[1,] 5 5 2
[2,] 2 1 4
> B[,c(2,3)] <- c #Mengganti elemen-elemen matriks B pada kolom ke-2 dan ke-3, dengan elemen-elemen pada matriks D
> B
  [,1] [,2] [,3]
[1,] 5 5 2
[2,] 2 1 4
> B[c(1),] <- D[c(2), ] #Mengganti elemen-elemen matriks B pada baris ke-1, dengan elemen-elemen matriks D, baris ke-2
Error in B[c(1), ] <- D[c(2), ] :
  number of items to replace is not a multiple of replacement length
> B
  [,1] [,2] [,3]
[1,] 5 5 2
[2,] 2 1 4
> C
  [,1] [,2]
[1,] 5 2
[2,] 1 4
> C[c(1),] <- D[c(2), ] #Mengganti elemen-elemen matriks C pada baris ke-1, dengan elemen-elemen pada matriks D, baris ke-2
> C
  [,1] [,2]
[1,] 2 4
[2,] 1 4
>

```

Gambar 5.12

5.7 Operasi Matematika pada Matriks (Bagian 1)

Perhatikan Gambar 5.13. Perintah R $B+C$ berarti

$$B = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}, C = \begin{bmatrix} 2 & 4 \\ 1 & 1 \end{bmatrix}, B + C = \begin{bmatrix} 1+2 & 3+4 \\ 2+1 & 2+1 \end{bmatrix} = \begin{bmatrix} 3 & 7 \\ 3 & 3 \end{bmatrix}$$

```

Console ~/ |
> A <- matrix(c(1,2,3,4,2,1), nrow=2)
> B <- matrix(c(1,2,3,2), nrow=2)
> C <- matrix(c(2,1,4,1), nrow=2)
> A
  [,1] [,2] [,3]
[1,] 1 3 2
[2,] 2 4 1
> B
  [,1] [,2]
[1,] 1 3
[2,] 2 2
> C
  [,1] [,2]
[1,] 2 4
[2,] 1 1
> B+C
  [,1] [,2]
[1,] 3 7
[2,] 3 3
> |

```

Gambar 5.13

Perintah R $A+B$ pada Gambar 5.14 memberi pesan kesalahan dikarenakan matriks A terdiri dari 2 baris dan 3 kolom, sementara matriks B terdiri dari 2 baris dan 2 kolom.


```

Console ~/
> A
  [,1] [,2] [,3]
[1,]  1   3   2
[2,]  2   4   1
> B
  [,1] [,2]
[1,]  1   3
[2,]  2   2
> C
  [,1] [,2]
[1,]  2   4
[2,]  1   1
> A+B
Error in A + B : non-conformable arrays
> |

```

Gambar 5.14

Perintah R **B/C** pada Gambar 5.15 berarti

$$B = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}, C = \begin{bmatrix} 2 & 4 \\ 1 & 1 \end{bmatrix}, B/C = \begin{bmatrix} 1/2 & 3/4 \\ 2/1 & 2/1 \end{bmatrix} = \begin{bmatrix} 0,5 & 0,75 \\ 2 & 2 \end{bmatrix}$$

```

Console ~/
> A
  [,1] [,2] [,3]
[1,]  1   3   2
[2,]  2   4   1
> B
  [,1] [,2]
[1,]  1   3
[2,]  2   2
> C
  [,1] [,2]
[1,]  2   4
[2,]  1   1
> B/C
  [,1] [,2]
[1,] 0.5 0.75
[2,] 2.0 2.00
> |

```

Gambar 5.15

Perintah R **B%%C** pada Gambar 5.16 berarti

$$B = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}, C = \begin{bmatrix} 2 & 4 \\ 1 & 1 \end{bmatrix}, B%%C = \begin{bmatrix} 1%%2 & 3%%4 \\ 2%%2 & 2%%1 \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ 0 & 0 \end{bmatrix}$$

```

Console ~/
> A
  [,1] [,2] [,3]
[1,]  1   3   2
[2,]  2   4   1
> B
  [,1] [,2]
[1,]  1   3
[2,]  2   2
> C
  [,1] [,2]
[1,]  2   4
[2,]  1   1
> B%%C
  [,1] [,2]
[1,]  1   3
[2,]  0   0
> |

```

Gambar 5.16

Perintah R $B^{\wedge}C$ pada Gambar 5.17 berarti

$$B = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}, C = \begin{bmatrix} 2 & 4 \\ 1 & 1 \end{bmatrix}, B^{\wedge}C = \begin{bmatrix} 1^2 & 3^4 \\ 2^1 & 2^1 \end{bmatrix} = \begin{bmatrix} 1 & 81 \\ 2 & 2 \end{bmatrix}$$

```

Console ~| ↻
> A
     [,1] [,2] [,3]
[1,]    1    3    2
[2,]    2    4    1
> B
     [,1] [,2]
[1,]    1    3
[2,]    2    2
> C
     [,1] [,2]
[1,]    2    4
[2,]    1    1
> B^C
     [,1] [,2]
[1,]    1   81
[2,]    2    2
> |
  
```

Gambar 5.17

Perintah R $B\%*\%C$ pada Gambar 5.18 berarti

$$B = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}, C = \begin{bmatrix} 2 & 4 \\ 1 & 1 \end{bmatrix}, B\%*\%C = \begin{bmatrix} (1 \times 2) + (3 \times 1) & (1 \times 4) + (3 \times 1) \\ (2 \times 2) + (2 \times 1) & (2 \times 4) + (2 \times 1) \end{bmatrix} = \begin{bmatrix} 5 & 7 \\ 6 & 10 \end{bmatrix}$$

```

Console ~| ↻
> A
     [,1] [,2] [,3]
[1,]    1    3    2
[2,]    2    4    1
> B
     [,1] [,2]
[1,]    1    3
[2,]    2    2
> C
     [,1] [,2]
[1,]    2    4
[2,]    1    1
> B*C
     [,1] [,2]
[1,]    5    7
[2,]    6   10
> |
  
```

Gambar 5.18

5.8 Fungsi $rowMeans()$, $colMeans()$, $rowSums()$, dan $colSums()$

Perhatikan Gambar 5.19. Perintah R $rowMeans(A)$ berarti

$$\frac{1 + 3 + 2}{3}; \frac{2 + 4 + 1}{3} = 2; 2,333$$

Perintah R $colMeans(A)$ berarti

$$\frac{1 + 2}{2}; \frac{3 + 4}{2}; \frac{2 + 1}{2} = 1,5; 3,5; 1,5$$

```

Console ~/
> A
  [,1] [,2] [,3]
[1,]  1   3   2
[2,]  2   4   1
> rowMeans(A)
[1] 2.000000 2.333333
> colMeans(A)
[1] 1.5 3.5 1.5
> rowSums(A)
[1] 6 7
> colSums(A)
[1] 3 7 3
>

```

Gambar 5.19

Perintah R **rowSums(A)** berarti

$$1 + 3 + 2 = 6$$

$$2 + 4 + 1 = 7$$

Perintah R **colSums(A)** berarti

$$1 + 2 = 3$$

$$3 + 4 = 7$$

$$2 + 1 = 3$$

5.9 Transpose Matriks

Gambar 5.20 memperlihatkan penggunaan fungsi $t()$ untuk menentukan *transpose* suatu matriks.

$$A = \begin{bmatrix} 1 & 3 & 2 \\ 2 & 4 & 1 \end{bmatrix}; A^T = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 2 & 1 \end{bmatrix}$$

```

Console ~/
> A
  [,1] [,2] [,3]
[1,]  1   3   2
[2,]  2   4   1
> t(A)
  [,1] [,2]
[1,]  1   2
[2,]  3   4
[3,]  2   1
> |

```

Gambar 5.20

5.10 Determinan Matriks

Gambar 5.21 memperlihatkan penggunaan fungsi $det()$ untuk menentukan determinan suatu matriks.

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \det(A) = (4 \times 1) - (2 \times 3) = -2$$

```

Console ~/
> A
      [,1] [,2]
[1,]    1    2
[2,]    3    4
> det(A)
[1] -2
>

```

Gambar 5.21

5.11 Invers Matriks

Gambar 5.22 memperlihatkan penggunaan fungsi *solve()* untuk menentukan *invers* suatu matriks.

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, A^{-1} = \frac{1}{(4 \times 1) - (3 \times 2)} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix} = \frac{1}{-2} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 1,5 & -0,5 \end{bmatrix}$$

```

Console ~/
> A <- matrix(c(1,2,3,4), ncol=2, byrow=TRUE)
> A
      [,1] [,2]
[1,]    1    2
[2,]    3    4
> solve(A)
      [,1] [,2]
[1,] -2.0  1.0
[2,]  1.5 -0.5
>

```

Gambar 5.22

5.12 Fungsi *eigen()* untuk Menentukan Nilai Eigen dan Vektor Eigen dari Matriks

Matriks persegi (*square matrix*) A dikatakan memiliki nilai eigen λ , dengan vektor eigen-nya yang bersesuaian (*eigenvector*) $x \neq 0$, jika

$$Ax = \lambda x$$

Misalkan diberikan matriks

$$A = \begin{bmatrix} 1 & -5 \\ -5 & 1 \end{bmatrix}$$

Gambar 5.23 memperlihatkan nilai eigen dan vektor eigen dari matriks A. Perhatikan bahwa

$$\begin{bmatrix} 1 & -5 \\ -5 & 1 \end{bmatrix} \begin{bmatrix} -0.7071068 \\ 0.7071068 \end{bmatrix} = 6 \begin{bmatrix} -0.7071068 \\ 0.7071068 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -5 \\ -5 & 1 \end{bmatrix} \begin{bmatrix} -0.7071068 \\ -0.7071068 \end{bmatrix} = -4 \begin{bmatrix} -0.7071068 \\ -0.7071068 \end{bmatrix}$$

Sehingga matriks A memiliki 2 pasang nilai eigen dan vektor eigen, yakni

$$6, \begin{bmatrix} -0.7071068 \\ 0.7071068 \end{bmatrix} \text{ dan } -4, \begin{bmatrix} -0.7071068 \\ -0.7071068 \end{bmatrix}$$

```

Console ~/ / ↵
> A
  [,1] [,2]
[1,]  1  -5
[2,] -5   1
> simpan <- eigen(A)
> simpan
$values
[1] 6 -4

$vectors
      [,1]      [,2]
[1,] -0.7071068 -0.7071068
[2,]  0.7071068 -0.7071068

> eigenvalue1 <- simpan$values[1]
> eigenvalue2 <- simpan$values[2]
> eigenvector1 <- simpan$vectors[,1]
> eigenvector2 <- simpan$vectors[,2]
> eigenvalue1
[1] 6
> eigenvalue2
[1] -4
> eigenvector1
[1] -0.7071068  0.7071068
> eigenvector2
[1] -0.7071068 -0.7071068
> |

```

Gambar 5.23

```

> A%%eigenvector1
      [,1]
[1,] -4.242641
[2,]  4.242641
> eigenvalue1*eigenvector1
[1] -4.242641  4.242641
> A%%eigenvector2
      [,1]
[1,]  2.828427
[2,]  2.828427
> eigenvalue2*eigenvector2
[1]  2.828427  2.828427
> |

```

Gambar 5.24

5.13 Eliminasi Gauss-Jordan untuk Menentukan Solusi dari Sistem Persamaan Linear

Misalkan diberikan sistem persamaan linear sebagai berikut.

$$2x + 4y + 6z = 20$$

$$4x + 6y + 2z = 26$$

$$6x - 2y + 4z = 18$$

Gambar 5.25 dan Gambar 5.26 diberikan ilustrasi dalam R terkait eliminasi Gauss-Jordan, untuk memperoleh solusi dari sistem persamaan linear di atas.

```

> A <- matrix(c(2, 4, 6, 20, 4, 6, 2, 26, 6, -2, 4, 18), ncol=4, byrow=TRUE)
> A
      [,1] [,2] [,3] [,4]
[1,]    2    4    6   20
[2,]    4    6    2   26
[3,]    6   -2    4   18
> A[2,] <- A[2,]-2*A[1,]
> A
      [,1] [,2] [,3] [,4]
[1,]    2    4    6   20
[2,]    0   -2  -10  -14
[3,]    6   -2    4   18
> A[3,] <- A[3,]-3*A[1,]
> A
      [,1] [,2] [,3] [,4]
[1,]    2    4    6   20
[2,]    0   -2  -10  -14
[3,]    0  -14  -14  -42
> A[3,] <- A[3,]-7*A[2,]
> A
      [,1] [,2] [,3] [,4]
[1,]    2    4    6   20
[2,]    0   -2  -10  -14
[3,]    0    0   56   56
> A[1,] <- A[1,]+2*A[2,]
> A
      [,1] [,2] [,3] [,4]
[1,]    2    0  -14   -8
[2,]    0   -2  -10  -14
[3,]    0    0   56   56
> A[1,] <- 4*A[1,]+A[3,]
> A
      [,1] [,2] [,3] [,4]
[1,]    8    0    0   24
[2,]    0   -2  -10  -14
[3,]    0    0   56   56

```

Gambar 5.25

```

> A[2,] <- 56*A[2,]+10*A[3,]
> A
      [,1] [,2] [,3] [,4]
[1,]    8    0    0   24
[2,]    0 -112    0 -224
[3,]    0    0   56   56
> A[1,] <- (1/8)*A[1,]
> A
      [,1] [,2] [,3] [,4]
[1,]    1    0    0    3
[2,]    0 -112    0 -224
[3,]    0    0   56   56
> A[2,] <- (1/112)*A[2,]
> A
      [,1] [,2] [,3] [,4]
[1,]    1    0    0    3
[2,]    0  -1    0   -2
[3,]    0    0   56   56
> A[2,] <- -1*A[2,]
> A
      [,1] [,2] [,3] [,4]
[1,]    1    0    0    3
[2,]    0    1    0    2
[3,]    0    0   56   56
> A[3,] <- (1/56)*A[3,]
> A
      [,1] [,2] [,3] [,4]
[1,]    1    0    0    3
[2,]    0    1    0    2
[3,]    0    0    1    1
> |

```

Gambar 5.26

Berdasarkan Gambar 5.26, diperoleh solusi $x = 3$, $y = 2$, dan $z = 1$. Perhatikan bahwa

$$2x + 4y + 6z = 2(3) + 4(2) + 6(1) = 20$$

$$4x + 6y + 2z = 4(3) + 6(2) + 2(1) = 26$$

$$6x - 2y + 4z = 6(3) - 2(2) + 4(1) = 18$$

5.14 Fungsi `rref()` untuk Menentukan Solusi dari Sistem Persamaan Linear

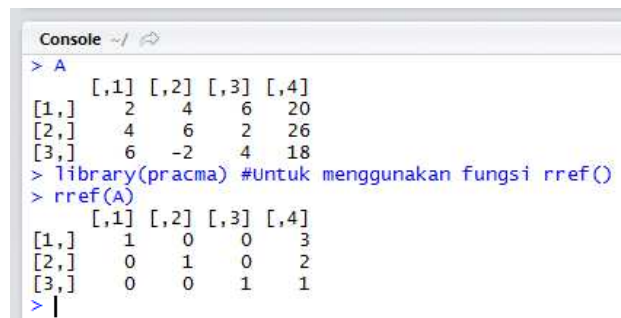
Misalkan diberikan sistem persamaan linear sebagai berikut.

$$2x + 4y + 6z = 20$$

$$4x + 6y + 2z = 26$$

$$6x - 2y + 4z = 18$$

Gambar 5.27 diperlihatkan ilustrasi dalam R penggunaan fungsi `rref()` untuk menentukan solusi dari sistem persamaan di atas. Diketahui solusi dari sistem persamaan linear tersebut adalah $x = 3, y = 2,$ dan $z = 1$.



