

Dasar Pemrograman R

dimas

2023-01-01

Contents

1	Objek dan Paket	5
1.1	Pendefinisian Objek	5
1.2	Paket (Package)	6
2	Tipe Data	7
2.1	Numerik (Numeric)	7
2.2	Bilangan Bulat (Integer)	8
2.3	Bilangan Kompleks (Complex)	8
2.4	Logika (Logical)	9
2.5	Teks (Character)	9
2.6	Faktor (Factor)	9
3	Struktur Data	11
3.1	Vektor (Vector)	11
3.2	Matriks (Matrix)	12
3.3	Daftar (List)	14
3.4	Kerangka Data (Data Frame)	14
4	Operator - operator pada R	17
4.1	Operator Aritmatika	17
4.2	Operator Logika	18
4.3	Operator Relasi	19

5	Plot dengan ggplot2	21
5.1	Bar Plot	21
5.2	Histogram	24
5.3	Pie Plot	26
5.4	Box Plot	28
5.5	Scatter Plot	30
5.6	Line Plot	33
5.7	Area Plot	33

R adalah salah satu bahasa pemrograman yang khusus dirancang dalam menggunakan metode-metode statistika. Program R memberikan banyak kemudahan bagi pemula yang ingin memulai belajar pemrograman. Walaupun program R adalah open source, namun dengan library yang sangat lengkap membuat program R menjadi mampu digunakan untuk mengolah data dengan metode yang lengkap.

Chapter 1

Objek dan Paket

1.1 Pendefinisian Objek

Objek merupakan sebuah wadah untuk menyimpan informasi yang telah didefinisikan oleh pengguna R. Penyimpanan informasi ke dalam sebuah objek memberikan kemudahan bagi pengguna R untuk memanggil informasi yang sama berulang kali hanya dengan memanggil objeknya saja. Seringkali sebuah informasi memiliki banyak komponen, sehingga jika harus memanggilnya berulang kali akan menjadi sangat tidak efektif. Sehingga dengan mendefinisikannya ke dalam sebuah objek akan mempercepat proses pengerjaannya.

Terdapat beberapa cara dalam mendefinisikan objek pada R seperti yang tersajikan pada tabel berikut:

Contoh penggunaannya adalah sebagai berikut:

```
x <- 4
y <- c(1, 4, 6, 8, 5)
z <- c("Indonesia", "Raya")
```

```
x
```

```
## [1] 4
```

Table 1.1: Operator Pendefinisian Objek pada R

Operator	Operasi
<-, «-, =	Mendefinisikan objek sebelah kiri
->, -»	Mendefinisikan objek sebelah kanan

```
y
```

```
## [1] 1 4 6 8 5
```

```
z
```

```
## [1] "Indonesia" "Raya"
```

1.2 Paket (Package)

Chapter 2

Tipe Data

Terdapat 6 tipe data pada bahasa pemrograman R. Setiap tipe data tersebut memiliki karakteristik sendiri sehingga tidak terjadi tumpang tindih dalam melakukan berbagai macam pengoperasiannya. Berikut adalah 6 tipe data pada bahasa pemrograman R:

2.1 Numerik (Numeric)

Tipe data numerik adalah tipe data yang berupa nilai/angka desimal. Tipe data ini merupakan tipe data yang dapat digunakan untuk melakukan operasi-operasi aritmatika seperti penjumlahan, pengurangan, perkalian, dsb. Jika kita mendefinisikan objek `x` dengan suatu nilai/angka, maka tipe objek tersebut adalah `numeric`.

```
x <- 2.6  
class(x)
```

```
## [1] "numeric"
```

Bahkan R akan mendefinisikan objek dengan tipe `numeric` jika berikan angka tanpa desimal.

```
x <- 5  
class(x)
```

```
## [1] "numeric"
```

2.2 Bilangan Bulat (Integer)

Seperti yang kita tahu bahwa pendefinisian angka pada suatu objek akan secara otomatis membuat objek tersebut bertipe `numeric`. Sedangkan untuk mendefinisikan objek dengan tipe `integer` harus mendefinisikan secara khusus objek tersebut dengan perintah `as.integer`.

```
x <- as.integer(5)
class(x)
```

```
## [1] "integer"
```

Selain itu kita dapat mendefinisikan sebuah objek yang bertipe `integer` dengan menambahkan huruf L kapital pada akhir angka.

```
x <- 3L
class(x)
```

```
## [1] "integer"
```

Bagaimana jika kita berikan angka desimal pada objek yang kita definisikan sebagai `integer`?

```
x <- as.integer(3.76)
```

2.3 Bilangan Kompleks (Complex)

Bilangan kompleks dalam matematika adalah bilangan yang didefinisikan dengan $a + bi$, dengan a dan b adalah bilangan real. Sedangkan i adalah bilangan imajiner dan menyebabkan bi menjadi imajiner. Bilangan imajiner sendiri memiliki sifat $i^2 = -1$. Kita harus secara langsung mendefinisikan objek sebagai bilangan kompleks agar mendapatkan sebuah objek yang bertipe `complex`.

```
x <- 2 + 4i
class(x)
```

```
## [1] "complex"
```


2.4 Logika (Logical)

Objek dengan tipe logika adalah objek yang hanya memiliki 2 nilai saja yaitu TRUE dan FALSE.

```
x <- TRUE
class(x)
```

```
## [1] "logical"
```

2.5 Teks (Character)

Pendefinisian objek dengan tipe teks (**character**) merupakan hal yang cukup mudah. Kita perlu menambahkan tanda petik " pada awal dan akhir teks. Setelah itu objek akan terdefiniskan sebagai **character**.

```
x <- "Aplikasi Komputer"
class(x)
```

```
## [1] "character"
```

```
x
```

```
## [1] "Aplikasi Komputer"
```

Apakah tipe data dari objek yang didefinisikan dengan nilai "2.4"?