```

Console ~/\
> A
      [,1] [,2] [,3] [,4]
[1,]    2    4    6    20
[2,]    4    6    2    26
[3,]    6   -2    4    18
> library(pracma) #Untuk menggunakan fungsi rref()
> rref(A)
      [,1] [,2] [,3] [,4]
[1,]    1    0    0     3
[2,]    0    1    0     2
[3,]    0    0    1     1
> |

```

Gambar 5.27

5.15 Fungsi `LUsplit()` dan `lu.decomposition()` untuk Dekomposisi LU

Misalkan diberikan matriks A sebagai berikut.

$$A = \begin{bmatrix} 4 & 12 & 8 & 4 \\ 1 & 7 & 18 & 9 \\ 2 & 9 & 20 & 20 \\ 3 & 11 & 15 & 14 \end{bmatrix}$$

Gambar 5.28 diberikan ilustrasi dalam R penggunaan fungsi `LUsplit()` untuk memperoleh matriks segitiga bawah L dan matriks segitiga atas U , yang memenuhi $LU = A$.

```

Console ~/
> A
  [,1] [,2] [,3] [,4]
[1,]  4  12   8   4
[2,]  1   7  18   9
[3,]  2   9  20  20
[4,]  3  11  15  14
> library(optR)
> Z <- optR(A, method="LU")
> LUSplit(Z$U)
$U
  [,1] [,2] [,3] [,4]
[1,]  4  12   8   4
[2,]  0   2   9  11
[3,]  0   0  -2 -14
[4,]  0   0   0 -16
$L
  [,1] [,2] [,3] [,4]
[1,] 1.00 0.0 0.00  0
[2,] 0.75 1.0 0.00  0
[3,] 0.25 2.0 1.00  0
[4,] 0.50 1.5 -1.25  1
> simpan$L %% simpan$U
  [,1] [,2] [,3] [,4]
[1,]  4  12   8   4
[2,]  3  11  15  14
[3,]  1   7  18   9
[4,]  2   9  20  20
>

Console ~/
> A
  [,1] [,2] [,3] [,4]
[1,]  4  12   8   4
[2,]  1   7  18   9
[3,]  2   9  20  20
[4,]  3  11  15  14
> library(matrixcalc)
> lu.decomposition(A)
$L
  [,1] [,2] [,3] [,4]
[1,] 1.00 0.00 0.00  0
[2,] 0.25 1.00 0.00  0
[3,] 0.50 0.75 1.00  0
[4,] 0.75 0.50 0.25  1
$U
  [,1] [,2] [,3] [,4]
[1,]  4  12   8   4
[2,]  0   4  16   8
[3,]  0   0   4  12
[4,]  0   0   0   4

```

Gambar 5.28

5.16 Fungsi *chol()* untuk Dekomposisi *Cholesky*

Misalkan diberikan matriks simetri A dengan dimensi 3×3 .

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Karena A matriks simetri, maka berlaku

$$A = A^T$$

Sehingga

$$a_{12} = a_{21}$$

$$a_{13} = a_{31}$$

$$a_{23} = a_{32}$$

Sebagai contoh misalkan diberikan matriks simetri A sebagai berikut.

$$A = \begin{bmatrix} 9 & -3 & 6 \\ -3 & 17 & -10 \\ 6 & -10 & 12 \end{bmatrix}$$

Perhatikan bahwa

$$A = A^T$$
$$a_{12} = -3 = a_{21}$$
$$a_{13} = 6 = a_{31}$$
$$a_{23} = -10 = a_{32}$$

Matriks simetri A dapat dinyatakan sebagai perkalian matriks segitiga bawah L dan matriks segitiga atas U , di mana berlaku

$$L = U^T$$

Sehingga

$$A = LU$$
$$A = U^T U$$

Pada Gambar 5.29 diberikan ilustrasi dalam R penggunaan fungsi `chol()` untuk menentukan matriks U^T dan U , sehingga memenuhi $U^T U = A$.

```
Console ~/ ↻
> A <- rbind(c(9,-3,6), c(-3,17,-10), c(6,-10,12))
> A
      [,1] [,2] [,3]
[1,]  9   -3   6
[2,] -3   17 -10
[3,]  6  -10  12
> chol(A)
      [,1] [,2] [,3]
[1,]  3   -1   2
[2,]  0    4  -2
[3,]  0    0   2
> u <- chol(A)
> U_transpose <- t(u)
> U
      [,1] [,2] [,3]
[1,]  3   -1   2
[2,]  0    4  -2
[3,]  0    0   2
> U_transpose
      [,1] [,2] [,3]
[1,]  3    0   0
[2,] -1    4   0
[3,]  2   -2   2
> U_transpose %% U
      [,1] [,2] [,3]
[1,]  9   -3   6
[2,] -3   17 -10
[3,]  6  -10  12
> |
```

Gambar 5.29

5.17 Fungsi `qr()` untuk Dekomposisi QR

Misalkan diberikan matriks A sebagai berikut.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Gambar 5.31 diberikan ilustrasi dalam R penggunaan fungsi `qr()` terkait dekomposisi QR, untuk memperoleh matriks Q dan R, sehingga memenuhi $QR = A$. Berdasarkan Gambar 5.31, diperoleh

$$Q = \begin{bmatrix} -0,57735 & 0,81650 & -0,000000000000000055511 \\ -0,57735 & -0,40825 & -0,70711 \\ -0,57735 & -0,40825 & 0,70711 \end{bmatrix}$$

$$R = \begin{bmatrix} -1,7321 & -1,1547 & -0,57735 \\ 0 & -0,8165 & -0,40825 \\ 0 & 0 & 0,70711 \end{bmatrix}$$

$$QR = A$$

$$Q^T Q = I$$

```

> A <- rbind(c(1,0,0), c(1,1,0), c(1,1,1))
> A
  [,1] [,2] [,3]
[1,]  1   0   0
[2,]  1   1   0
[3,]  1   1   1
> y <- qr(A)
> Q <- qr.Q(y)
> R <- qr.R(y)
> y
$qr
      [,1] [,2] [,3]
[1,] -1.73205 -1.15470 -0.57735
[2,]  0.57735 -0.81650 -0.40825
[3,]  0.57735  0.70711  0.70711

$rank
[1] 3

$qraux
[1] 1.57735 1.70711 0.70711

$pivot
[1] 1 2 3

attr(,"class")
[1] "qr"
> Q
      [,1] [,2] [,3]
[1,] -0.57735  0.81650 -5.5511e-17
[2,] -0.57735 -0.40825 -7.0711e-01
[3,] -0.57735 -0.40825  7.0711e-01
> R
      [,1] [,2] [,3]
[1,] -1.7321 -1.1547 -0.57735
[2,]  0.0000 -0.8165 -0.40825
[3,]  0.0000  0.0000  0.70711
> Q %>% R
      [,1] [,2] [,3]
[1,]  1 1.1102e-16 1.6259e-17
[2,]  1 1.0000e+00 0.0000e+00
[3,]  1 1.0000e+00 1.0000e+00
> t(Q) %>% Q
      [,1] [,2] [,3]
[1,] 1.0000e+00 -1.6653e-16  0.0000e+00
[2,] -1.6653e-16  1.0000e+00 -1.1102e-16
[3,]  0.0000e+00 -1.1102e-16  1.0000e+00
> |

```

Gambar 5.30

5.18 Memberi Nama Pada Kolom dan Baris dengan Fungsi *colnames()* dan *rownames()*

Gambar 5.31 diberikan ilustrasi dalam R penggunaan fungsi *colnames()* dan *rownames()* untuk memberi nama pada kolom dan baris.

```
Console ~/ |
> A <- rbind(c(1,2,3), c(4,5,6))
> A
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> colnames(A)
NULL
> rownames(A)
NULL
> colnames(A) <- c("Kolom Pertama", "Kolom Kedua", "Kolom Ketiga")
> A
      Kolom Pertama Kolom Kedua Kolom Ketiga
[1,]             1           2           3
[2,]             4           5           6
> rownames(A) <- c("Baris Pertama", "Baris Kedua")
> A
      Kolom Pertama Kolom Kedua Kolom Ketiga
Baris Pertama      1           2           3
Baris Kedua        4           5           6
> rownames(A)
[1] "Baris Pertama" "Baris Kedua"
> A[,1]
Baris Pertama      1
Baris Kedua        4
> A[1,]
Kolom Pertama      1
Kolom Kedua        2
Kolom Ketiga       3

Console ~/ |
> A["Baris Pertama",]
Kolom Pertama      1 Kolom Kedua      2 Kolom Ketiga      3
> A["Baris Kedua",]
Kolom Pertama      4 Kolom Kedua      5 Kolom Ketiga      6
> A["kolom kedua"]
Baris Pertama      2
Baris Kedua        5
> A["Baris Pertama", "kolom ketiga"]
[1] 3
> A["Baris Kedua", "kolom ketiga"]
[1] 6
>

Console ~/ |
> A
      Kolom Pertama Kolom Kedua Kolom Ketiga
Baris Pertama      1           2           3
Baris Kedua        4           5           6
> rowMeans(A)
Baris Pertama      2
Baris Kedua        5
> colMeans(A)
Kolom Pertama      2.5
Kolom Kedua        3.5
Kolom Ketiga       4.5
> rowSums(A)
Baris Pertama      6
Baris Kedua       15
> colSums(A)
Kolom Pertama      5
Kolom Kedua        7
Kolom Ketiga       9
> |
```

Gambar 5.31

5.19 Fungsi *apply()* untuk Menampilkan Berbagai Ukuran Deskriptif berdasarkan Baris atau Kolom dari Matriks

Gambar 5.32 diberikan ilustrasi dalam R penggunaan *apply()* untuk menampilkan berbagai ukuran deskriptif berdasarkan baris atau kolom dari matriks. Berdasarkan Gambar 5.32, diketahui:

- ⇒ Perintah R **`apply(A, 1, sum)`** #Jumlah dari masing-masing baris (nilai 1) berarti menjumlahkan bilangan-bilangan dari matriks A berdasarkan baris, yakni $1 + 2 + 3 = 6$ dan $4 + 5 + 6 = 15$. Pada perintah R **`apply(A, 1, sum)`**, angka “1” tersebut memerintahkan perhitungan berdasarkan baris.
- ⇒ Perintah R **`apply(A, 2, sum)`** #Jumlah dari masing-masing kolom (nilai 2) berarti menjumlahkan bilangan-bilangan dari matriks A berdasarkan kolom, yakni $1 + 4 = 5$, $2 +$

$5 = 7$ dan $3 + 6 = 9$. Pada perintah R `apply(A, 2, sum)`, angka “2” tersebut memerintahkan perhitungan berdasarkan kolom.

⇒ Perintah R `apply(A, sum)` memberikan pesan kesalahan, begitu juga dengan perintah R `apply(A, 3, sum)`.

```
Console ~1 ↻
> A <- rbind(c(1,2,3), c(4,5,6))
> A
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
> apply(A, 1, sum) #Jumlah dari masing-masing baris (nilai 1)
[1]  6 15
> apply(A, 2, sum) #Jumlah dari masing-masing kolom (nilai 2)
[1]  5  7  9
> apply(A, sum) #Memberi pesan kesalahan
Error in match.fun(FUN) : argument "FUN" is missing, with no default
> apply(A, 3, sum) #Memberi pesan kesalahan
Error in if (d2 == 0L) { : missing value where TRUE/FALSE needed
> apply(A, 1, mean) #Rata-rata dari masing-masing baris (nilai 1)
[1]  2  5
> apply(A, 2, mean) #Rata-rata dari masing-masing kolom (nilai 2)
[1]  2.5 3.5 4.5
> apply(A, 1, max) #Maksimum dari masing-masing baris (nilai 1)
[1]  3  6
> apply(A, 2, max) #Maksimum dari masing-masing kolom (nilai 2)
[1]  4  5  6
> apply(A, 1, min) #Minimum dari masing-masing baris (nilai 1)
[1]  1  4
> apply(A, 2, min) #Minimum dari masing-masing kolom (nilai 2)
[1]  1  2  3
> apply(A, 1, sd) #Standar deviasi dari masing-masing baris (nilai 1)
[1]  1  1
> apply(A, 2, sd) #Standar deviasi dari masing-masing kolom (nilai 2)
[1]  2.1213 2.1213 2.1213
> apply(A, 1, function(x) length(x)) #Jumlah elemen dari masing-masing baris (nilai 1)
[1]  3  3
> apply(A, 2, function(x) length(x)) #Jumlah elemen dari masing-masing kolom (nilai 2)
[1]  2  2  2
> |
```

Gambar 5.32

Perhatikan Gambar 5.33. Perintah `RA <- matrix(nrow = 4, ncol = 5)` berarti membuat matriks bernama A dengan jumlah baris sebanyak 4 dan jumlah kolom sebanyak 5. Selanjutnya menampilkan elemen dari matriks A, yang ternyata seluruhnya adalah “NA” atau “*not available* (tidak tersedia)”. Selanjutnya:

⇒ Perintah R `A[1,1] <- 2` berarti menugaskan bilangan 2 ke matriks A pada posisi baris ke-1 dan kolom ke-1.

⇒ Perintah R `A[1,3] <- 5` berarti menugaskan bilangan 5 ke matriks A pada posisi baris ke-1 dan kolom ke-3, dan seterusnya.

```

Console
> A <- matrix(nrow = 4, ncol = 5)
> A
      [,1] [,2] [,3] [,4] [,5]
[1,]   NA   NA   NA   NA   NA
[2,]   NA   NA   NA   NA   NA
[3,]   NA   NA   NA   NA   NA
[4,]   NA   NA   NA   NA   NA
> A[1,1] <- 2
> A
      [,1] [,2] [,3] [,4] [,5]
[1,]     2   NA   NA   NA   NA
[2,]   NA   NA   NA   NA   NA
[3,]   NA   NA   NA   NA   NA
[4,]   NA   NA   NA   NA   NA
> A[1,3] <- 5
> A
      [,1] [,2] [,3] [,4] [,5]
[1,]     2   NA     5   NA   NA
[2,]   NA   NA   NA   NA   NA
[3,]   NA   NA   NA   NA   NA
[4,]   NA   NA   NA   NA   NA
> A[2,2] <- 3
> A
      [,1] [,2] [,3] [,4] [,5]
[1,]     2   NA     5   NA   NA
[2,]   NA     3   NA   NA   NA
[3,]   NA   NA   NA   NA   NA
[4,]   NA   NA   NA   NA   NA
> A[1,4] <- 6
> A[1,5] <- 8
> A[2,4] <- 1
> A[3,1] <- 2
> A[3,4] <- 1
> A[3,3] <- 2
> A[4,1] <- 5
> A[4,5] <- 2
> A
      [,1] [,2] [,3] [,4] [,5]
[1,]     2   NA     5     6     8
[2,]   NA     3   NA     1   NA
[3,]     2   NA     2     1   NA
[4,]     5   NA   NA     NA     2
> rowSums(A)
[1] NA NA NA NA
> rowSums(A, na.rm=TRUE)
[1] 21  4  5  7
> colSums(A)
[1] NA NA NA NA NA
> colSums(A, na.rm=TRUE)
[1]  9  3  7  8 10
> apply(A, 1, sum) #Jumlah dari masing-masing baris (nilai 1)
[1] NA NA NA NA
> apply(A, 1, sum, na.rm=T) #Jumlah dari masing-masing baris (nilai 1)
[1] 21  4  5  7
> apply(A, 1, function(x) length(x), na.rm=T) #Jumlah elemen dari masing-masing baris (nilai 1)
Error in FUN(newX[, i], ...) : unused argument (na.rm = TRUE)
> apply(A, 1, function(x) length(which(!is.na(x)))) #Jumlah elemen dari masing-masing baris (nilai 1)
[1] 4 2 3 2
>
> apply(A, 2, function(x) length(x)) #Jumlah elemen dari masing-masing kolom (nilai 2)
[1] 4 4 4 4 4
> apply(A, 2, function(x) length(x), na.rm=T) #Jumlah elemen dari masing-masing kolom (nilai 2)
Error in FUN(newX[, i], ...) : unused argument (na.rm = TRUE)
> apply(A, 2, function(x) length(which(!is.na(x)))) #Jumlah elemen dari masing-masing kolom (nilai 2)
[1] 3 1 2 3 2
>

```

Gambar 5.33

- ⇒ Perintah R **rowSums(A)** bertujuan untuk menjumlahkan bilangan-bilangan berdasarkan baris. Namun hasil dari perintah **rowSums(A)** adalah NA NA NA NA.
- ⇒ Selanjutnya perintah R yang sebelumnya **rowSums(A)** diganti menjadi **rowSums(A, na.rm=TRUE)**. Perintah R **rowSums(A, na.rm=TRUE)** memberikan hasil 21 4 5 7.

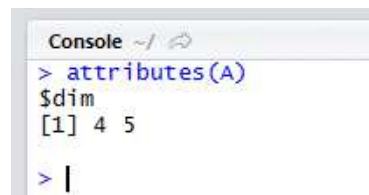
$$2 + 5 + 6 + 8 = 21$$

$$3 + 1 = 4$$

$$2 + 2 + 1 = 5$$

$$5 + 2 = 7$$

- ⇒ Perintah R **apply(A, 1, function(x) length(which(!is.na(x))))** memberikan hasil 4 2 3 2. Nilai 4 berarti banyaknya bilangan pada baris pertama sebanyak 4, yakni 2, 5, 6, dan 8. Nilai 2 berarti banyaknya bilangan pada baris kedua sebanyak 2, yakni 3 dan 1.
- ⇒ Pada Gambar 5.34 perintah R **attributes(A)** memberikan informasi dimensi dari matriks A, yakni terdiri dari 4 baris dan 5 kolom.



```
Console ~ | ↻
> attributes(A)
$dim
[1] 4 5
> |
```

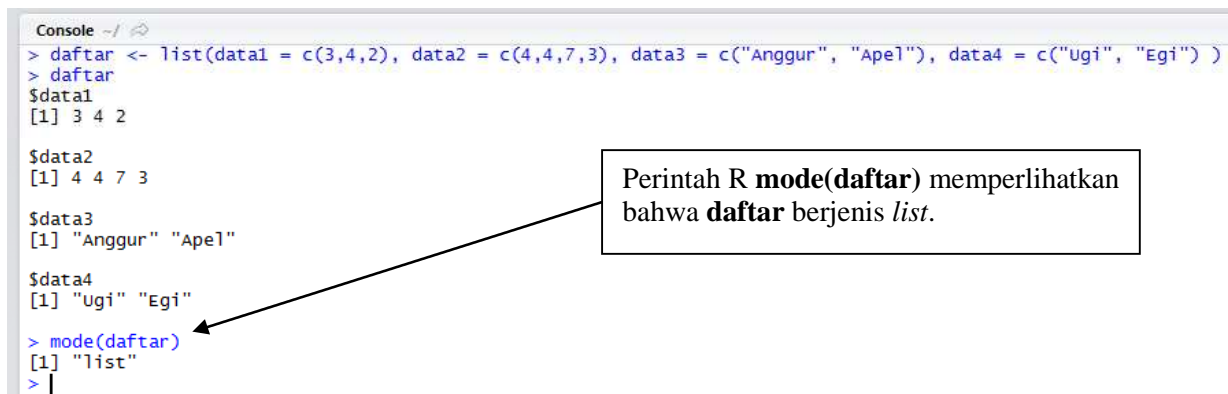
Gambar 5.34

BAB 6

LIST

6.1 Membuat *List* dengan Fungsi `list()`

Dalam R, fungsi `list()` digunakan untuk membuat suatu *list*. Gambar 6.1 diberikan contoh membuat *list* bernama **daftar**. Perhatikan bahwa pada *list* yang bernama **daftar** terdiri dari 4 anggota *list* (*list members*) atau 4 vektor, yakni **data1**, **data2**, **data3**, dan **data4**. Elemen dari vektor **data1** adalah 3, 4, dan 2. Elemen dari vektor **data2** adalah 4, 4, 7, dan 3, dan seterusnya. Perintah R `mode(daftar)` memperlihatkan bahwa **daftar** berjenis *list*.



```
Console ~/ |
> daftar <- list(data1 = c(3,4,2), data2 = c(4,4,7,3), data3 = c("Anggur", "Apel"), data4 = c("Ugi", "Egi") )
> daftar
$data1
[1] 3 4 2

$data2
[1] 4 4 7 3

$data3
[1] "Anggur" "Apel"

$data4
[1] "Ugi" "Egi"

> mode(daftar)
[1] "list"
> |
```

Perintah R `mode(daftar)` memperlihatkan bahwa **daftar** berjenis *list*.

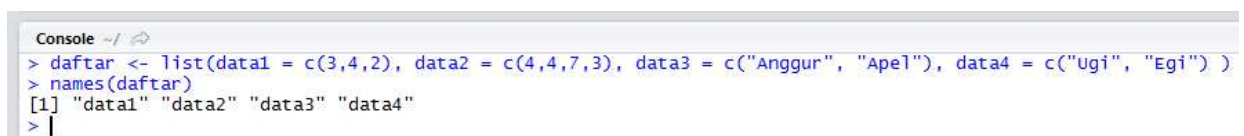
Gambar 6.1

6.2 Memeroleh Nama Anggota *List* (*Tags*) dengan Fungsi `names()`

Norman Matloff dalam bukunya yang berjudul “*The Art of R Programming*” menyatakan sebagai berikut (2009).

“If the elements in a list do have names, e.g. with **name**, **salary** and **union** for *j* above, these names are called **tags**. The value associated with a tag is indeed called its **value**. You can obtain the tags via `names()`.”

Berdasarkan Gambar 6.1, **data1**, **data2**, **data3**, dan **data4** merupakan nama-nama dari anggota *list*. Pada Gambar 6.2, perintah R `names(daftar)` bertujuan untuk mengetahui nama-nama anggota dari *list* **daftar**.



```
Console ~/ |
> daftar <- list(data1 = c(3,4,2), data2 = c(4,4,7,3), data3 = c("Anggur", "Apel"), data4 = c("Ugi", "Egi") )
> names(daftar)
[1] "data1" "data2" "data3" "data4"
> |
```

Gambar 6.2

6.3 Mengetahui Jumlah Anggota List dengan Fungsi `length()`

Berdasarkan Gambar 6.2, diketahui jumlah anggota dari `list` **daftar** sebanyak 4. Untuk mengetahui jumlah anggota `list`, dapat digunakan fungsi `length()`. Berdasarkan Gambar 6.3, perintah R `length(daftar)` bertujuan untuk mengetahui banyaknya anggota dari `list` **daftar**, yakni sebanyak 4.

```
Console ~/ |
> daftar <- list(data1 = c(3,4,2), data2 = c(4,4,7,3), data3 = c("Anggur", "Ape1"), data4 = c("ugi", "Egi"))
> names(daftar)
[1] "data1" "data2" "data3" "data4"
> length(daftar)
[1] 4
> |
```

Gambar 6.3

6.4 Mengakses Anggota List

Gambar 6.4 diberikan ilustrasi untuk mengakses anggota `list` **daftar**. Berdasarkan Gambar 6.4, perintah R `daftar[1]` bertujuan mengakses anggota pertama dari `list` **daftar**, yakni **data1**, serta terlihat bahwa elemen dari **data1**, yakni 3, 4, dan 2. Perintah R `mode(daftar[1])` memberikan informasi bahwa `daftar[1]` berjenis `list`.

```
Console ~/ |
> daftar <- list(data1 = c(3,4,2), data2 = c(4,4,7,3), data3 = c("Anggur", "Ape1"), data4 = c("ugi", "Egi"))
> daftar[1]
$data1
[1] 3 4 2

> mode(daftar[1])
[1] "list"
> |
```

Gambar 6.4

Pada Gambar 6.5, perintah R `daftar[2]` bertujuan mengakses anggota kedua dari `list` **daftar**, yakni **data2**, serta terlihat bahwa elemen dari **data2**, yakni 4, 4, 7, dan 3. Perintah R `mode(daftar[2])` memberikan informasi bahwa `daftar[2]` berjenis `list`.

```
Console ~/ |
> daftar <- list(data1 = c(3,4,2), data2 = c(4,4,7,3), data3 = c("Anggur", "Ape1"), data4 = c("ugi", "Egi"))
> daftar[1]
$data1
[1] 3 4 2

> mode(daftar[1])
[1] "list"
> daftar[2]
$data2
[1] 4 4 7 3

> mode(daftar[2])
[1] "list"
> |
```

Gambar 6.5

Mengakses anggota dari *list* juga dapat menggunakan nama dari anggota *list*. Sebagai contoh perintah R `daftar["data1"]` bertujuan mengakses anggota *list* `daftar` bernama `data1`. Perintah R `daftar["data2"]` bertujuan mengakses anggota *list* `daftar` bernama `data2`. Perintah R `daftar[c(1,4)]` bertujuan mengakses anggota pertama dan keempat dari *list* `daftar`, yakni `data1` dan `data4`. Perhatikan Gambar 6.6.

```

Console -1
> daftar["data1"]
$data1
[1] 3 4 2

> daftar["data2"]
$data2
[1] 4 4 7 3

> daftar[c(1,4)]
$data1
[1] 3 4 2

$data4
[1] "Ugi" "Egi"

> daftar[c("data1", "data4")]
$data1
[1] 3 4 2

$data4
[1] "Ugi" "Egi"

```

Gambar 6.6

6.5 Mengakses Elemen dari Anggota *List*

Pada Gambar 6.7, perhatikan hasil dari perintah R `daftar[1]` dan `daftar[[1]]`. Hasil dari perintah R `daftar[1]` adalah

```

$data1
[1] 3 4 2

```

Sementara hasil dari perintah R `daftar[[1]]` adalah

```

[1] 3 4 2

```

Jenis dari `daftar[1]` adalah *list*, sementara jenis `daftar[[1]]` adalah *numeric* (vektor berjenis numerik). Perintah R `daftar[[1]]` atau `daftar[["data1"]]` atau `daftar$data1` mengakses elemen dari anggota *list* pertama.

```

Console ~/
> daftar
$data1
[1] 3 4 2

$data2
[1] 4 4 7 3

$data3
[1] "Anggur" "Ape1"

$data4
[1] "Ugi" "Egi"

> mode(daftar)
[1] "list"
> simpan1 <- daftar[1]
> simpan1
$data1
[1] 3 4 2

> mode(simpan1)
[1] "list"
> simpan2 <- daftar[[1]]
> simpan2
[1] 3 4 2
> mode(simpan2)
[1] "numeric"
> daftar[["data2"]]
[1] 4 4 7 3
> mode(daftar[["data2"]])
[1] "numeric"
> |

Console ~/
> daftar[[1]]
[1] 3 4 2
> mode(daftar[[1]])
[1] "numeric"
> daftar[["data1"]]
[1] 3 4 2
> mode(daftar[["data1"]])
[1] "numeric"
> daftar$data1
[1] 3 4 2
> mode(daftar$data1)
[1] "numeric"
> |

```

Gambar 6.7

6.6 Mengganti Nama dari Anggota List

Nama dari anggota *list* dapat diganti. Gambar 6.8 diberikan ilustrasi mengganti nama dari anggota *list* **daftar**. Pada awalnya, nama-nama anggota dari *list* **daftar** adalah **data1**, **data2**, **data3**, dan **data4**. Perintah R **names(daftar)[3] <- c("Buah")** bertujuan mengganti nama dari anggota *list* **daftar** yang ketiga (pada indeks ke-3) dengan nama “**Buah**”. Sementara perintah R **names(daftar)[c(1,4)] <- c("tes 1", "tes 4")** bertujuan untuk mengganti nama dari anggota *list* **daftar** yang pertama dan keempat (pada indeks ke-1 dan indeks ke-4) dengan nama “**tes 1**” dan “**tes 4**”.

```

Console ~/
> daftar
$data1
[1] 3 4 2

$data2
[1] 4 4 7 3

$data3
[1] "Anggur" "Ape1"

$data4
[1] "Ugi" "Egi"

> names(daftar)
[1] "data1" "data2" "data3" "data4"
> names(daftar)[3] <- c("Buah")
> names(daftar)
[1] "data1" "data2" "Buah" "data4"
> names(daftar)[c(1,4)] <- c("tes 1", "tes 4")
> names(daftar)
[1] "tes 1" "data2" "Buah" "tes 4"
> |

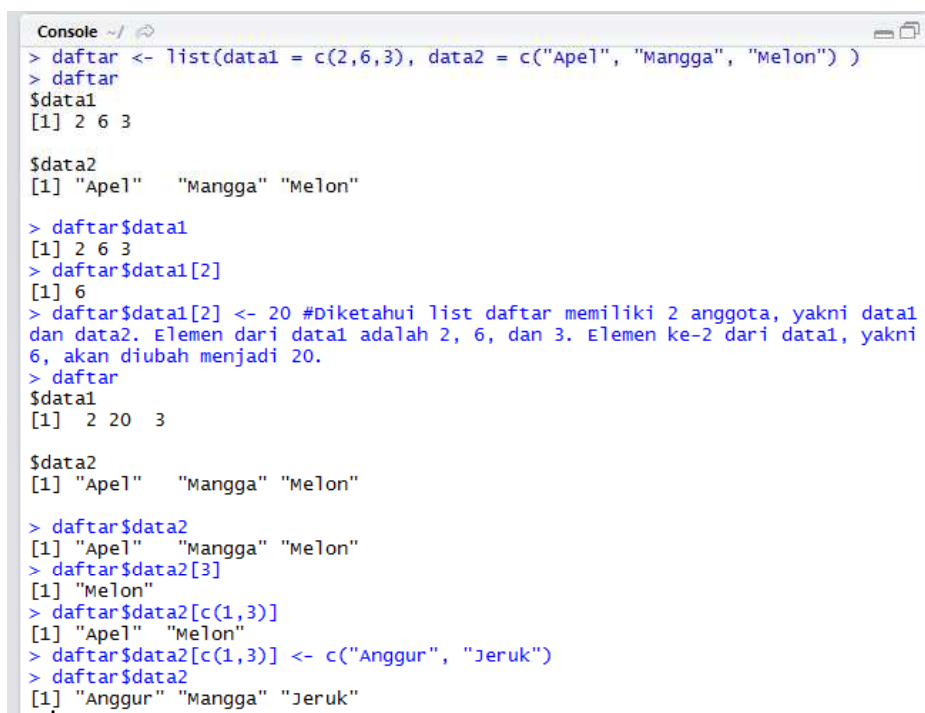
```

Gambar 6.8

6.7 Mengganti Elemen dari Anggota List

Di samping nama dari anggota *list* dapat diganti, elemen dari anggota *list* juga dapat diganti. Gambar 6.9 diberikan ilustrasi mengganti elemen dari anggota *list*.

- ⇒ Pertama, diketahui *list* **daftar** memiliki 2 anggota, yakni **data1** dan **data2**. Elemen dari **data1** adalah 2, 6, dan 3, sementara elemen dari **data2** adalah “Apel”, “Mangga”, dan “Melon”.
- ⇒ Selanjutnya elemen kedua dari **data1**, yakni 6, akan diubah menjadi 20. Diketahui nilai 6 terletak pada posisi atau indeks ke-2 pada **data1**. Perintah R untuk mengganti 6 menjadi 20 adalah **daftar\$data1[2] <- 20**.
- ⇒ Elemen pertama dan ketiga dari **data2**, yakni “Apel” dan “Melon”, akan diubah menjadi “Anggur” dan “Jeruk”. Diketahui “Apel” terletak pada posisi atau indeks ke-1 dari **data2**, sementara “Melon” terletak pada posisi atau indeks ke-3 dari **data2**. Perintah R untuk mengganti “Apel” menjadi “Anggur” dan “Melon” menjadi “Jeruk” adalah **daftar\$data2[c(1,3)] <- c("Anggur", "Jeruk")**.