2.6 Faktor (Factor)

Faktor adalah tipe data pada bahasa pemrograman R yang digunakan untuk mendefinisikan sebuah objek menjadi sebuah objek dengan tipe data kategorik. Perintah yang digunakan untuk merubah sebuah objek menjadi sebuah faktor adalah **factor()**.

```
x <- factor(c(1, 2, 3))
y <- factor(c("SD", "SMP", "SMA", "SMA", "PT"))
x
```

```
## [1] 1 2 3
## Levels: 1 2 3
```

```
y
```

```
## [1] SD  SMP SMA SMA PT  
## Levels: PT SD SMA SMP
```

Sebuah objek yang telah didefinisikan sebagai faktor akan memiliki `levels` yang merupakan daftar kategori yang terdapat pada objek tersebut. Kita dapat menggunakan perintah `levels` untuk dapat memunculkan `levels` nya saja.

```
levels(x)
```

```
## [1] "1" "2" "3"
```

```
levels(y)
```

```
## [1] "PT" "SD" "SMA" "SMP"
```

- Apa perbedaan *output* dari objek yang bertipe **factor** dengan objek yang bertipe **numeric**?
- Apa perbedaan *output* dari objek yang bertipe **factor** dengan objek yang bertipe **character**?

Chapter 3

Struktur Data

Struktur data pada bahasa pemrograman R adalah sebuah konsep yang digunakan untuk menyimpan data berdasarkan kebutuhannya. Berdasarkan struktur datanya, terdapat 4 struktur data yang dapat kita definisikan pada program R:

- Vektor (`vector`)
- Matriks (`matrix`)
- Pendaftaran (`list`)
- Kerangka Data (`data.frame`)

3.1 Vektor (Vector)

Vektor adalah struktur data yang paling sederhana pada bahasa pemrograman R. Vektor memuat barisan data dengan tipe data yang sama. Pendefinisian vektor dilakukan dengan menggunakan perintah `c()`. Semua data ditulis didalam tanda kurung dan setiap data dipisahkan dengan tanda koma ,.

```
x <- c(1:5)
x
```

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

```
y <- c("Hipertensi", "Diabetes", "Asam Urat")
y
```

```
## [1] "Hipertensi" "Diabetes"    "Asam Urat"
```

```
length(x)
```

```
## [1] 5
```

```
length(y)
```

```
## [1] 3
```

Apa yang terjadi apabila sebuah vektor diisi dengan 2 tipe data? misalnya `c(12, 4, TRUE)`

Selanjutnya kita dapat memanggil anggota dari sebuah objek `vector` dengan menggunakan tanda `[x]` setelah objek dengan `x` adalah nilai yang menyatakan *data ke-x*.

```
x[2]
```

```
## [1] 2
```

```
y[3]
```

```
## [1] "Asam Urat"
```

3.2 Matriks (Matrix)

Sama halnya dengan vektor, matriks merupakan struktur data yang hanya dapat menyimpan 1 tipe data saja. Perbedaan antara struktur data ‘matrix’ dengan ‘vector’ berada pada dimensi datanya. Vektor merupakan struktur data berdimensi 1 (hanya memiliki panjang data). Sedangkan matriks adalah struktur data yang berdimensi 2 (memiliki dimensi dan panjang data).

Pendefinisian matriks dilakukan dengan menggunakan perintah `matrix` dengan syntax: `matrix(data, nrow, ncol, byrow=FALSE)`.

```
x <- matrix(
  c(1:8), 2, 4
)
x
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    5    7
## [2,]    2    4    6    8
```

```
dim(x)
```

```
## [1] 2 4
```

dengan:

- `data` adalah data yang akan kita gunakan
- `nrow` adalah jumlah baris
- `ncol` adalah jumlah kolom
- `byrow` adalah perintah opsional untuk memilih agar data yang kita miliki didaftarkan berdasarkan baris atau kolom

Apakah yang akan terjadi apabila kita menggunakan `byrow=FALSE` dan `byrow=TRUE`?

Berbeda dengan objek yang berupa `vector`, pemanggilan data pada objek `matrix` menggunakan tanda `[x,y]` dimana `x` adalah *data baris ke-x* dan `y` adalah *data kolom ke-y*. Apabila kita menggunakan tanda `[x,]` saja, maka kita memanggil semua data pada *baris ke-x*. Sedangkan apabila kita menggunakan tanda `[,y]` saja, maka kita memanggil semua data pada *kolom ke-y*.

```
x[2,3]
```

```
## [1] 6
```

```
x[2,]
```

```
## [1] 2 4 6 8
```

```
x[,4]
```

```
## [1] 7 8
```

```
x[6]
```

```
## [1] 6
```

Catatan: `x` dan `y` pada `[x,y]` dapat berupa `vector`.

3.3 Daftar (List)

Struktur data `list` dalam R adalah struktur data yang dapat mendaftarkan beberapa objek sekaligus tanpa perlu khawatir dengan tipe data yang berbeda. Struktur data ‘list’ dapat juga dikatakan sebagai vektor yang dapat menyimpan berbagai macam objek. Pendefinisian list dilakukan dengan menggunakan perintah `list()`.

```
a <- c(1, 3, 6, 2)
b <- c("Apa", "kabar", "anda", "hari", "ini", "?")
c <- c(TRUE, FALSE, FALSE, FALSE, TRUE)
x <- list(a=a, b=b, c=c)
x
```

```
## $a
## [1] 1 3 6 2
##
## $b
## [1] "Apa" "kabar" "anda" "hari" "ini" "?"
##
## $c
## [1] TRUE FALSE FALSE FALSE TRUE
```

Selanjutnya kita dapat memanggil masing-masing objek pada sebuah list dengan syntax: `ListObject$ObjekInList` atau dengan menggunakan `[[x]]` dimana `x` adalah sebuah angka yang merujuk pada **daftar ke berapa**.

```
x$a
```

```
## [1] 1 3 6 2
```

```
x[[3]]
```