```
Console ~/
> daftar <- list(data1 = c(2,6,3), data2 = c("Apel", "Mangga", "Melon"))
> daftar
 $data1
 [1] 2 6 3

 $data2
 [1] "Apel" "Mangga" "Melon"

> daftar$data1
 [1] 2 6 3
> daftar$data1[2]
 [1] 6
> daftar$data1[2] <- 20 #Diketahui list daftar memiliki 2 anggota, yakni data1
dan data2. Elemen dari data1 adalah 2, 6, dan 3. Elemen ke-2 dari data1, yakni
6, akan diubah menjadi 20.
> daftar
 $data1
 [1] 2 20 3

 $data2
 [1] "Apel" "Mangga" "Melon"

> daftar$data2
 [1] "Apel" "Mangga" "Melon"
> daftar$data2[3]
 [1] "Melon"
> daftar$data2[c(1,3)]
 [1] "Apel" "Melon"
> daftar$data2[c(1,3)] <- c("Anggur", "Jeruk")
> daftar
 $data1
 [1] "Anggur" "Mangga" "Jeruk"
```

Gambar 6.9

6.8 Menghapus Nama dan Memberi Nama dari Anggota List

Nama dari anggota *list* juga dapat dihapus. Gambar 6.10 diberikan ilustrasi dalam R menghapus nama dari anggota *list* **daftar**.

```

Console -/
> daftar
$data1
[1] 2 20 3

$data2
[1] "Anggur" "Mangga" "Jeruk"

> names(daftar)
[1] "data1" "data2"
> names(daftar) <- NULL
> daftar
[[1]]
[1] 2 20 3

[[2]]
[1] "Anggur" "Mangga" "Jeruk"

> names(daftar)
NULL
> mode(daftar)
[1] "list"

```

Gambar 6.10

Berdasarkan Gambar 6.10, diketahui anggota *list* **daftar** sebanyak 2, dengan nama **data1** dan **data2**. Pada Gambar 6.11, selanjutnya nama dari anggota *list* **daftar** dihapus dengan perintah R **names(daftar) <- NULL**, sehingga nama **data1** menjadi [1] dan nama **data2** menjadi [2]. Kemudian diberi nama kembali, yakni [1] menjadi **data pertama** dan [2] menjadi **data kedua**.

```

> daftar
$data1
[1] 2 20 3

$data2
[1] "Anggur" "Mangga" "Jeruk"

> names(daftar)
[1] "data1" "data2"
> names(daftar) <- NULL
> daftar
[[1]]
[1] 2 20 3

[[2]]
[1] "Anggur" "Mangga" "Jeruk"

> names(daftar)
NULL
> mode(daftar)
[1] "list"
> names(daftar) <- c("data pertama", "data kedua")
> daftar
$`data pertama`
[1] 2 20 3

$`data kedua`
[1] "Anggur" "Mangga" "Jeruk"

> names(daftar)
[1] "data pertama" "data kedua"

```

Gambar 6.11

6.9 Mengubah *List* Menjadi Vektor dengan Fungsi *unlist()*

Gambar 6.12 diberikan ilustrasi mengubah *list* menjadi vektor.

```

> daftar <- list(data1 = c(1,2), data2 = c(3,4), data3 = c(5,6))
> daftar
$data1
[1] 1 2

$data2
[1] 3 4

$data3
[1] 5 6

> mode(daftar)
[1] "list"
> daftar <- unlist(daftar)
> daftar
data11 data12 data21 data22 data31 data32
  1      2      3      4      5      6
> mode(daftar)
[1] "numeric"
> names(daftar) <- NULL
> daftar
[1] 1 2 3 4 5 6
> |

```

Gambar 6.12

Penggunaan fungsi *unlist()* mengembalikan suatu vektor. Berdasarkan Gambar 6.12, pada awalnya **daftar** adalah suatu *list*. Selanjutnya digunakan fungsi *unlist()*, yakni **daftar <- unlist(daftar)**. Hasilnya adalah **daftar** adalah suatu vektor berjenis numerik dengan nilai 1, 2, 3, 4, 5, dan 6.

6.10 Menambah dan Menghapus Anggota dari Suatu List

Anggota dari suatu *list* dapat ditambah atau dihapus. Gambar 6.13 diberikan ilustrasi menambah anggota dari *list* **daftar**. Pada awalnya anggota dari *list* **daftar** sebanyak 2, yakni **data1** dan **data2**. Selanjutnya perintah R **daftar\$data3 <- c("Medan", "Bandung", "Jogja")** bertujuan untuk menambah anggota dari *list* **daftar**, dengan nama anggota **data3**. Sehingga anggota *list* **daftar** sebanyak 3, yakni **data1**, **data2**, dan **data3**. Kemudian anggota *list* **daftar** yang kedua, yakni **data2** dihapus dengan perintah R **daftar <- daftar[-c(2)]**, sehingga anggota *list* **daftar** yang sekarang adalah **data1** dan **data3**.

```

Console ~1
> daftar <- list(data1 = c(1,2,3), data2 = c("Anggur", "Pepaya"))
> daftar
$data1
[1] 1 2 3

$data2
[1] "Anggur" "Pepaya"

> daftar$data3 <- c("Medan", "Bandung", "Jogja")
> daftar
$data1
[1] 1 2 3

$data2
[1] "Anggur" "Pepaya"

$data3
[1] "Medan" "Bandung" "Jogja"

> daftar <- daftar[-c(2)]
> daftar
$data1
[1] 1 2 3

$data3
[1] "Medan" "Bandung" "Jogja"

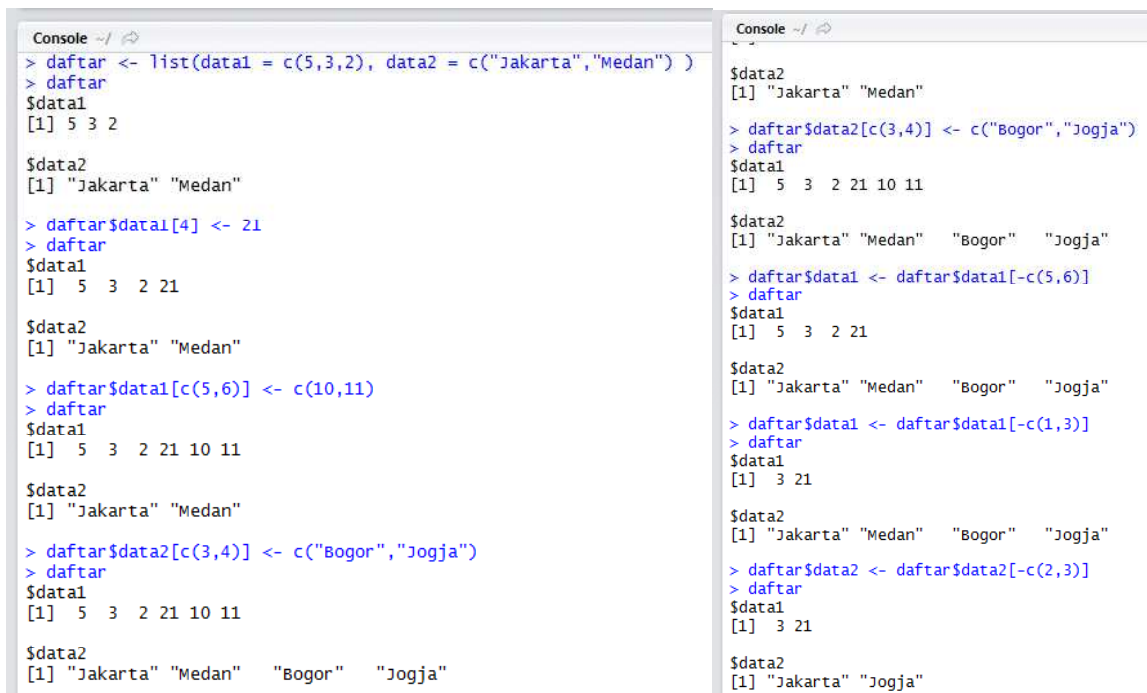
```

Gambar 6.13

6.11 Menambah dan Menghapus Elemen dari Anggota List

Elemen dari anggota *list* juga dapat ditambah atau dihapus. Pada Gambar 6.14, pada awalnya anggota dari *list* **daftar** adalah **data1** dan **data2**. Elemen dari **data1** adalah 5, 3, dan 2. Perintah R **daftar\$data1[c(4)] <- 21** bertujuan untuk menugaskan bilangan 21 disimpan ke dalam **data1**. Pada “**c(4)**” menyatakan indeks ke-4 dari **data1**. Sehingga elemen dari **data1** saat ini adalah 5, 3, 2, dan 21. Kemudian elemen dari **data1** akan ditambah lagi dengan bilangan 10 dan 11. Perintah R untuk menambah bilangan 10 dan 11 di **data1** adalah **daftar\$data1[c(5,6)] <- c(10,11)**. Perintah R **daftar\$data1[c(5,6)] <- c(10,11)** dapat juga diartikan menugaskan bilangan 10 dan 11 ke **data1** pada indeks ke-5 dan indeks ke-6. Sehingga elemen dari **data1** saat ini adalah 5, 3, 2, 21, 10, dan 11.

Diketahui elemen dari **data2** adalah “Jakarta” dan “Medan”. Perintah R **daftar\$data2[c(3,4)] <- c(“Bogor”, “Jogja”)** bertujuan menambah atau menugaskan “Bogor” dan “Jogja” ke **data2** pada indeks ke-3 dan indeks ke-4. Sehingga elemen dari **data2** adalah “Jakarta”, “Medan”, “Bogor”, dan “Jogja”. Selanjutnya misalkan ingin dihapus elemen dari **data1** pada indeks ke-5 dan indeks ke-6, yakni bilangan 10 dan 11. Perintah R untuk menghapus elemen dari **data1** pada indeks ke-5 dan indeks ke-6 adalah **daftar\$data1 <- daftar\$data1[-c(5,6)]**. Sehingga elemen dari **data1** saat ini menjadi 5, 3, 2, dan 21. Perintah R **daftar\$data1 <- daftar\$data1[-c(1,3)]** bertujuan untuk menghapus elemen dari **data1** pada indeks ke-1 dan indeks ke-3. Sehingga elemen dari **data1** saat ini menjadi 3 dan 21. Perintah R **daftar\$data2 <- daftar\$data2[-c(2,3)]** bertujuan untuk menghapus elemen dari **data2** pada indeks ke-2 dan indeks ke-3. Sehingga elemen dari **data2** saat ini menjadi “Jakarta” dan “Jogja”.



```
Console ~/
> daftar <- list(data1 = c(5,3,2), data2 = c("Jakarta","Medan"))
> daftar
$data1
[1] 5 3 2

$data2
[1] "Jakarta" "Medan"

> daftar$data1[4] <- 21
> daftar
$data1
[1] 5 3 2 21

$data2
[1] "Jakarta" "Medan"

> daftar$data1[c(5,6)] <- c(10,11)
> daftar
$data1
[1] 5 3 2 21 10 11

$data2
[1] "Jakarta" "Medan"

> daftar$data2[c(3,4)] <- c("Bogor","Jogja")
> daftar
$data1
[1] 5 3 2 21 10 11

$data2
[1] "Jakarta" "Medan" "Bogor" "Jogja"

> daftar$data1 <- daftar$data1[-c(5,6)]
> daftar
$data1
[1] 5 3 2 21

$data2
[1] "Jakarta" "Medan" "Bogor" "Jogja"

> daftar$data1 <- daftar$data1[-c(1,3)]
> daftar
$data1
[1] 3 21

$data2
[1] "Jakarta" "Medan" "Bogor" "Jogja"

> daftar$data2 <- daftar$data2[-c(2,3)]
> daftar
$data1
[1] 3 21

$data2
[1] "Jakarta" "Jogja"
```

Gambar 6.14

6.12 Fungsi Regresi Linear $lm()$ Mengembalikan *List*

Gambar 6.15 diberikan ilustrasi penggunaan fungsi $lm()$ untuk melakukan metode statistika regresi linear, dengan Y sebagai variabel tak bebas dan X sebagai variabel bebas. Fungsi $lm()$ mengembalikan *list*. Pada perintah R **regresi <- lm(Y ~ X)**, hasil dari **lm(Y ~ X)** ditugaskan atau disimpan ke **regresi**. Hasil dari perintah R **mode(regresi)** memperlihatkan bahwa **regresi** berjenis *list*. Perintah R **names(regresi)** memperlihatkan seluruh anggota dari *list regresi*, yakni “*coefficients*”, “*residuals*”, “*effects*”, dan sebagainya. Perintah R **regresi\$coefficients** menampilkan elemen dari *coefficients*, yakni berupa nilai *intercept* (-0,333333) dan *slope* (1,1428571).

```
Console ~/
> X <- c(1,2,3,4,5,6)
> Y <- c(2,1,3,4,4,8)
> regresi <- lm(Y~X)
> summary(regresi)

Call:
lm(formula = Y ~ X)

Residuals:
    1     2     3     4     5     6 
1.19048 -0.95238 -0.09524 -0.23810 -1.38095  1.47619 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.33333   1.1846   -0.281   0.7924
X              1.1429   0.3042    3.757   0.0198 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.272 on 4 degrees of freedom
Multiple R-squared:  0.7792,    Adjusted R-squared:  0.724 
F-statistic: 14.12 on 1 and 4 DF,  p-value: 0.01982

> mode(regresi)
[1] "list"
> names(regresi)
 [1] "coefficients" "residuals"    "effects"      "rank"         "fitted.values" "assign"
 [7] "qr"           "df.residual"  "xlevels"      "call"         "terms"        "model"
> regresi$coefficients
(Intercept)      X
-0.3333333    1.1428571
> regresi$residuals
    1     2     3     4     5     6 
1.1904762 -0.9523810 -0.0952381 -0.2380952 -1.3809524  1.4761905
> regresi$call
lm(formula = Y ~ X)
```

Gambar 6.15

BAB 7

DATA FRAME

7.1 Antara Data Frame, List, dan Matriks

Gambar 7.1 diberikan contoh *data frame*, *list*, dan matriks. Perhatikan bahwa **contoh_dataframe** adalah *data frame*, **contoh_list** adalah *list*, dan **contoh_matriks** adalah matriks.

- ⇒ Matriks dan *data frame* sama-sama memiliki struktur baris dan kolom, sementara *list* tidak. Perhatikan bahwa perintah R **contoh_dataframe[1,2]** akan menampilkan data pada *data frame* **contoh_dataframe** pada baris ke-1 dan kolom ke-2, yakni 27. Perintah R **contoh_matriks[3,1]** akan menampilkan data pada matriks **contoh_matriks** pada baris ke-3 dan kolom ke-1, yakni 5. Perhatikan bahwa perintah R **contoh_list[1,2]** memberikan pesan kesalahan.

```
Console ~1 ↻
> contoh_dataframe <- data.frame>Nama=c("Ugi","Egi"), Usia=c(27,23))
> contoh_list <- list>Nama=c("Ugi","Egi","Sinta"), Usia=c(27,25,24,29,28))
> contoh_matriks <- matrix(c(1,2,3,4,5,6), ncol = 2, byrow = T)
> contoh_dataframe
  Nama Usia
1 Ugi   27
2 Egi   23
> contoh_dataframe[1,2]
[1] 27
> contoh_dataframe[2,1]
[1] Egi
Levels: Egi Ugi
> contoh_dataframe[2,2]
[1] 23
> contoh_list
$Nama
[1] "Ugi" "Egi" "Sinta"

$Usia
[1] 27 25 24 29 28

> contoh_list[2]
$Usia
[1] 27 25 24 29 28

> contoh_list[1,2]
Error in contoh_list[1, 2] : incorrect number of dimensions
> contoh_matriks
  [,1] [,2]
[1,]  1    2
[2,]  3    4
[3,]  5    6
> contoh_matriks[3,1]
[1] 5
~1
```

Gambar 7.1

- ⇒ Pada *data frame* **contoh_dataframe**, terdapat dua vektor atau kolom, yakni **Nama** dan **Usia**. **Nama** berjenis *character*, sementara **Usia** berjenis numerik. Pada *list* **contoh_list** juga terdapat dua vektor atau anggota (*list member*), yakni **Nama** dan **Usia**, yang masing-masing

berjenis *character* dan *numeric*. Pada *data frame* dan *list* sama-sama dapat memiliki beberapa vektor yang berbeda jenis, seperti yang telah diperlihatkan. Namun pada *data frame*, panjang dari tiap-tiap vektor atau kolom harus sama, sementara pada *list* dapat berbeda.

⇒ Pada *data frame*, panjang dari masing-masing vektor atau kolom harus sama, begitu juga pada matriks.

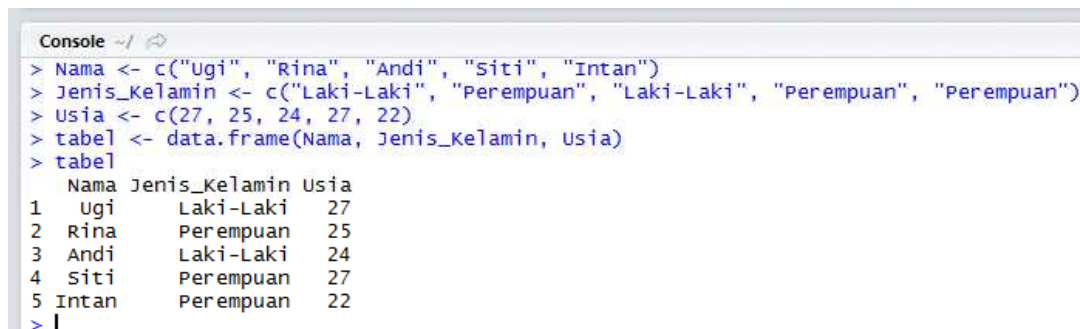
Norman Matloff dalam bukunya yang berjudul “*The Art of R Programming*” menyatakan sebagai berikut (2009).

“*On an intuitive level, a data frame is like a matrix, with a rows-and-column structure. However, it differs from a matrix in that each column may have a different mode. For instance, one column may be numbers and another column might be character strings.*

On a technical level, a data frame is a list of equal-length vectors. Each column is one element of the list.”

7.2 Membuat *Data Frame* dengan Fungsi *data.frame()*

Dalam R, fungsi *data.frame()* digunakan untuk membuat suatu *data frame*. Gambar 7.1 diberikan contoh membuat *data frame* bernama **tabel**. Perhatikan bahwa pada *data frame* yang bernama **tabel** terdiri dari 3 vektor atau kolom, yakni **Nama**, **Jenis_Kelamin**, dan **Usia**. Masing-masing vektor memiliki panjang (*length*) yang sama, yakni 5. Pada *data frame*, panjang dari masing-masing kolom atau vektor harus sama.



```
Console ~/\> Nama <- c("Ugi", "Rina", "Andi", "Siti", "Intan")
> Jenis_Kelamin <- c("Laki-Laki", "Perempuan", "Laki-Laki", "Perempuan", "Perempuan")
> Usia <- c(27, 25, 24, 27, 22)
> tabel <- data.frame>Nama, Jenis_Kelamin, Usia)
> tabel
  Nama Jenis_Kelamin Usia
1  Ugi      Laki-Laki  27
2  Rina     Perempuan  25
3  Andi      Laki-Laki  24
4  Siti     Perempuan  27
5  Intan     Perempuan  22
> |
```

Gambar 7.2

Pada Gambar 7.3, bermaksud juga untuk membuat *data frame* bernama **tabel2**, namun memberikan pesan kesalahan. Hal ini karena panjang dari vektor **Nama2**, **Jenis_Kelamin2**, dan **Usia** berbeda-beda. Panjang dari vektor **Nama2** adalah 3, panjang dari vektor **Jenis_Kelamin2** adalah 2, dan panjang dari vektor **Usia** adalah 4.

```

> Nama2 <- c("Fitri", "Iman", "Zeri")
> Jenis_Kelamin2 <- c("Perempuan", "Laki-Laki")
> Usia <- c(24, 21, 32, 25)
> tabel2 <- data.frame(Nama2, Jenis_Kelamin2, Usia)
Error in data.frame(Nama2, Jenis_Kelamin2, Usia) :
  arguments imply differing number of rows: 3, 2, 4
> |

```

Gambar 7.3

Berdasarkan Gambar 7.2, *data frame* **tabel** memiliki tiga kolom, kolom pertama dengan nama **Nama**, kolom kedua dengan nama **Jenis_Kelamin**, dan kolom ketiga dengan nama **Usia**.

7.3Memperoleh Nama-Nama Kolom dari *Data Frame* dengan Fungsi `names()` dan `colnames()`

Diketahui sebelumnya, *data frame* **tabel** memiliki tiga kolom, dengan nama **Nama**, **Jenis_Kelamin**, dan **Usia**. Fungsi `names()` dapat digunakan untuk memperoleh nama-nama kolom dari *data frame*. Pada Gambar 7.4, perintah R `names(tabel)` dan `colnames(tabel)` menampilkan nama-nama kolom atau vektor dari *data frame* **tabel**, yakni **Nama**, **Jenis_Kelamin**, dan **Usia**.

```

Console ~/ |
> Nama <- c("Ugi","Rina","Andi","Siti","Intan")
> Jenis_Kelamin <- c("Laki-Laki","Perempuan","Laki-Laki","Perempuan","Perempuan")
> Usia <- c(27,25,24,27,22)
>
> tabel <- data.frame(Nama, Jenis_Kelamin, Usia)
> tabel
  Nama Jenis_Kelamin Usia
1  Ugi      Laki-Laki  27
2  Rina     Perempuan  25
3  Andi     Laki-Laki  24
4  Siti     Perempuan  27
5  Intan    Perempuan  22
> names(tabel)
[1] "Nama"      "Jenis_Kelamin" "Usia"
> colnames(tabel)
[1] "Nama"      "Jenis_Kelamin" "Usia"
> |

```

Gambar 7.4

7.4Mengetahui Jumlah Kolom dari Suatu *Data Frame* Fungsi `length()`

Berdasarkan Gambar 7.4, diketahui jumlah kolom dari *data frame* **tabel** sebanyak 3. Untuk mengetahui jumlah kolom dari suatu *data frame*, dapat digunakan fungsi `length()`. Berdasarkan Gambar 7.5, perintah R `length(tabel)` bertujuan untuk mengetahui banyaknya kolom atau vektor dari *data frame* **tabel**, yakni sebanyak 3.

```

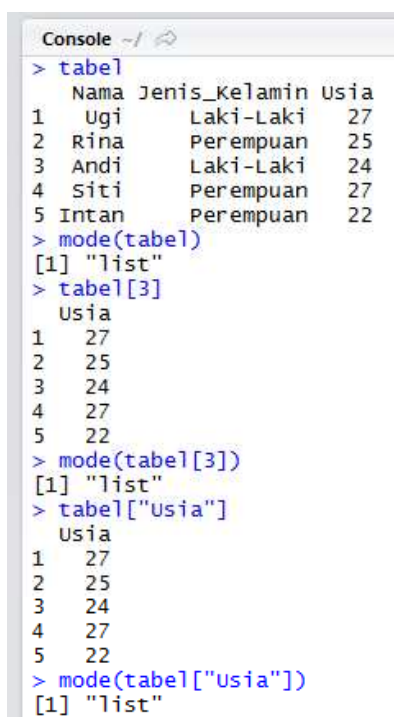
Console ~/ |
> tabel
  Nama Jenis_Kelamin Usia
1  Ugi      Laki-Laki  27
2  Rina     Perempuan  25
3  Andi     Laki-Laki  24
4  Siti     Perempuan  27
5  Intan    Perempuan  22
> length(tabel)
[1] 3
> |

```

Gambar 7.5

7.5 Mengakses Kolom pada Data Frame

Gambar 7.6 diberikan ilustrasi untuk mengakses atau menampilkan kolom pada *data frame* **tabel**. Berdasarkan Gambar 7.6, perintah R **tabel[3]** bertujuan mengakses atau menampilkan kolom atau vektor ketiga dari *data frame* **tabel**, yakni **Usia**, serta terlihat bahwa elemen dari **Usia**, yakni 27, 25, 24, 27, dan 22. Perintah R **mode(tabel[3])** memberikan informasi bahwa **tabel[3]** berjenis *list*. Begitu juga perintah R **tabel["Usia"]** memberikan hasil yang sama dengan perintah R **tabel[3]**.



```
Console -/ ↻
> tabel
  Nama Jenis_Kelamin Usia
1  Ugi      Laki-Laki  27
2  Rina     Perempuan  25
3  Andi     Laki-Laki  24
4  Siti     Perempuan  27
5  Intan    Perempuan  22
> mode(tabel)
[1] "list"
> tabel[3]
  Usia
1  27
2  25
3  24
4  27
5  22
> mode(tabel[3])
[1] "list"
> tabel["Usia"]
  Usia
1  27
2  25
3  24
4  27
5  22
> mode(tabel["Usia"])
[1] "list"
```

Gambar 7.6

Pada Gambar 7.7, perintah R **tabel[2]** bertujuan mengakses atau menampilkan kolom kedua dari *data frame* **tabel**, yakni **Jenis_Kelamin**, serta terlihat bahwa elemen dari **Jenis_Kelamin**, yakni “Laki-Laki”, “Perempuan”, “Laki-Laki”, “Perempuan”, dan “Perempuan”. Perintah R **mode(tabel[2])** memberikan informasi bahwa **tabel[2]** berjenis *list*. Perintah R **tabel["Jenis_Kelamin"]** memberikan hasil yang sama dengan perintah R **tabel[2]**. Sementara pada Gambar 7.8, perintah R **tabel[c(1,3)]** bertujuan mengakses atau menampilkan kolom pertama dan ketiga dari *data frame* **tabel**, yakni **Nama** dan **Usia**.

```

Console -/ ↻
> tabel[2]
  Jenis_kelamin
1     Laki-Laki
2     Perempuan
3     Laki-Laki
4     Perempuan
5     Perempuan
> mode(tabel[2])
[1] "list"
> tabel["Jenis_kelamin"]
  Jenis_kelamin
1     Laki-Laki
2     Perempuan
3     Laki-Laki
4     Perempuan
5     Perempuan
> mode(tabel["Jenis_kelamin"])
[1] "list"
> |

```

Gambar 7.7

```

Console -/ ↻
> tabel[c(1,3)]
  Nama Usia
1  Ugi   27
2  Rina  25
3  Andi  24
4  Siti  27
5  Intan 22
> |

```

Gambar 7.8

7.6 Mengakses Baris pada Data Frame

Baris dari suatu *data frame* juga dapat diakses. Perhatikan Gambar 7.9. Berdasarkan Gambar 7.9, perintah R `tabel[1,]` bertujuan untuk mengakses atau menampilkan data pada baris pertama, dari *data frame* `tabel`. Sementara perintah R `tabel[5,]` bertujuan untuk mengakses atau menampilkan data pada baris kelima, dari *data frame* `tabel`. Perintah R `tabel[c(2,4),]` bertujuan untuk mengakses atau menampilkan data pada baris kedua dan keempat, dari *data frame* `tabel`.

```

Console -/ ↻
> tabel
  Nama Jenis_kelamin Usia
1  Ugi     Laki-Laki   27
2  Rina    Perempuan   25
3  Andi    Laki-Laki   24
4  Siti    Perempuan   27
5  Intan   Perempuan   22
> tabel[1,]
  Nama Jenis_kelamin Usia
1  Ugi     Laki-Laki   27
> tabel[5,]
  Nama Jenis_kelamin Usia
5  Intan   Perempuan   22
> tabel[c(2,4),]
  Nama Jenis_kelamin Usia
2  Rina    Perempuan   25
4  Siti    Perempuan   27
>

```

Gambar 7.9

7.7 Mengakses Elemen dari Kolom pada Data Frame

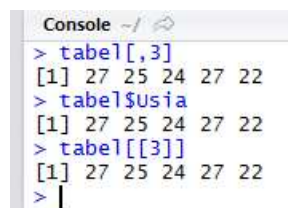
Pada Gambar 7.10, perhatikan hasil dari perintah R `tabel[,3]` dan `tabel[3]`. Hasil dari perintah R `tabel[,3]` adalah

```
[1] 27 25 24 27 22
```

Sementara hasil dari perintah R `tabel[3]` adalah

	Usia
1	27
2	25
3	24
4	27
5	22

Jenis dari `tabel[,3]` adalah *numeric*, sementara jenis `tabel[3]` adalah *list*. Perintah R `tabel[,3]` bertujuan untuk mengakses elemen (data) dari kolom ketiga dari *data frame* `tabel`. Perintah R `tabel[3]` memberikan hasil yang sama dengan perintah R `tabel$Usia` atau `tabel[[3]]`.

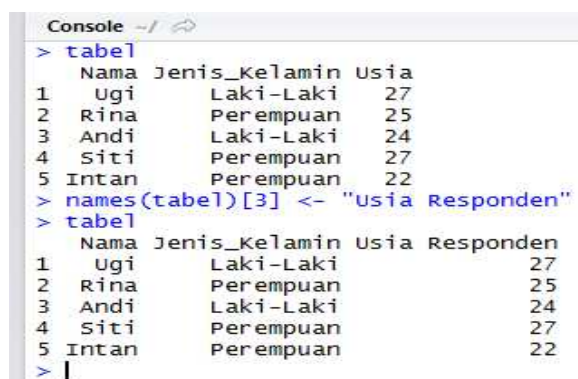


```
Console -/ ↻
> tabel[,3]
[1] 27 25 24 27 22
> tabel$Usia
[1] 27 25 24 27 22
> tabel[[3]]
[1] 27 25 24 27 22
> |
```

Gambar 7.10

7.8 Mengganti Nama Kolom dari Data Frame dengan Fungsi `names()` dan `colnames()`

Nama kolom atau vektor dari *data frame* dapat diganti. Perhatikan Gambar 7.11. Pada Gambar 7.11, nama kolom `Usia` diganti menjadi `Usia Responden`. Perintah R untuk mengganti nama kolom `Usia` menjadi `Usia Responden` adalah `names(tabel)[3] <- "Usia Responden"`.



```
Console -/ ↻
> tabel
  Nama Jenis_Kelamin Usia
1  Ugi      Laki-Laki  27
2  Rina     Perempuan  25
3  Andi     Laki-Laki  24
4  Siti     Perempuan  27
5  Intan    Perempuan  22
> names(tabel)[3] <- "Usia Responden"
> tabel
  Nama Jenis_Kelamin Usia Responden
1  Ugi      Laki-Laki      27
2  Rina     Perempuan     25
3  Andi     Laki-Laki     24
4  Siti     Perempuan     27
5  Intan    Perempuan     22
> |
```

Gambar 7.11

Pada Gambar 7.12, perintah R `names(tabel)[c(1,2)] <- c("Nama Responden", "Jenis Kelamin Responden")` bertujuan mengganti nama **Nama** menjadi **Nama Responden** dan mengganti nama **Jenis_Kelamin** menjadi **Jenis Kelamin Responden**.