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

3.4 Kerangka Data (Data Frame)

Struktur data yang berupa *data frame* (`data.frame`) merupakan struktur data yang akan paling sering kita gunakan dalam pengolahan data. Struktur data ini digunakan untuk mendefinisikan sebuah tabel data yang mana setiap kolom adalah nama-nama objek/variabel pada *data frame* tersebut. Setiap objek/variabel dalam `data.frame` merupakan sebuah **vector**. Artinya setiap

Table 3.1: Data Mahasiswa 2020

nama	jk	tb	bb
Ahmad	Laki-laki	170	70
Ganjar	Laki-laki	169	67
Lusi	Perempuan	160	45
Andina	Perempuan	154	40
Elok	Perempuan	163	52

objek/variabel dalam `data.frame` hanya dapat memiliki 1 tipe data saja. Selain itu setiap objek/variabel yang berada dalam `data.frame` harus memiliki jumlah data (`length()`) yang sama.

Perintah `data.frame()` adalah perintah yang digunakan untuk mendefinisikan sebuah objek sebagai sebuah data frame.

```
nama <- c("Ahmad", "Ganjar", "Lusi", "Andina", "Elok")
jk   <- factor(c("Laki-laki", "Laki-laki", "Perempuan", "Perempuan", "Perempuan"))
tb   <- c(170, 169, 160, 154, 163)
bb   <- c(70, 67, 45, 40, 52)
x    <- data.frame(nama, jk, tb, bb)
x
```

```
##      nama      jk  tb bb
## 1  Ahmad Laki-laki 170 70
## 2  Ganjar Laki-laki 169 67
## 3   Lusi Perempuan 160 45
## 4 Andina Perempuan 154 40
## 5   Elok Perempuan 163 52
```

Selanjutnya untuk memanggil objek/variabel pada sebuah `data.frame` mirip dengan memanggil data pada sebuah `matrix`. Selain itu kita dapat pula memanggil dengan cara menyebutkan nama objek/variabel pada `data.frame` tersebut.

```
x[,2]
```

```
## [1] Laki-laki Laki-laki Perempuan Perempuan Perempuan
## Levels: Laki-laki Perempuan
```

```
x[4,]
```

```
##      nama      jk  tb bb
## 4 Andina Perempuan 154 40
```

```
x$nama
```

```
## [1] "Ahmad" "Ganjar" "Lusi" "Andina" "Elok"
```

```
x[,c('jk', 'nama')]
```

```
##           jk    nama
## 1 Laki-laki  Ahmad
## 2 Laki-laki  Ganjar
## 3 Perempuan Lusi
## 4 Perempuan Andina
## 5 Perempuan Elok
```


Chapter 4

Operator - operator pada R

Operator - operator pada R adalah perintah untuk melakukan suatu operasi seperti operasi aritmatika, logika dan relasi. Secara umum terdapat 4 tipe operator pada R, yaitu:

- Operator Aritmatika
- Operator Logika
- Operator Relasi

4.1 Operator Aritmatika

Operator aritmatika adalah operator yang digunakan untuk melakukan komputasi dengan menggunakan operasi - operasi matematika seperti penjumlahan, pengurangan, dll. Berikut adalah daftar operator aritmatika pada R:

Operator umumnya digunakan untuk pengoperasian pada objek bertipe `numeric` atau `integer` dengan struktur data berupa `vector` dan `matrix`.

Table 4.1: Operator Aritmatika pada R

Operator	Operasi
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
^	Perpangkatan
%%	Modulo

Table 4.2: Hasil Penggunaan Operator pada R

Operator	Operasi	Hasil
+	$x + y$	8.0
-	$x - y$	-2.0
*	$x * y$	15.0
/	x / y	0.6
^	$x ^ y$	243.0
%%	$x \% y$	3.0

Table 4.3: Operator Logika pada R

Operator	Operasi
!	Bukan/Ingkaran
&	Logika DAN pada masing-masing anggota objek
&&	Logika DAN
	Logika ATAU
	Logika ATAU pada masing-masing anggota objek

Berikut adalah contoh penggunaannya:

```
x <- 3
y <- 5
```

4.2 Operator Logika

Operator logika akan menghasilkan nilai `TRUE` dan `FALSE`. Seperti halnya logika matematika, terdapat 2 operator logika yaitu *dan* (`AND`), *atau* (`OR`) dan ingkaran (`NOT`). Berikut adalah daftar operator logika pada R:

Berikut adalah contoh hasil penggunaannya:

```
x <- c(TRUE, FALSE, 5)
y <- c(TRUE, FALSE, FALSE)
z <- c(FALSE, TRUE, TRUE)

!x
```

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

Table 4.4: Hasil Penggunaan Operator Logika

P1	P2	NegasiP1	NegasiP2	DAN	ATAU
TRUE	TRUE	FALSE	FALSE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE	FALSE	TRUE
FALSE	TRUE	TRUE	FALSE	FALSE	TRUE
FALSE	FALSE	TRUE	TRUE	FALSE	FALSE

```
x&y
```

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

```
x&&z
```

```
## Warning in x && z: 'length(x) = 3 > 1' in coercion to 'logical(1)'
```

```
## Warning in x && z: 'length(x) = 3 > 1' in coercion to 'logical(1)'
```

```
## [1] FALSE
```

```
x|y
```

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

```
x||z
```

```
## Warning in x || z: 'length(x) = 3 > 1' in coercion to 'logical(1)'
```

```
## [1] TRUE
```

4.3 Operator Relasi

Operator relasi adalah operator yang digunakan untuk membandingkan 2 buah nilai/objek. Hasil *output* dari operator relasi ini adalah **TRUE** dan **FALSE**. Berikut adalah daftar operator relasi:

Operator relasi dapat dioperasikan layaknya operator logika. Berikut adalah contoh penggunaannya:

Table 4.5: Operasi Relasi pada R

Operator	Operasi
<	Kurang dari
<=	Kurang dari sama dengan
>	Lebih dari
>=	Lebih dari sama dengan
==	Sama dengan
!=	Tidak sama dengan

```
x <- 3  
y <- 7  
z <- 7  
x<y
```

```
## [1] TRUE
```

```
x>y
```

```
## [1] FALSE
```

```
x<=y
```

```
## [1] TRUE
```

```
y<=z
```

```
## [1] TRUE
```

```
x!=y
```

```
## [1] TRUE
```

```
y==z
```

```
## [1] TRUE
```

Chapter 5

Plot dengan ggplot2

Plot adalah salah satu hal mendasar yang dapat digunakan untuk menggambarkan distribusi data. Program R pun memiliki perintah - perintah untuk membuat sebuah plot. Namun perintah - perintah tersebut masih merupakan perintah yang sederhana. Hasilnya, gambar kurang menarik untuk disajikan pada khalayak umum.

`ggplot2` merupakan sebuah paket yang dapat digunakan pada program R dengan cara melakukan instalasi terlebih dahulu menggunakan perintah `install.package("ggplot2")`. Paket `ggplot2` memberikan fasilitas bagi penggunaanya dalam membuat sebuah plot yang menarik untuk dilihat. Setelah anda memahami jenis-jenis plot beserta cara penggunaannya, kita dapat menggunakan paket `ggpubr` yang telah disusun khusus untuk kebutuhan publikasi/penerbitan.

Setelah instalasi paket `ggplot2` selesai, selanjutnya kita dapat memanggil paket tersebut dengan perintah

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.1
```

5.1 Bar Plot

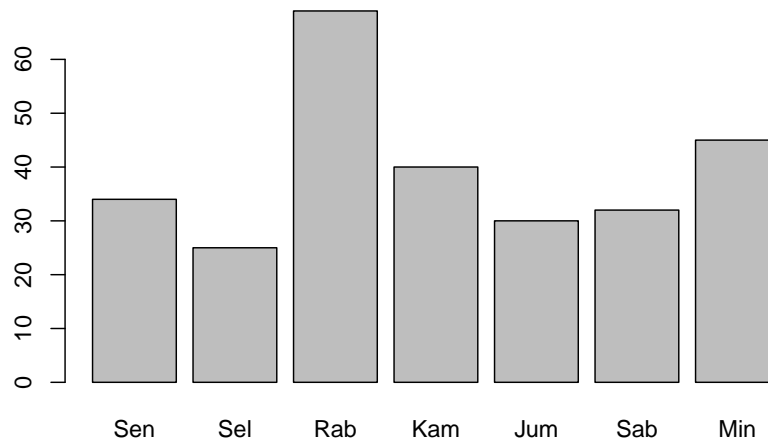
Grafik batang merupakan grafik yang digunakan untuk menunjukkan frekuensi dari data. Grafik batang lebih sering digunakan untuk menunjukkan perbedaan jumlah dari data yang bertipe kategorik seperti jenis kelamin, asal daerah, status pernikahan, dsb. Selanjutnya kita akan membuat sebuah grafik batang dengan menggunakan data berikut:

```
jpc <- c(34, 25, 69, 40, 30, 32, 45)
hr  <- c("Sen", "Sel", "Rab", "Kam", "Jum", "Sab", "Min")
x   <- data.frame(Hari = hr, Pasien = jpc)
x
```

```
##   Hari Pasien
## 1 Sen      34
## 2 Sel      25
## 3 Rab      69
## 4 Kam      40
## 5 Jum      30
## 6 Sab      32
## 7 Min      45
```

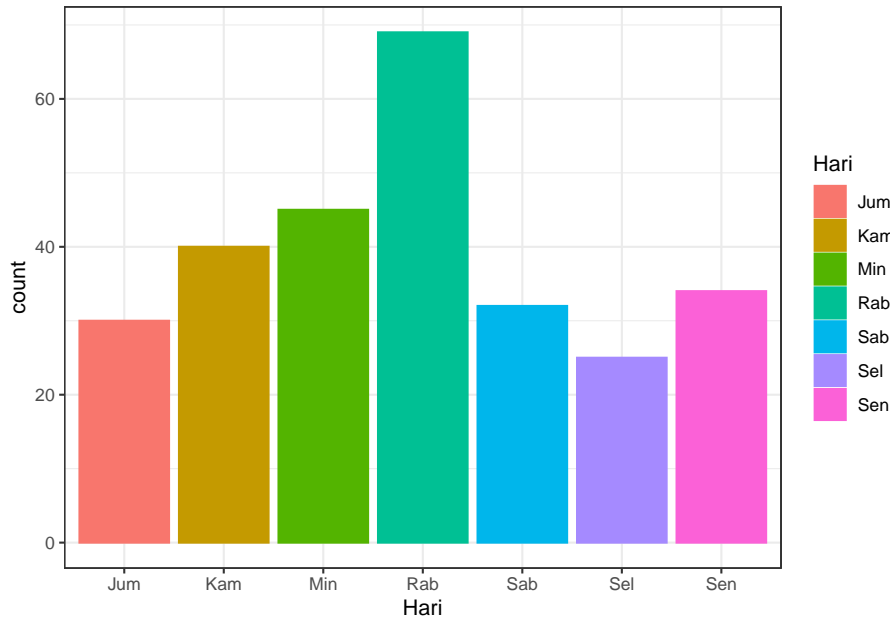
Terdapat perintah `barplot` dalam R yang merupakan perintah yang sudah tersedia saat melakukan instalasi R. Walaupun perintah `barplot` dapat kita gunakan, namun hasil yang diberikan sangat sederhana dan lebih cocok digunakan untuk sebuah laporan. Berikut adalah cara menggunakan perintah `barplot`:

```
barplot(x$Pasien,
        names.arg = x$Hari)
```



Selanjutnya kita akan menggunakan paket `ggplot2` untuk menyajikan grafik batang yang lebih menarik. Berikut adalah perintah untuk membuat grafik batang dengan menggunakan paket `ggplot2`:

```
p <- ggplot(x, aes(Hari)) +  
  geom_bar(aes(weight=Pasien, fill=Hari, colour=Hari)) +  
  theme_bw()  
p
```



penjelasan:

- `ggplot(x, aes(Hari))` adalah perintah untuk membuat sebuah objek `ggplot` dari variabel `Hari` pada data `x`
- `geom_bar(aes(weight=Pasien, fill=Hari, colour=Hari))`
 - `geom_bar()` adalah perintah untuk membuat grafik batang menggunakan `ggplot`
 - `weight` adalah banyak datanya (dalam kasus yang kita kerjakan: banyaknya Pasien setiap hari)
 - `fill` bertujuan untuk memberi warna batang (harus sama dengan `aes(Hari)` pada `ggplot` agar setiap batang memiliki warna yang berbeda)
 - `colour` bertujuan untuk memberi warna garis (harus sama dengan `aes(Hari)` pada `ggplot` agar setiap batang memiliki warna yang berbeda)
- `theme_bw()` bertujuan untuk menentukan tema **black and white** pada grafik

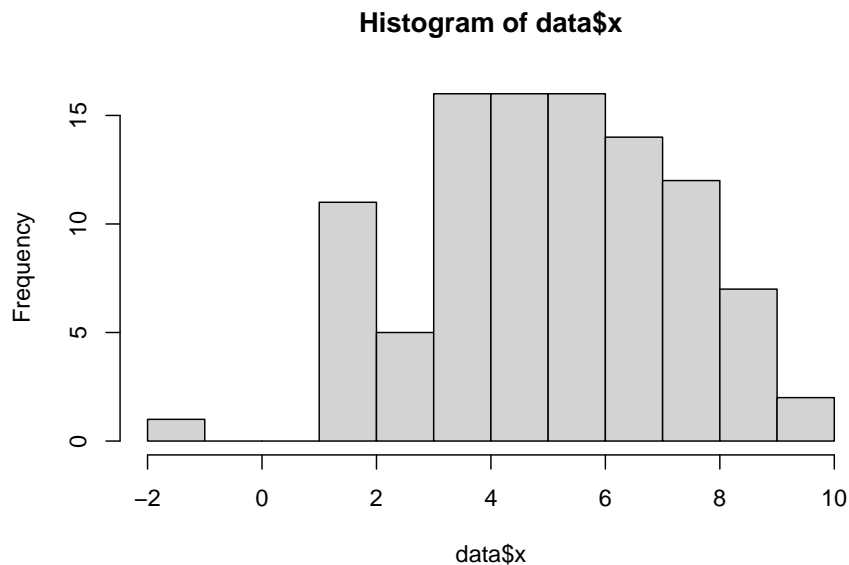
5.2 Histogram

Histogram merupakan grafik batang yang dapat menunjukkan seberapa sering suatu nilai yang berbeda terjadi. Histogram lebih sering digunakan untuk melihat distribusi dari suatu data. Berbeda dengan grafik batang, kita perlu menggunakan data numerik dalam membuat sebuah histogram. Berikut adalah data acak yang dibangkitkan dengan perintah `rnorm`:

```
data <- data.frame(x = rnorm(100,5,2))
```

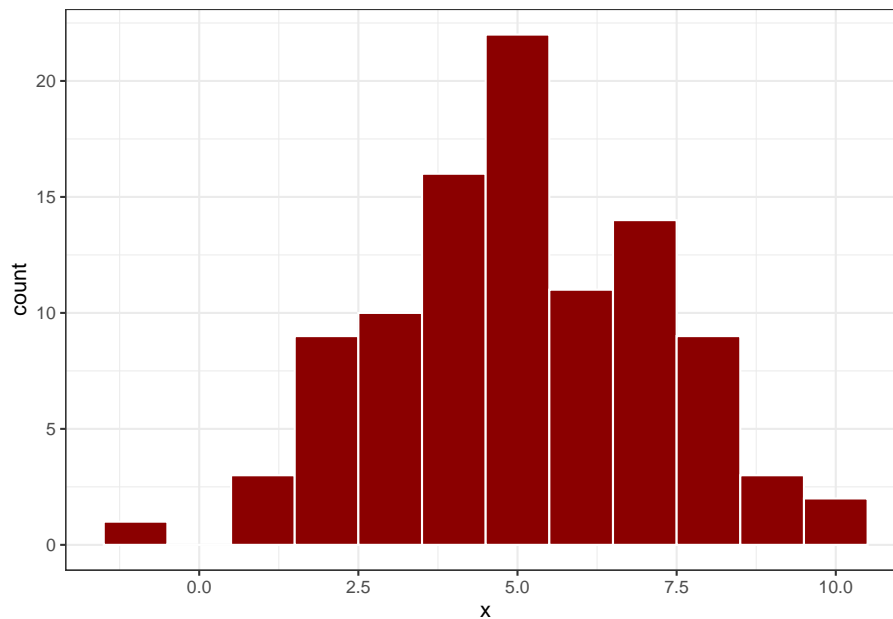
Kita dapat menggunakan perintah `hist` yang telah tersedia pada program R saat melakukan instalasi program R.

```
hist(data$x)
```



Selanjutnya kita akan menggunakan perintah yang tersedia pada paket `ggplot2`.