```

Console ~/
> tabel
  Nama Jenis_Kelamin Usia
1  Ugi      Laki-Laki   27
2  Rina    Perempuan    25
3  Andi    Laki-Laki    24
4  Siti    Perempuan    27
5  Intan   Perempuan    22
> names(tabel)[3] <- "Usia Responden"
> tabel
  Nama Jenis_Kelamin Usia Responden
1  Ugi      Laki-Laki      27
2  Rina    Perempuan      25
3  Andi    Laki-Laki      24
4  Siti    Perempuan      27
5  Intan   Perempuan      22
> names(tabel)[c(1,2)] <- c("Nama Responden", "Jenis Kelamin Responden")
> tabel
  Nama Responden Jenis Kelamin Responden Usia Responden
1      Ugi      Laki-Laki      27
2      Rina    Perempuan      25
3      Andi    Laki-Laki      24
4      Siti    Perempuan      27
5      Intan   Perempuan      22

```

Gambar 7.12

Pada Gambar 7.13, digunakan fungsi `colnames()` untuk mengganti nama kolom.

```

Console ~/
> tabel
  Nama Responden Jenis Kelamin Responden Usia Responden
1      Ugi      Laki-Laki      27
2      Rina    Perempuan      25
3      Andi    Laki-Laki      24
4      Siti    Perempuan      27
5      Intan   Perempuan      22
> colnames(tabel)[1] <- "Nama"
> tabel
  Nama Jenis Kelamin Responden Usia Responden
1  Ugi      Laki-Laki      27
2  Rina    Perempuan      25
3  Andi    Laki-Laki      24
4  Siti    Perempuan      27
5  Intan   Perempuan      22
> colnames(tabel)[c(2:3)] <- c("Jenis", "usia")
> tabel
  Nama      Jenis  usia
1  Ugi Laki-Laki  27
2  Rina Perempuan  25
3  Andi Laki-Laki  24
4  Siti Perempuan  27
5  Intan Perempuan  22
>

```

Gambar 7.13

7.9 Mengganti Data dari Data Frame

Data atau elemen pada *data frame* dapat diganti. Pada Gambar 7.14, diketahui usia Siti adalah 27. Nilai 27 berada pada baris ke-4 dan kolom ke-3. Nilai 27 kemudian diganti menjadi 30. Perintah R untuk mengganti usia 27 menjadi 30 adalah `tabel[4,3] <- 30`.

```

Console ~1 ↵
> tabel
  Nama      Jenis Usia
1  Ugi  Laki-Laki  27
2  Rina Perempuan  25
3  Andi  Laki-Laki  24
4  Siti  Perempuan  27
5  Intan Perempuan  22
> tabel[4,3]
[1] 27
> tabel[4,3] <- 30
> tabel[4,3]
[1] 30
> tabel
  Nama      Jenis Usia
1  Ugi  Laki-Laki  27
2  Rina Perempuan  25
3  Andi  Laki-Laki  24
4  Siti  Perempuan  30
5  Intan Perempuan  22
> |

```

Gambar 7.14

Pada Gambar 7.15, nama Intan diganti menjadi Andi, begitu juga dengan jenis kelaminnya menjadi laki-laki.

```

Console ~1 ↵
> tabel
  Nama      Jenis Usia
1  Ugi  Laki-Laki  27
2  Rina Perempuan  25
3  Andi  Laki-Laki  24
4  Siti  Perempuan  30
5  Intan Perempuan  22
> tabel[5,1] <- "Andi"
> tabel[5,2] <- "Laki-Laki"
> tabel
  Nama      Jenis Usia
1  Ugi  Laki-Laki  27
2  Rina Perempuan  25
3  Andi  Laki-Laki  24
4  Siti  Perempuan  30
5  Andi  Laki-Laki  22
> |

```

Gambar 7.15

7.10 Menghapus Kolom pada Data Frame

Kolom pada *data frame* dapat dihapus. Pada Gambar 7.16, kolom **Jenis** dihapus. Kolom **Jenis** berada pada indeks ke-2. Perintah R untuk menghapus kolom **Jenis** adalah `tabel <- tabel[-c(2)]`.

```

Console ~1 ↵
> tabel
  Nama      Jenis Usia
1  Ugi  Laki-Laki  27
2  Rina Perempuan  25
3  Andi  Laki-Laki  24
4  Siti  Perempuan  30
5  Andi  Laki-Laki  22
> tabel <- tabel[-c(2)]
> tabel
  Nama      Usia
1  Ugi      27
2  Rina     25
3  Andi     24
4  Siti     30
5  Andi     22
> |

```

Gambar 7.16

7.11 Menghapus Data pada Data Frame

Data pada *data frame* dapat dihapus. Pada Gambar 7.17, data pada baris ke-4, yakni Siti dihapus. Perintah R untuk menghapus data pada baris ke-4 adalah `tabel <- tabel[-4,]`. Selanjutnya data Andi dengan usia 24 tahun dan Andi dengan usia 22 tahun dihapus dengan perintah R `tabel <- tabel[-c(3,4),]`.

```
Console ~/ |
> tabel
  Nama Usia
1 Ugi    27
2 Rina   25
3 Andi   24
4 Siti   30
5 Andi   22
> tabel <- tabel[-4,]
> tabel
  Nama Usia
1 Ugi    27
2 Rina   25
3 Andi   24
5 Andi   22
> tabel <- tabel[-c(3,4),]
> tabel
  Nama Usia
1 Ugi    27
2 Rina   25
>
```

Gambar 7.17

7.12 Menambah Kolom pada Data Frame

Selain dihapus, kolom pada *data frame* dapat ditambah. Pada Gambar 7.18, perintah R

```
tabel[3] <-c("S1","S2")
names(tabel)[3] <- "Pendidikan"
```

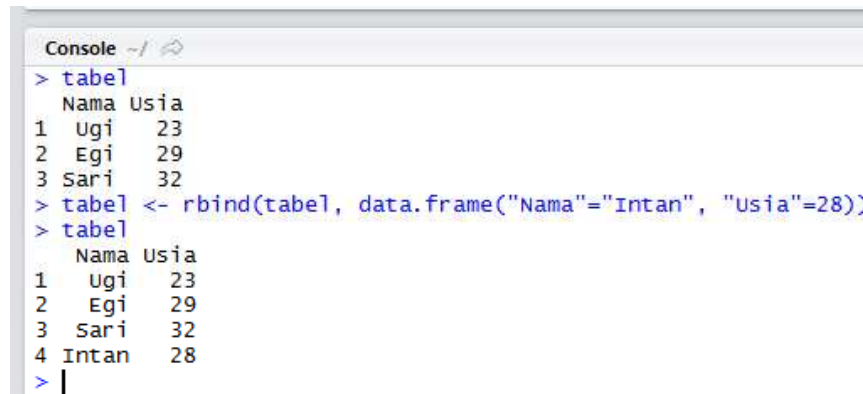
bertujuan untuk menambah kolom dengan nama **Pendidikan**.

```
Console ~/ |
> tabel
  Nama Usia
1 Ugi    27
2 Rina   25
> tabel[3] <-c("S1","S2")
> tabel
  Nama Usia v3
1 Ugi    27 S1
2 Rina   25 S2
> names(tabel)[3] <- "Pendidikan"
> tabel
  Nama Usia Pendidikan
1 Ugi    27         S1
2 Rina   25         S2
> tabel["Pendidikan"]
  Pendidikan
1         S1
2         S2
> |
```

Gambar 7.18

7.13 Menambah Data pada *Data Frame*

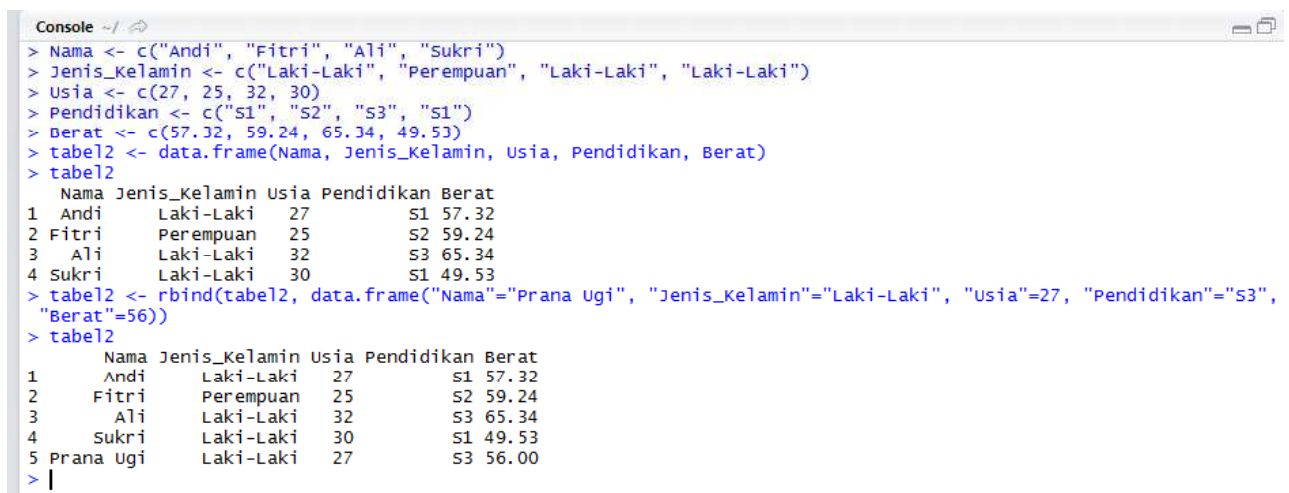
Gambar 7.19 memperlihatkan penambahan data pada *data frame* **tabel**. Pada Gambar 7.19, perintah R **tabel <- rbind(tabel, data.frame("Nama"="Intan", "Usia"=28))** bertujuan untuk menambah data dengan nama Intan, berusia 28 tahun.



```
Console ~/
> tabel
  Nama Usia
1 Ugi   23
2 Egi   29
3 Sari  32
> tabel <- rbind(tabel, data.frame("Nama"="Intan", "Usia"=28))
> tabel
  Nama Usia
1 Ugi   23
2 Egi   29
3 Sari  32
4 Intan 28
> |
```

Gambar 7.19

Pada Gambar 7.20, perintah R **tabel2 <- rbind(tabel2, data.frame("Nama"="Prana Ugi", "Jenis_Kelamin"="Laki-Laki", "Usia"=27, "Pendidikan"="S3", "Berat"=56))** bertujuan untuk menambah data pada *data frame* **tabel2** dengan nama Prana Ugi, berjenis kelamin laki-laki, usia 27 tahun, pendidikan S3, dan berat 56.



```
Console ~/
> Nama <- c("Andi", "Fitri", "Ali", "Sukri")
> Jenis_Kelamin <- c("Laki-Laki", "Perempuan", "Laki-Laki", "Laki-Laki")
> Usia <- c(27, 25, 32, 30)
> Pendidikan <- c("S1", "S2", "S3", "S1")
> Berat <- c(57.32, 59.24, 65.34, 49.53)
> tabel2 <- data.frame(Nama, Jenis_Kelamin, Usia, Pendidikan, Berat)
> tabel2
  Nama Jenis_Kelamin Usia Pendidikan Berat
1 Andi   Laki-Laki    27         S1  57.32
2 Fitri Perempuan    25         S2  59.24
3 Ali    Laki-Laki    32         S3  65.34
4 Sukri  Laki-Laki    30         S1  49.53
> tabel2 <- rbind(tabel2, data.frame("Nama"="Prana ugi", "Jenis_kelamin"="Laki-Laki", "Usia"=27, "Pendidikan"="S3",
"Berat"=56))
> tabel2
  Nama Jenis_Kelamin Usia Pendidikan Berat
1 Andi   Laki-Laki    27         S1  57.32
2 Fitri  Perempuan    25         S2  59.24
3 Ali    Laki-Laki    32         S3  65.34
4 Sukri  Laki-Laki    30         S1  49.53
5 Prana Ugi Laki-Laki    27         S3  56.00
> |
```

Gambar 7.20

7.14 Menggabungkan *Data Frame* dengan Fungsi *rbind()* berdasarkan Baris

Pada Gambar 7.21, diperlihatkan dua *data frame*, yakni **tabel** dan **tabel2**. Jumlah kolom atau vektor pada *data frame* **tabel** dan **tabel2** adalah 2, yakni **Nama** dan **Usia**. Selanjutnya data pada *data frame* **tabel** dan **tabel2** digabung berdasarkan baris, dengan perintah R **rbind(tabel, tabel2)**. Hasilnya diperlihatkan pada Gambar 7.21.

```
Console ~/ |
> tabel
  Nama Usia
1  Ugi  23
2  Egi  29
3  Sari 32
4  Intan 28
> tabel2
  Nama Usia
1  Budi  27
2  Sinta 27
3  Rani  25
4  Andi  31
> tabel_gabungan <- rbind(tabel, tabel2)
> tabel_gabungan
  Nama Usia
1  Ugi  23
2  Egi  29
3  Sari 32
4  Intan 28
5  Budi  27
6  Sinta 27
7  Rani  25
8  Andi  31
~ |
```

Gambar 7.21

7.15 Menggabungkan *Data Frame* dengan Fungsi `dataframe()` berdasarkan Kolom

Pada Gambar 7.22, diperlihatkan dua *data frame*, yakni `tabel_gabungan` dan `tabel3`. Kolom-kolom atau vektor-vektor pada *data frame* `tabel_gabungan` adalah **Nama** dan **Usia**, sementara kolom-kolom atau vektor-vektor pada *data frame* `tabel3` adalah **Berat** dan **Jenis_Kelamin**. Selanjutnya kolom-kolom pada *data frame* `tabel_gabungan` dan `tabel3` digabung, dengan perintah R `data.frame(tabel_gabungan, tabel3)`. Hasilnya diperlihatkan pada Gambar 7.22.

```

> tabel_gabungan
  Nama Usia
1  Ugi    23
2  Egi    29
3  Sari   32
4  Intan  28
5  Budi   27
6  Sinta  27
7  Rani   25
8  Andi   31
> Berat <- c(54.43, 67.53, 66.65, 72.32, 49.54, 56.54, 52.45, 56.43)
> Jenis_kelamin <- c("Laki-Laki", "Laki-Laki", "Perempuan", "Perempuan", "Laki-Laki", "Perempuan", "Perempuan", "Laki-Laki")
> tabel3 <- data.frame(Berat, Jenis_kelamin)
> tabel3
  Berat Jenis_kelamin
1 54.43 Laki-Laki
2 67.53 Laki-Laki
3 66.65 Perempuan
4 72.32 Perempuan
5 49.54 Laki-Laki
6 56.54 Perempuan
7 52.45 Perempuan
8 56.43 Laki-Laki
> tabel_gabungan <- data.frame(tabel_gabungan, tabel3)
> tabel_gabungan
  Nama Usia Berat Jenis_kelamin
1  Ugi    23 54.43 Laki-Laki
2  Egi    29 67.53 Laki-Laki
3  Sari   32 66.65 Perempuan
4  Intan  28 72.32 Perempuan
5  Budi   27 49.54 Laki-Laki
6  Sinta  27 56.54 Perempuan
7  Rani   25 52.45 Perempuan
8  Andi   31 56.43 Laki-Laki

```

Gambar 7.22

7.16 Membuat *Data Frame* berdasarkan Matriks

Perhatikan Gambar 7.23. Pada Gambar 7.23, dibuat suatu matriks bernama **X** dengan jumlah baris sebanyak 3 dan jumlah kolom sebanyak 2. Selanjutnya perintah `R Y <- data.frame(X)` bertujuan untuk membuat *data frame* bernama **Y** berdasarkan data-data pada matriks **X**.

```

Console ~1 ↻
> X <- matrix(c(1,2,3,4,5,6), nrow = 3) #Membuat matriks dengan nama X
> X
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> Y <- data.frame(X) #Membuat data frame bernama Y berdasarkan matriks X
> Y
  x1 x2
1  1  4
2  2  5
3  3  6
> |

```

Gambar 7.23

7.17 Fungsi `is.data.frame()`, `is.list()`, `is.matrix()`, dan `is.vector()`

Pada Gambar 7.24 diperlihatkan penggunaan fungsi `is.data.frame()`, `is.list()`, `is.matrix()`, dan `is.vector()`. Penggunaan fungsi `is.data.frame()` bertujuan untuk mengetahui apakah suatu objek merupakan *data frame* atau tidak. Penggunaan fungsi `is.data.frame()` akan menghasilkan nilai logika TRUE jika suatu objek merupakan *data frame*, FALSE bila suatu objek bukan *data frame*.

- ⇒ Pada Gambar 7.24, dibentuk *data frame* bernama **contoh_dataframe**, *list* bernama **contoh_list**, matriks bernama **contoh_matriks**, dan vektor bernama **contoh_vektor**.
- ⇒ Perintah R **is.data.frame(contoh_dataframe)** menghasilkan nilai TRUE, yang berarti **contoh_dataframe** merupakan *data frame*. Perintah R **is.data.frame(contoh_list)** menghasilkan nilai FALSE, yang berarti **contoh_list** bukan merupakan *data frame*.
- ⇒ Perhatikan bahwa berdasarkan Gambar 7.24, terdapat informasi menarik bahwa suatu *data frame* juga merupakan *list*, yakni pada perintah R **is.list(contoh_dataframe)** yang menghasilkan TRUE.

```

Console ~/
> contoh_dataframe <- data.frame>Nama=c("Ugi", "Egi"), Usia=c(27,23))
> contoh_list <- list>Nama=c("Ugi", "Egi", "Sinta"), Usia=c(27,25,24,29,28))
> contoh_matriks <- matrix(c(1,2,3,4,5,6), ncol=2, byrow=T)
> contoh_vektor <- c(4,3,4)
> contoh_dataframe
  Nama Usia
1  Ugi  27
2  Egi  23
> contoh_list
$Nama
[1] "Ugi" "Egi" "Sinta"

$Usia
[1] 27 25 24 29 28

> contoh_matriks
  [,1] [,2]
[1,]  1  2
[2,]  3  4
[3,]  5  6
> contoh_vektor
[1] 4 3 4
> is.data.frame(contoh_dataframe)
[1] TRUE
> is.data.frame(contoh_list)
[1] FALSE
> is.data.frame(contoh_matriks)
[1] FALSE
> is.data.frame(contoh_vektor)
[1] FALSE
> is.list(contoh_dataframe)
[1] TRUE
> is.list(contoh_list)
[1] TRUE
> is.list(contoh_matriks)
[1] FALSE
> is.list(contoh_vektor)
[1] FALSE
> is.matrix(contoh_dataframe)

```

Gambar 7.24

```

Console ~/1 ↵
$Usia
[1] 27 25 24 29 28

> contoh_matriks
  [,1] [,2]
[1,]  1  2
[2,]  3  4
[3,]  5  6
> contoh_vektor
[1] 4 3 4
> is.data.frame(contoh_dataframe)
[1] TRUE
> is.data.frame(contoh_list)
[1] FALSE
> is.data.frame(contoh_matriks)
[1] FALSE
> is.data.frame(contoh_vektor)
[1] FALSE
> is.list(contoh_dataframe)
[1] TRUE
> is.list(contoh_list)
[1] TRUE
> is.list(contoh_matriks)
[1] FALSE
> is.list(contoh_vektor)
[1] FALSE
> is.matrix(contoh_dataframe)
[1] FALSE
> is.matrix(contoh_list)
[1] FALSE
> is.matrix(contoh_matriks)
[1] TRUE
> is.vector(contoh_vektor)
[1] TRUE
> is.vector(contoh_dataframe)
[1] FALSE

```

Gambar 7.25

7.18 Contoh Kode Program R: Simulasi Distribusi Sampling Rata-Rata Sampel

Berikut diberikan kode program R, untuk menghasilkan distribusi sampling dari rata-rata sampel.

```

library(prob)
seluruh_sampel <- urnsamples(c(1,2,3), size=2, replace=TRUE, ordered=TRUE)
seluruh_sampel
mode(seluruh_sampel)
class(seluruh_sampel)
jumlah_seluruh_sampel <- length(seluruh_sampel$X1)
jumlah_seluruh_sampel
jumlah_kolom <- length(seluruh_sampel)
jumlah_kolom

X1 <- seluruh_sampel[1]
mode(X1)
X1 <- unlist(X1)
mode(X1)
X2 <- seluruh_sampel[2]
mode(X2)
X2 <- unlist(X2)
mode(X2)

X1
X2

names(X1) <- NULL

```

```

names(X2) <- NULL

X1
X2

mode(X1)
mode(X2)

fungsi_rata_rata <- function(X1,X2, jumlah_seluruh_sampel)
{
  rata_rata_sampel <- vector(length=jumlah_seluruh_sampel)
  i=1
  for(n in X1)
  {
    rata_rata_sampel[i] = (X1[i] + X2[i])/2
    i = i+1
  }
  return(rata_rata_sampel)
}

rata_rata_sampel <- fungsi_rata_rata(X1, X2, jumlah_seluruh_sampel)
rata_rata_sampel

barplot(table(rata_rata_sampel))

data_frame <- data.frame(X1, X2, rata_rata_sampel)
data_frame

```

Gambar 7.26 sampai dengan Gambar 7.29 merupakan hasil eksekusi dari kode program R tersebut. Andaikan suatu populasi terdiri dari 3 bilangan, yakni 1, 2, dan 3. Selanjutnya dari 3 bilangan tersebut, dipilih 2, dengan memperhatikan urutan. Maka seluruh kemungkinan sampelnya adalah

(1,1), (1,2), (1,3)
 (2,1), (2,2), (2,3)
 (3,1), (3,2), (3,3)

Perhatikan bahwa terdapat 9 sampel yang mungkin terbentuk. Selanjutnya, dihitung nilai rata-rata dari masing-masing sampel tersebut. Sebagai contoh nilai rata-rata dari (1,1) adalah 2, nilai rata-rata dari (1,2) adalah 1,5, nilai rata-rata dari (1,3) adalah 4, dan seterusnya.

```

Console ~1 ↻
> library(prob)
> seluruh_sampel <- urnsamples(c(1,2,3), size=2, replace=TRUE, ordered=TRUE)
> seluruh_sampel
  x1 x2
1 1 1
2 2 1
3 3 1
4 1 2
5 2 2
6 3 2
7 1 3
8 2 3
9 3 3
> mode(seluruh_sampel)
[1] "list"
> class(seluruh_sampel)
[1] "data.frame"
> jumlah_seluruh_sampel <- length(seluruh_sampel$x1)
> jumlah_seluruh_sampel
[1] 9
> jumlah_kolom <- length(seluruh_sampel)
> jumlah_kolom
[1] 2
>
> x1 <- seluruh_sampel[1]
> mode(x1)
[1] "list"
> x1 <- unlist(x1)
> mode(x1)
[1] "numeric"
> x2 <- seluruh_sampel[2]
> mode(x2)
[1] "list"
> x2 <- unlist(x2)
> mode(x2)
[1] "numeric"

```

Gambar 7.26

```

Console ~1 ↻
> x1 <- unlist(x1)
> mode(x1)
[1] "numeric"
> x2 <- seluruh_sampel[2]
> mode(x2)
[1] "list"
> x2 <- unlist(x2)
> mode(x2)
[1] "numeric"
>
> x1
x11 x12 x13 x14 x15 x16 x17 x18 x19
  1  2  3  1  2  3  1  2  3
> x2
x21 x22 x23 x24 x25 x26 x27 x28 x29
  1  1  1  2  2  2  3  3  3
>
> names(x1) <- NULL
> names(x2) <- NULL
>
> x1
[1] 1 2 3 1 2 3 1 2 3
> x2
[1] 1 1 1 2 2 2 3 3 3
>
> mode(x1)
[1] "numeric"
> mode(x2)
[1] "numeric"
>
> fungsi_rata_rata <- function(x1,x2, jumlah_seluruh_sampel)
+ {
+   rata_rata_sampel <- vector(length=jumlah_seluruh_sampel)
+   i=1
+   for(n in x1)
+   {
+     rata_rata_sampel[i] = (x1[i] + x2[i])/2
+     i = i+1
+   }
+ }

```

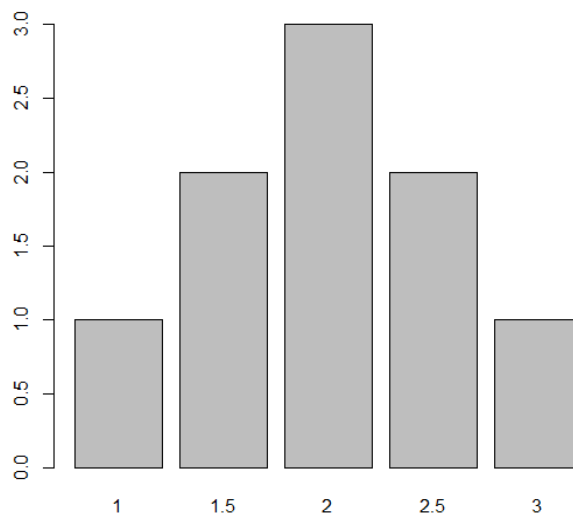
Gambar 7.27

```

Console ~1 ↻
>
> mode(X1)
[1] "numeric"
> mode(X2)
[1] "numeric"
>
> fungsi_rata_rata <- function(X1,X2, jumlah_seluruh_sampel)
+ {
+   rata_rata_sampel <- vector(length=jumlah_seluruh_sampel)
+   i=1
+   for(n in X1)
+   {
+     rata_rata_sampel[i] = (X1[i] + X2[i])/2
+     i = i+1
+   }
+   return(rata_rata_sampel)
+ }
>
> rata_rata_sampel <- fungsi_rata_rata(X1, X2, jumlah_seluruh_sampel)
> rata_rata_sampel
[1] 1.0 1.5 2.0 1.5 2.0 2.5 2.0 2.5 3.0
>
> barplot(table(rata_rata_sampel))
>
> data_frame <- data.frame(X1, X2, rata_rata_sampel)
> data_frame
  X1 X2 rata_rata_sampel
1  1  1             1.0
2  2  1             1.5
3  3  1             2.0
4  1  2             1.5
5  2  2             2.0
6  3  2             2.5
7  1  3             2.0
8  2  3             2.5
9  3  3             3.0
>

```

Gambar 7.28



Gambar 7.29

7.19Berlatih

Gambar 7.30 diberikan *data frame* bernama **simpan_data** yang terdiri dari 4 vektor atau kolom, yakni **Nama**, **Jenis_Kelamin**, **Pendidikan**, dan **Jumlah_Bersaudara**.


```

Console ~/ /
> Nama <- c("Ugi", "Egi", "Andi", "Udin", "Boy", "Sinta", "Rini", "Rani", "Fitri", "Dini", "Rudi", "Firman",
"Agung", "Deni", "Salman", "Fina", "Suci", "wati", "Gina", "Silvia")
> Jenis_Kelamin <- c("Laki-Laki", "Laki-Laki", "Laki-Laki", "Laki-Laki", "Laki-Laki", "Perempuan", "Per
empuan", "Perempuan", "Perempuan", "Laki-Laki", "Laki-Laki", "Laki-Laki", "Laki-Laki", "Laki
-Laki", "Perempuan", "Perempuan", "Perempuan", "Perempuan", "Perempuan")
> Pendidikan <- c("S1", "S2", "S3", "S1", "S2", "S2", "S2", "S1", "S3", "S2", "S1", "S2", "S3", "S1", "S1", "S1"
, "S2", "S3", "S1", "S1")
> Jumlah_Bersaudara <- c(2,3,1,2,5,2,4,2,3,5,2,3,1,2,5,2,4,2,3,1)
> simpan_data <- data.frame>Nama, Jenis_Kelamin, Pendidikan, Jumlah_Bersaudara)
> simpan_data
  Nama Jenis_Kelamin Pendidikan Jumlah_Bersaudara
1  Ugi      Laki-Laki         S1                2
2  Egi      Laki-Laki         S2                3
3  Andi     Laki-Laki         S3                1
4  Udin     Laki-Laki         S1                2
5  Boy      Laki-Laki         S2                5
6  Sinta    Perempuan         S2                2
7  Rini     Perempuan         S2                4
8  Rani     Perempuan         S1                2
9  Fitri    Perempuan         S3                3
10 Dini     Perempuan         S2                5
11 Rudi     Laki-Laki         S1                2
12 Firman  Laki-Laki         S2                3
13 Agung  Laki-Laki         S3                1
14 Deni    Laki-Laki         S1                2
15 Salman Laki-Laki         S1                5
16 Fina    Perempuan         S1                2
17 Suci    Perempuan         S2                4
18 wati    Perempuan         S3                2
19 Gina    Perempuan         S1                3
20 Silvia  Perempuan         S1                1
>

```

Gambar 7.30

Perintah R pada Gambar 7.31 `dim(simpan_data)` bertujuan untuk mengetahui dimensi dari *data frame* `simpan_data`. Diketahui *data frame* `simpan_data` terdiri dari 20 baris (responden) dan 4 kolom (vektor).

```

Console ~/ /
> dim(simpan_data)
[1] 20 4
>

```

Gambar 7.31

Perintah R pada Gambar 7.32 `names(simpan_data)` dan `colnames(simpan_data)` bertujuan untuk mengetahui nama-nama kolom atau vektor.

```

Console ~/ /
> names(simpan_data)
[1] "Nama"      "Jenis_kelamin"  "Pendidikan"    "Jumlah_Bersaudara"
> colnames(simpan_data)
[1] "Nama"      "Jenis_kelamin"  "Pendidikan"    "Jumlah_Bersaudara"
> |

```

Gambar 7.32

Perintah R pada Gambar 7.33 `levels(factor(simpan_data$Pendidikan))` bertujuan untuk mengetahui *level* dari suatu variabel. Pada variabel `Pendidikan` terdiri dari 3 *level*, yakni "S1", "S2", dan "S3". Sementara pada variabel `Jenis_Kelamin` terdiri dari 2 *level*, yakni "Laki-Laki" dan "Perempuan".