```
p <- ggplot(data, aes(x)) +
  geom_histogram(binwidth = 1,
                 color = "white",
                 fill = "darkred") +
  theme_bw()
p
```

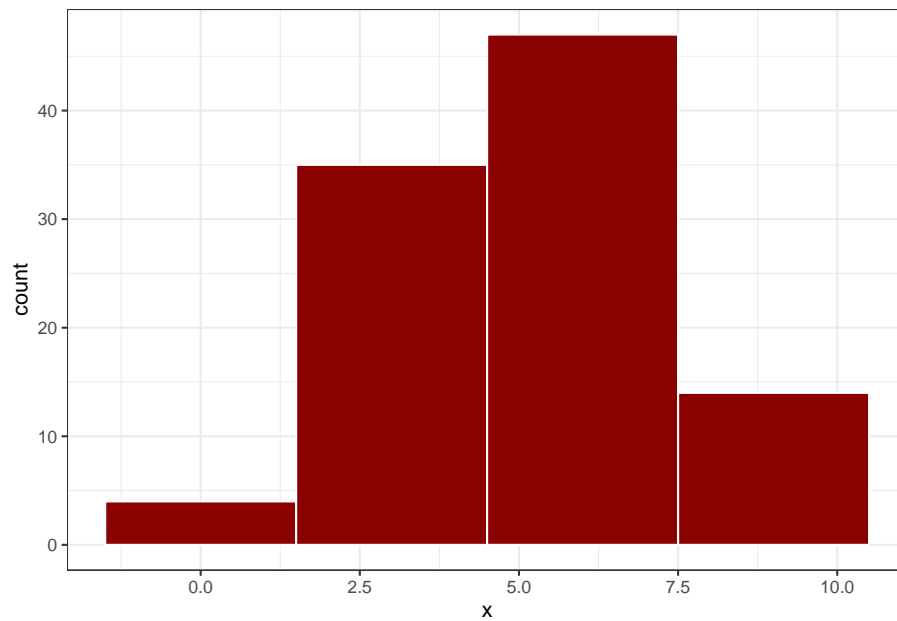



penjelasan:

- `ggplot(data, aes(x))` adalah perintah untuk membuat sebuah objek `ggplot` dari objek `x` pada data `data`
- `geom_histogram(binwidth = 1, color = "white", fill= "darkred")`
 - `geom_histogram()` adalah perintah untuk membuat histogram menggunakan `ggplot`
 - `bandwidth` adalah lebar dari masing-masing batang
 - `fill` bertujuan untuk memberi warna batang (dalam kasus ini kita akan berikan warna yang sama untuk semua batang)
 - `colour` bertujuan untuk memberi warna garis (dalam kasus ini kita akan berikan warna yang sama untuk semua garis)
- `theme_bw()` bertujuan untuk menentukan tema **black and white** pada plot

Selanjutnya bandingkan dengan histogram yang memiliki `bandwidth` berbeda.

```
p <- ggplot(data, aes(x)) +
  geom_histogram(binwidth = 3,
                 color = "white",
                 fill= "darkred") +
  theme_bw()
p
```



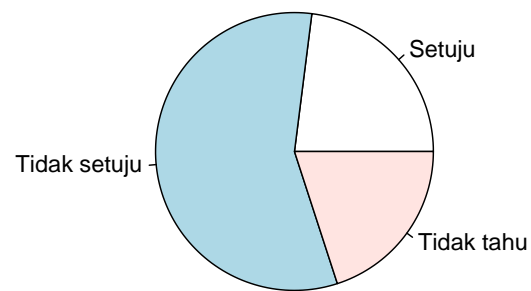
5.3 Pie Plot

Pie plot atau diagram lingkaran lebih sering digunakan untuk menunjukkan perbandingan antar kategori pada sebuah variabel. Berikut ini adalah data yang akan kita gunakan untuk membuat diagram lingkaran:

```
jumlah <- c(23, 57, 20)
label <- c("Setuju", "Tidak setuju", "Tidak tahu")
x <- data.frame(label, jumlah)
```

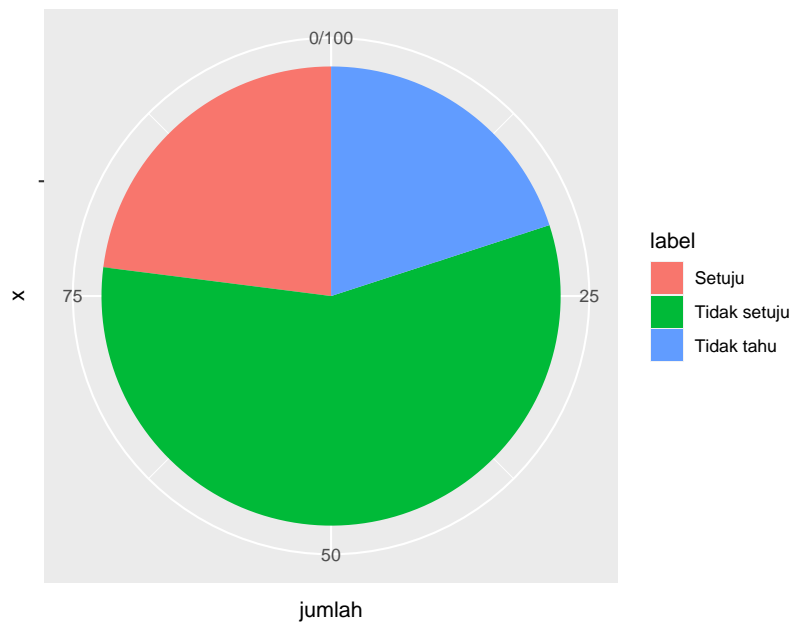
Menggunakan perintah `pie`

```
pie(x$jumlah, labels = x$label)
```



Selanjutnya berikut ini adalah perintah untuk membuat diagram lingkaran dengan menggunakan paket `ggplot2`

```
p <- ggplot(x, aes(x="", y=jumlah, fill=label)) +  
  geom_bar(stat = "identity")  
p + coord_polar("y", start = 0)
```



penjelasan:

- `ggplot(x, aes(x="", y=jumlah, fill=label))` adalah perintah untuk membuat sebuah objek `ggplot` dengan data `x` dengan nilai `y` adalah jumlah dan `label` menjadi warna pembeda di setiap nilai pada `y`
- `geom_bar(stat = "identity")`
 - `geom_bar()` adalah perintah untuk membuat grafik batang menggunakan `ggplot`
 - `stat = "identity"` bertujuan untuk membuat grafik batang dalam bentuk stacked bar chart
- `coord_polar("y", start = 0)` bertujuan untuk membentuk stacked pada koordinat polar mengikuti `angle = y` dan dimulai dari titik nol `start = 0`

5.4 Box Plot

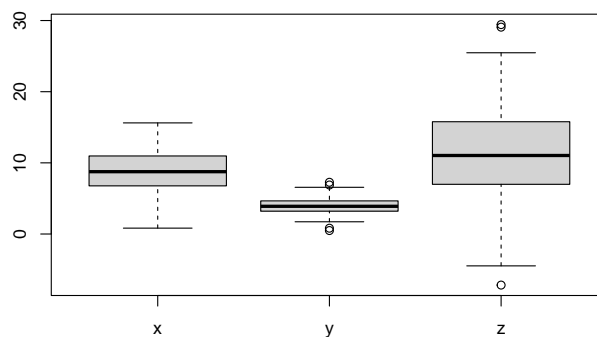
Box Plot dapat menggambarkan sebaran data yang bersifat **numerik**. Box plot dibuat dengan menggunakan nilai-nilai pada ukuran pemusatan data. Nilai-nilai tersebut ialah **1st quartile**, **median**, **3rd quartile**, dan **interquartile range**. Selanjutnya kita akan menggunakan 3 data berbeda yang bersebar normal untuk menunjukkan keunggulan dalam menggunakan box plot.

```
x <- rnorm(250, 9, 3)
y <- rnorm(250, 4, 1)
z <- rnorm(250, 11, 6)
```

Menggunakan perintah boxplot

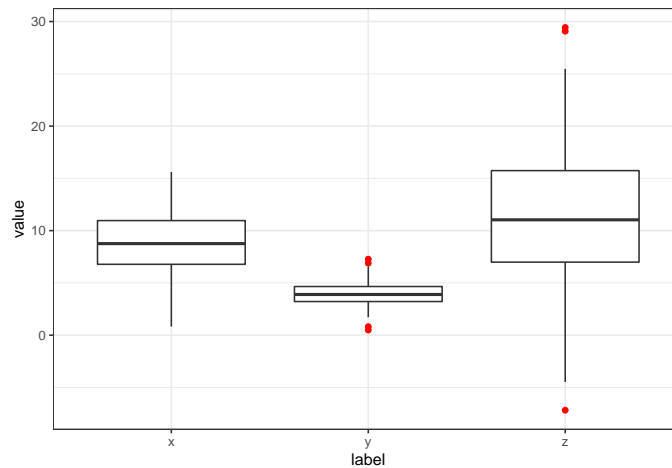
```
boxplot(x, y, z,
        names = c("x", "y", "z"))
data <- data.frame(label = c(rep(c("x", "y", "z"),
                                each=250)),
                   value = c(x, y, z))
head(data)
```

```
##  label    value
## 1     x 14.324879
## 2     x  9.036283
## 3     x 11.663369
## 4     x 13.613847
## 5     x  7.452330
## 6     x 12.950421
```



Menggunakan paket ggplot2

```
p <- ggplot(data, aes(x=label, y=value)) +
  geom_boxplot(outlier.colour = "red") +
  theme_bw()
p
```



penjelasan:

- `ggplot(data, aes(x=label, y=value))` adalah perintah untuk membuat sebuah objek `ggplot` dengan nilai `value` sebagai `y` dan `label` menjadi warna pembeda di setiap nilai `y` menggunakan data `data`
- `geom_boxplot(outlier.colour = "red")`
 - `geom_boxplot()` adalah perintah untuk membuat box plot menggunakan `ggplot`
 - `outlier.colour = "red"` memberikan warna merah pada garis box plot (optional)

Menggunakan paket `ggplot2`

5.5 Scatter Plot

Scatter Plot (diagram pencar) merupakan grafik yang dibuat dengan tujuan untuk memasang 2 nilai/variabel. Diagram pencar dapat menggambarkan hubungan dari 2 nilai/variabel tersebut. Apabila titik-titik dalam diagram pencar menuju kearah kanan atas dari kiri bawah, maka terdapat hubungan yang positif dari kedua nilai/variabel begitu pula sebaliknya. Kita akan menggunakan data berikut untuk menunjukkan hubungan yang positif dan negatif dari 2 variabel.

```
n <- 10
r <- 0.75

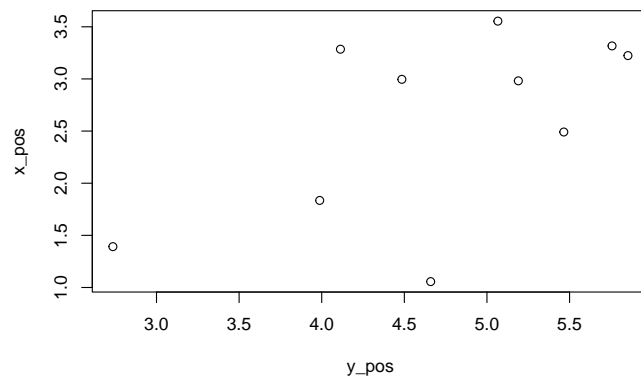
library(MASS)
```

```
data_pos <- mvrnorm(n, mu=c(3, 5), Sigma=matrix(c(1, r, r, 1), nrow=2))
x_pos <- data_pos[, 1]
y_pos <- data_pos[, 2]