```

Console ~/ |
> levels(factor(simpan_data$Pendidikan))
[1] "S1" "S2" "S3"
> levels(factor(simpan_data$Jenis_kelamin))
[1] "Laki-Laki" "Perempuan"
> |

```

Gambar 7.33

Berdasarkan Gambar 7.34, diketahui jumlah responden laki-laki sebanyak 10 dan jumlah responden perempuan sebanyak 10. Perintah R untuk menampilkan distribusi frekuensi untuk data pada variabel **Jenis_Kelamin** adalah `table(factor(simpan_data$Jenis_Kelamin))`. Diketahui jumlah responden yang berpendidikan S1 sebanyak 9, berpendidikan S2 sebanyak 7, dan berpendidikan S3 sebanyak 4. Perintah R untuk menampilkan distribusi frekuensi untuk data pada variabel **Pendidikan** adalah `table(factor(simpan_data$Pendidikan))`.

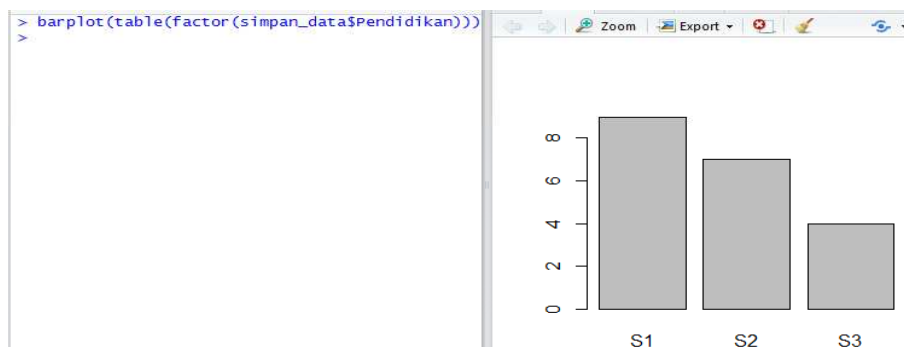
```

Console ~/ |
> table(factor(simpan_data$Jenis_kelamin))
Laki-Laki Perempuan
      10         10
> table(factor(simpan_data$Pendidikan))
S1 S2 S3
 9  7  4
> |

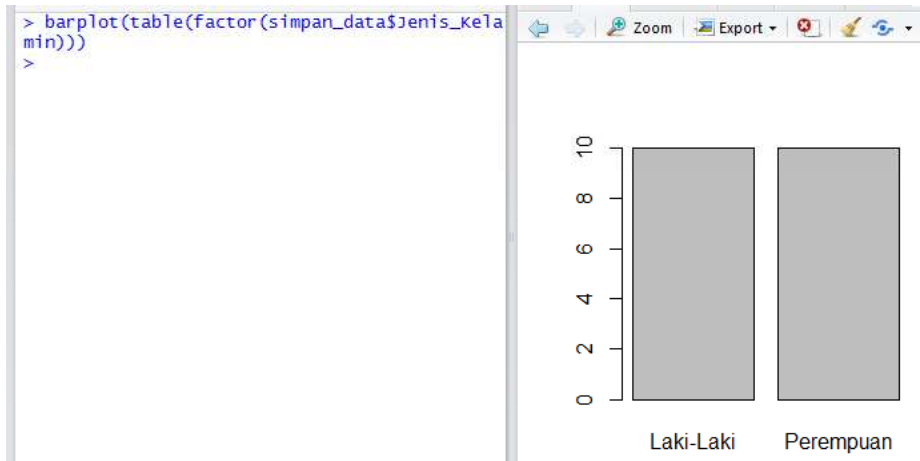
```

Gambar 7.34

Perintah R pada Gambar 7.35 `barplot(table(factor(simpan_data$Jenis_Kelamin)))` bertujuan untuk menyajikan grafik batang frekuensi berdasarkan data pada kolom **Jenis_Kelamin**, sementara perintah R pada Gambar 7.36 `barplot(table(factor(simpan_data$Pendidikan)))` bertujuan untuk menyajikan grafik batang frekuensi berdasarkan data pada kolom **Pendidikan**. Perintah R pada Gambar 7.37 `simpan_data[1]` bertujuan untuk menampilkan data pada kolom ke-1 (**Nama**) dari *data frame* **simpan_data**, atau bisa juga dengan perintah R `simpan_data["Nama"]`. Perintah R pada Gambar 7.38 `simpan_data[c(1,3)]` dan `simpan_data[c("Nama","Pendidikan")]` bertujuan untuk menampilkan data pada kolom ke-1 dan kolom ke-3.



Gambar 7.35



Gambar 7.36

```
> simpan_data[1]
```

	Nama
1	Ugi
2	Egi
3	Andi
4	Udin
5	Boy
6	Sinta
7	Rini
8	Rani
9	Fitri
10	Dini
11	Rudi
12	Firman
13	Agung
14	Deni
15	Salman
16	Fina
17	Suci
18	wati
19	Gina
20	silvia

```
> simpan_data["Nama"]
```

	Nama
1	Ugi
2	Egi
3	Andi
4	Udin
5	Boy
6	Sinta
7	Rini
8	Rani
9	Fitri
10	Dini
11	Rudi
12	Firman
13	Agung
14	Deni
15	Salman
16	Fina
17	Suci
18	wati
19	Gina
20	silvia

Gambar 7.37

```
> simpan_data[c(1,3)]
```

	Nama	Pendidikan
1	Ugi	S1
2	Egi	S2
3	Andi	S3
4	Udin	S1
5	Boy	S2
6	Sinta	S2
7	Rini	S2
8	Rani	S1
9	Fitri	S3
10	Dini	S2
11	Rudi	S1
12	Firman	S2
13	Agung	S3
14	Deni	S1
15	Salman	S1
16	Fina	S1
17	Suci	S2
18	wati	S3
19	Gina	S1
20	silvia	S1

```
> simpan_data[c("Nama", "Pendidikan")]
```

	Nama	Pendidikan
1	Ugi	S1
2	Egi	S2
3	Andi	S3
4	Udin	S1
5	Boy	S2
6	Sinta	S2
7	Rini	S2
8	Rani	S1
9	Fitri	S3
10	Dini	S2
11	Rudi	S1
12	Firman	S2
13	Agung	S3
14	Deni	S1
15	Salman	S1
16	Fina	S1
17	Suci	S2
18	wati	S3
19	Gina	S1
20	silvia	S1

Gambar 7.38

Perintah R pada Gambar 7.39 `simpan_data[5,]` bertujuan untuk menampilkan data pada responden ke-5. Perintah R pada Gambar 7.40 `simpan_data[c(7,12,20),]` bertujuan untuk menampilkan data pada responden ke-7, 12, dan 20.

```
> simpan_data[5,]
  Nama Jenis_Kelamin Pendidikan Jumlah_Bersaudara
5   Boy      Laki-Laki         S2                5
> |
```

Gambar 7.39

```
> simpan_data[c(7,12,20),]
  Nama Jenis_Kelamin Pendidikan Jumlah_Bersaudara
7   Rini      Perempuan         S2                4
12 Firman     Laki-Laki         S2                3
20 Silvia    Perempuan         S1                1
> |
```

Gambar 7.40

```
> simpan_data[c(5:10),]
  Nama Jenis_Kelamin Pendidikan Jumlah_Bersaudara
5   Boy      Laki-Laki         S2                5
6   Sinta    Perempuan         S2                2
7   Rini      Perempuan         S2                4
8   Rani      Perempuan         S1                2
9   Fitri     Perempuan         S3                3
10  Dini      Perempuan         S2                5
> |
```

Gambar 7.41

Perintah R pada Gambar 7.41 `simpan_data[c(5:10),]` bertujuan untuk menampilkan data pada responden ke-5 sampai dengan responden ke-10. Perintah R pada Gambar 7.42 `simpan_data[c(5:10),][c(1,3)]` bertujuan untuk menampilkan data pada responden ke-5 sampai dengan responden ke-10 dan hanya menampilkan kolom **Nama** dan **Pendidikan**.

```
Console ~/ / ↻
> simpan_data[c(5:10),][c(1,3)]
  Nama Pendidikan
5   Boy          S2
6   Sinta        S2
7   Rini         S2
8   Rani         S1
9   Fitri        S3
10  Dini         S2
>
```

Gambar 7.42

```

Console
> simpan_indeks <- which(simpan_data["Pendidikan"]=="S2")
> simpan_data[simpan_indeks,][c("Nama", "Pendidikan")]
  Nama Pendidikan
2   Egi         S2
5   Boy         S2
6  Sinta        S2
7   Rini        S2
10  Dini        S2
12  Firman      S2
17  Suci        S2
> |

Console
> simpan_data["Pendidikan"]=="S2"
Pendidikan
[1,] FALSE
[2,] TRUE
[3,] FALSE
[4,] FALSE
[5,] TRUE
[6,] TRUE
[7,] TRUE
[8,] FALSE
[9,] FALSE
[10,] TRUE
[11,] FALSE
[12,] TRUE
[13,] FALSE
[14,] FALSE
[15,] FALSE
[16,] FALSE
[17,] TRUE
[18,] FALSE
[19,] FALSE
[20,] FALSE
> which(simpan_data["Pendidikan"]=="S2")
[1] 2 5 6 7 10 12 17
> |

```

Gambar 7.43

Gambar 7.43 bertujuan untuk menampilkan responden yang berpendidikan S2. Perhatikan bahwa terdapat 7 responden yang berpendidikan S2, yakni Egi, Boy, Sinta, Rini, Dini, Firman, dan Suci.

- ⇒ Perintah R `simpan_data["Pendidikan"]=="S2"` akan menampilkan nilai logika, yakni TRUE atau FALSE dari masing-masing responden.
- ⇒ Jika responden berpendidikan S2, maka akan menampilkan nilai TRUE, sementara jika responden tidak berpendidikan S2, maka akan menampilkan nilai FALSE.
- ⇒ Berdasarkan Gambar 7.42, responden yang berpendidikan S2 berada pada posisi atau indeks ke-2, 5, 6, 7, 10, 12, dan 17.
- ⇒ Perintah R `which(simpan_data["Pendidikan"]=="S2")` akan menampilkan indeks responden yang berpendidikan S2. Perintah R `simpan_indeks <- which(simpan_data["Pendidikan"]=="S2")` bertujuan menyimpan indeks responden yang berpendidikan S2.
- ⇒ Perintah R `simpan_data[simpan_indeks,][c("Nama", "Pendidikan")]` bertujuan untuk menampilkan responden yang berpendidikan S2, dan hanya menampilkan kolom **Nama** dan **Pendidikan**.

```

Console ~/
> simpan_data["Pendidikan"]=="S1" & simpan_data["Jumlah_Bersaudara"]==2
Pendidikan
[1,] TRUE
[2,] FALSE
[3,] FALSE
[4,] TRUE
[5,] FALSE
[6,] FALSE
[7,] FALSE
[8,] TRUE
[9,] FALSE
[10,] FALSE
[11,] TRUE
[12,] FALSE
[13,] FALSE
[14,] TRUE
[15,] FALSE
[16,] TRUE
[17,] FALSE
[18,] FALSE
[19,] FALSE
[20,] FALSE
> which(simpan_data["Pendidikan"]=="S1" & simpan_data["Jumlah_Bersaudara"]==2)
[1] 1 4 8 11 14 16
> simpan_indeks <- which(simpan_data["Pendidikan"]=="S1" & simpan_data["Jumlah_Bersaudara"]==2)
> simpan_data[simpan_indeks,][c("Nama", "Pendidikan", "Jumlah_Bersaudara")]
  Nama Pendidikan Jumlah_Bersaudara
1  Ugi           S1                2
4  Udin          S1                2
8  Rani          S1                2
11 Rudi          S1                2
14 Deni          S1                2
16 Fina          S1                2
> |

```

Gambar 7.44

Pada Gambar 7.44, bertujuan untuk menampilkan responden yang berpendidikan S1 dengan jumlah bersaudara sebanyak 2. Diketahui terdapat 6 responden yang berpendidikan S1 dengan jumlah bersaudara sebanyak 2, yakni Ugi, Udin, Rani, Rudi, Deni, dan Fina.

```

Console ~/
> simpan_indeks <- which(simpan_data["Pendidikan"]=="S3" & (simpan_data["Jumlah_Bersaudara"]==2 | simpan_data["Jumlah_Bersaudara"]==3) )
> simpan_data[simpan_indeks,][c("Nama", "Pendidikan", "Jumlah_Bersaudara")]
  Nama Pendidikan Jumlah_Bersaudara
9  Fitri          S3                3
18 Wati           S3                2
> |

```

Gambar 7.45

Pada Gambar 7.45 bertujuan untuk menampilkan responden yang berpendidikan S3, dengan jumlah bersaudara 2 atau 3.

```

Console ~/
> simpan_data
  Nama Jenis_Kelamin Pendidikan Jumlah_Bersaudara
1   Ugi   Laki-Laki      S1           2
2   Egi   Laki-Laki      S2           3
3   Andi  Laki-Laki      S3           1
4   Udin  Laki-Laki      S1           2
5   Boy   Laki-Laki      S2           5
6   Sinta Perempuan      S2           2
7   Rini  Perempuan      S2           4
8   Rani  Perempuan      S1           2
9   Fitri Perempuan      S3           3
10  Dini  Perempuan      S2           5
11  Rudi  Laki-Laki      S1           2
12  Firman Laki-Laki      S2           3
13  Agung Laki-Laki      S3           1
14  Deni  Laki-Laki      S1           2
15  Salman Laki-Laki      S1           5
16  Fina  Perempuan      S1           2
17  Suci  Perempuan      S2           4
18  wati  Perempuan      S3           2
19  Gina  Perempuan      S1           3
20  silvia Perempuan      S1           1
> library(dplyr)
> grup <- group_by(simpan_data, Jenis_Kelamin, Pendidikan)
> summarise(grup, Jumlah_Responden = length(Jumlah_Bersaudara) )
source: local data frame [6 x 3]
Groups: Jenis_Kelamin [?]

  Jenis_Kelamin Pendidikan Jumlah_Responden
  <fctr>      <fctr>      <int>
1   Laki-Laki      S1           5
2   Laki-Laki      S2           3
3   Laki-Laki      S3           2
4   Perempuan      S1           4
5   Perempuan      S2           4
6   Perempuan      S3           2
>

```

Gambar 7.46

Berdasarkan Gambar 7.46, diketahui:

- ⇒ Jumlah responden yang berjenis kelamin laki-laki dan berpendidikan S1 sebanyak 5 responden.
- ⇒ Jumlah responden yang berjenis kelamin laki-laki dan berpendidikan S2 sebanyak 3 responden.
- ⇒ Jumlah responden yang berjenis kelamin laki-laki dan berpendidikan S3 sebanyak 2 responden.
- ⇒ Jumlah responden yang berjenis kelamin perempuan dan berpendidikan S1 sebanyak 4 responden.
- ⇒ Jumlah responden yang berjenis kelamin perempuan dan berpendidikan S2 sebanyak 5 responden.
- ⇒ Jumlah responden yang berjenis kelamin perempuan dan berpendidikan S3 sebanyak 2 responden.

```

> simpan_data
  Nama Jenis_kelamin Pendidikan Jumlah_Bersaudara
1   Ugi   Laki-Laki      S1           2
2   Egi   Laki-Laki      S2           3
3   Andi  Laki-Laki      S3           1
4   Udin  Laki-Laki      S1           2
5   Boy   Laki-Laki      S2           5
6   Sinta Perempuan     S2           2
7   Rini  Perempuan     S2           4
8   Rani  Perempuan     S1           2
9   Fitri Perempuan     S3           3
10  Dini  Perempuan     S2           5
11  Rudi  Laki-Laki      S1           2
12  Firman Laki-Laki      S2           3
13  Agung Laki-Laki      S3           1
14  Deni  Laki-Laki      S1           2
15  Salman Laki-Laki      S1           5
16  Fina  Perempuan     S1           2
17  Suci  Perempuan     S2           4
18  wati  Perempuan     S3           2
19  Gina  Perempuan     S1           3
20  Silvia Perempuan     S1           1
> library(dplyr)
> grup <- group_by(simpan_data, Jenis_kelamin, Pendidikan)
> summarise(filter(grup, Jumlah_Bersaudara==3), Jumlah_Responden = length(Jumlah_Bersaudara))
Source: local data frame [3 x 3]
Groups: jenis_kelamin [?]

  Jenis_kelamin Pendidikan Jumlah_Responden
1   <fctr>      <fctr>      <int>
2   Laki-Laki     S2           2
1   Perempuan     S1           1
3   Perempuan     S3           1
>

```

Gambar 7.47

Berdasarkan Gambar 7.47, diketahui:

- ⇒ Terdapat 2 responden berjenis kelamin laki-laki, berpendidikan S2, **dengan jumlah bersaudara sebanyak 3.**
- ⇒ Terdapat 1 responden berjenis kelamin perempuan, berpendidikan S1, **dengan jumlah bersaudara sebanyak 3.**
- ⇒ Terdapat 1 responden berjenis kelamin perempuan, berpendidikan S3, **dengan jumlah bersaudara sebanyak 3.**