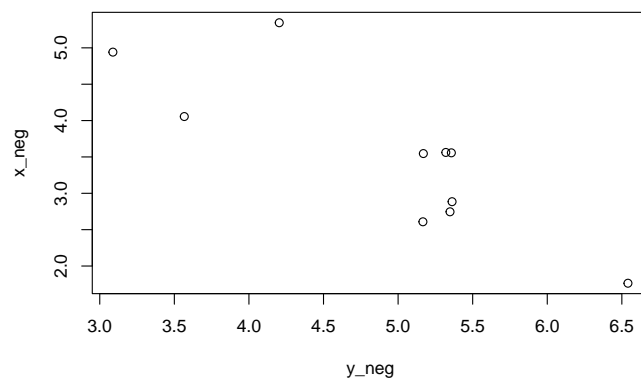
data_neg <- mvrnorm(n, mu=c(3, 5), Sigma=matrix(c(1, -r, -r, 1), nrow=2))
x_neg <- data_neg[, 1]
y_neg <- data_neg[, 2]
```

Menggunakan perintah plot pada program R:

```
plot(y_pos, x_pos)
```



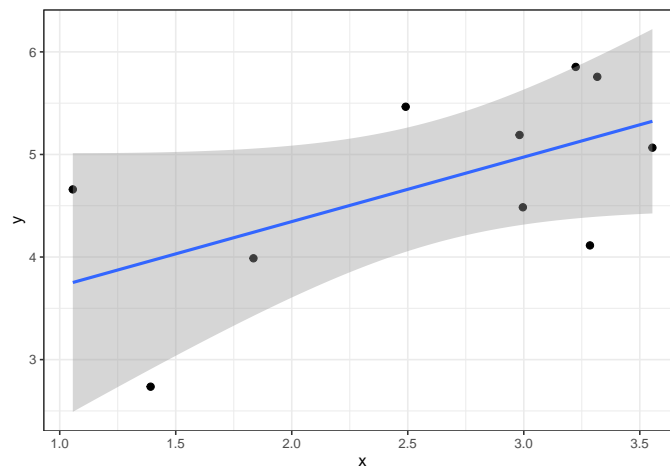
```
plot(y_neg, x_neg)
```



Gambar pertama menunjukkan adanya hubungan yang positif pada variabel `y_pos` dan `x_pos`. Sedangkan pada gambar kedua adalah sebaliknya, yaitu hubungan yang negatif. selanjutnya kita akan membangun diagram pencar menggunakan paket `ggplot2`.

```
data_pos <- data.frame(data_pos)
colnames(data_pos) = c("x", "y")
p <- ggplot(data_pos, aes(x,y))
p + geom_point(size=2) +
  geom_smooth(method=lm) +
  theme_bw()
```

```
## `geom_smooth()` using formula 'y ~ x'
```



penjelasan:

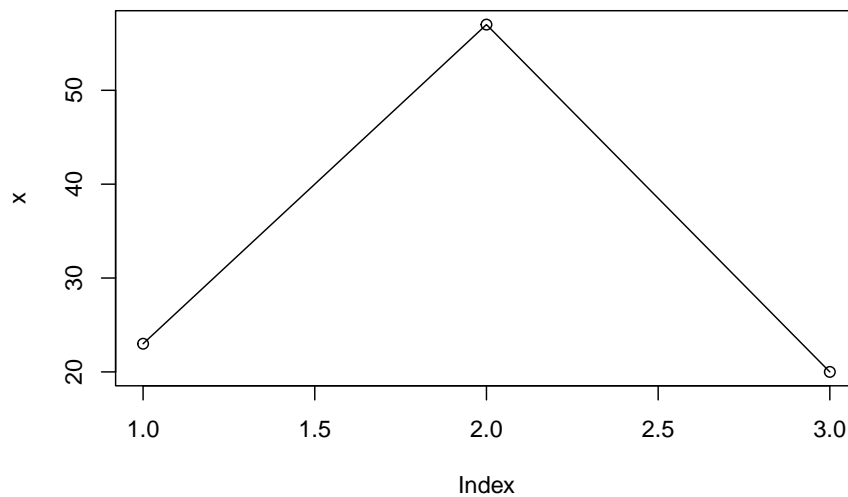
- `data_pos` adalah objek yang memuat `data_pos` baru setelah diubah kedalam bentuk `data frame`
- `colnames()` adalah perintah untuk memberikan nama kolom/variabel pada `data frame`
- `ggplot(data_pos, aes(x,y))` adalah perintah untuk membuat sebuah objek `ggplot` dengan nilai `value` sebagai `y` dan `label` menjadi warna berbeda di setiap nilai `y` menggunakan data `data`
- `geom_point()` adalah perintah untuk membuat diagram pencar
 - `size=2` adalah perintah untuk mengubah ukuran titik diagram pencar menjadi sebesar 2 poin
- `geom_smooth(method=lm)` adalah perintah untuk membuat garis regresi linier sebagai penegas hubungan antara 2 variabel (optional)

5.6 Line Plot

```
x <- c(23, 57, 20)
names(x) <- c("Setuju", "Tidak setuju", "Tidak tahu")
```

Menggunakan perintah `pie`

```
plot(x, type = "o")
```



5.7 Area Plot

```
data("diamonds")
head(diamonds)
```

```
## # A tibble: 6 x 10
##   carat cut      color clarity depth table price      x      y      z
##   <dbl> <ord>    <ord> <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23 Ideal    E     SI2     61.5   55   326  3.95  3.98  2.43
## 2  0.21 Premium  E     SI1     59.8   61   326  3.89  3.84  2.31
```

```
## 3 0.23 Good      E      VS1      56.9      65      327      4.05      4.07      2.31
## 4 0.29 Premium   I      VS2      62.4      58      334      4.2       4.23      2.63
## 5 0.31 Good      J      SI2      63.3      58      335      4.34      4.35      2.75
## 6 0.24 Very Good J      VVS2     62.8      57      336      3.94      3.96      2.48
```

Note: data `diamonds` adalah data yang sudah tersedia dalam program R

Menggunakan paket `ggplot2`

```
p <- ggplot(diamonds, aes(x = price, fill = cut))
p + geom_area(stat = "bin")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

