# Modul 3

Ahmad Abdullah Azzam

9/19/2021

**A. Tujuan Praktikum**

Memahami jenis-jenis tipe data pada R

**B. Alokasi Waktu**

1 x pertemuan = 120 menit

**C. Dasar Teori**

Variasi tipe data pada R memfasilitasi keberagaman jenis variabel data. Sebagai contoh, terdapat data yang terdiri dari sekumpulan angka dan data lain yang berisi sekumpulan karakter. Pada contoh lain, ada pula data yang berbentuk tabel maupun kumpulan (*list*) angka sederhana. Dengan bantuan fungsi `class`, kita akan mendapatkan kemudahan dalam mendefinisikan tipe data yang kita miliki:

```
a<-2
class(a)
```

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

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

Agar dapat bekerja secara efisien dalam menggunakan bahasa pemrograman R, penting untuk mempelajari terlebih dahulu tipe data dari variabel-variabel yang kita miliki sehingga akan mempermudah dalam penentuan proses analisis data yang dapat dilakukan terhadap variabelvariabel tersebut.

***Data Frames***

Cara paling umum yang dapat digunakan untuk menyimpan *dataset* dalam R adalah dalam tipe *data frame*. Secara konseptual, kita dapat menganggap *data frame* sebagai tabel yang terdiri dari baris yang memiliki nilai pengamatan dan berbagai variabel yang didefinisikan dalam bentuk kolom. Tipe data ini sangat umum digunakan untuk *dataset*, karena *data frame* dapat menggabungkan berbagai jenis tipe data dalam satu objek. Untuk memahami tipe *data frame*, silahkan mengakses contoh *dataset* pada `library(dslabs)`dan pilih *dataset* "*murders*" menggunakan fungsi `data`:

```
library(dslabs)
data(murders)
```

Untuk memastikan bahwa *dataset* tersebut tipenya adalah *data frame*, dapat digunakan perintah berikut:

```
class(murders)
```

```
## [1] "data.frame"
```

```
#> [1] "data.frame"
```

Untuk memeriksa lebih lanjut isi *dataset*, dapat pula digunakan fungsi `str` untuk mencari tahu lebih rinci mengenai struktur suatu objek:

```
str(murders)
```

```
## 'data.frame':    51 obs. of  5 variables:
##  $ state     : chr  "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ abb       : chr  "AL" "AK" "AZ" "AR" ...
##  $ region    : Factor w/ 4 levels "Northeast","South",..: 2 4 4 2 4 4 1 2 2 2 ...
##  $ population: num  4779736 710231 6392017 2915918 37253956 ...
##  $ total     : num  135 19 232 93 1257 ...
```

```
#> 'data.frame': 51 obs. of 5 variables:
#> $ state : chr "Alabama" "Alaska" "Arizona" "Arkansas" ...
#> $ abb : chr "AL" "AK" "AZ" "AR" ...
#> $ region : Factor w/ 4 levels "Northeast","South",..: 2 4 4 2 4 4 1 2 2
#> 2 ...
#> $ population: num 4779736 710231 6392017 2915918 37253956 ...
#> $ total : num 135 19 232 93 1257 ...
```

Dengan menggunakan fungsi `str`, dapat diketahui bahwa *dataset* "*murders*" terdiri dari 51 baris dan lima variabel: *state, abb, region, population, dan total.* Selanjutnya, untuk melihat contoh enam baris pertama pada dataset, dapat digunakan fungsi `head`:

```
head(murders)
```

```
##        state abb region population total
## 1    Alabama  AL  South    4779736   135
## 2     Alaska  AK   West     710231    19
## 3    Arizona  AZ   West    6392017   232
## 4   Arkansas  AR  South    2915918    93
## 5 California  CA   West   37253956  1257
## 6   Colorado  CO   West    5029196    65
```

```
#> state abb region population total
#> 1 Alabama AL South 4779736 135
#> 2 Alaska AK West 710231 19
#> 3 Arizona AZ West 6392017 232
#> 4 Arkansas AR South 2915918 93
#> 5 California CA West 37253956 1257
#> 6 Colorado CO West 5029196 65
```

Untuk analisis awal tiap variabel yang diwakili dalam bentuk kolom pada tipe *data frame*, dapat digunakan operator aksesor ($) dengan cara berikut:

```
murders$population
```

```
##  [1]  4779736   710231  6392017  2915918 37253956  5029196  3574097   897934
##  [9]   601723 19687653  9920000  1360301  1567582 12830632  6483802  3046355
```

```
## [17]  2853118  4339367  4533372  1328361  5773552  6547629  9883640  5303925
## [25]  2967297  5988927   989415  1826341  2700551  1316470  8791894  2059179
## [33] 19378102  9535483   672591 11536504  3751351  3831074 12702379  1052567
## [41]  4625364   814180  6346105 25145561  2763885   625741  8001024  6724540
## [49]  1852994  5686986   563626
```

```
#> [1] 4779736 710231 6392017 2915918 37253956 5029196 3574097
#> [8] 897934 601723 19687653 9920000 1360301 1567582 12830632
#> [15] 6483802 3046355 2853118 4339367 4533372 1328361 5773552
#> [22] 6547629 9883640 5303925 2967297 5988927 989415 1826341
#> [29] 2700551 1316470 8791894 2059179 19378102 9535483 672591
#> [36] 11536504 3751351 3831074 12702379 1052567 4625364 814180
#> [43] 6346105 25145561 2763885 625741 8001024 6724540 1852994
#> [50] 5686986 563626
```

Untuk mengetahui nama-nama dari lima variabel yang dapat dievaluasi menggunakan operator aksesor, sebelumnya, melalui fungsi `str`, telah kita ketahui bahwa variabel yang dimiliki *dataset* adalah: *state, abb, region, population, dan total.* Sebagai alternatif, terdapat pula fungsi `name`, yang dapat digunakan seperti contoh dibawah ini:

```
names(murders)
```

```
## [1] "state"      "abb"        "region"     "population" "total"
```

```
#> [1] "state" "abb" "region" "population" "total"
```

***Vector : numeric, character, dan logical***

Objek `murders$population` terdiri dari sekumpulan *numeric* atau data-data angka. Sehingga, kita dapat mendefinisikan bahwa tipe data `murders$population` berupa *vector.* Angka tunggal secara teknis dapat didefinisikan sebagai vektor dengan panjang 1, tetapi secara umum kita akan menggunakan *vector* sebagai istilah untuk merujuk ke objek yang terdiri dari beberapa entri.Untuk mengidentifikasi banyaknya entri dalam suatu vector dapat digunakan fungsi `length`seperti contoh berikut:

```
length(murders$population)
```

```
## [1] 51
```

```
#> [1] 51
```

*Vector* khusus ini bertipe *numeric* karena populasi terdiri dari data-data angka:

```
class(murders$population)
```

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

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

Secara matematis, nilai-nilai dalam `murders$population` adalah berupa *integer.* Namun, secara default, data angka akan diberikan tipe *numeric* meskipun sebenarnya data tersebut merupakan bilangan bulat. Misalnya, `class(1)`akan mengidentifikasi nilai 1 sebagai tipe *numeric.* Untuk mengubah tipe *numeric*

menjadi *integer*, dapat digunakan fungsi `as.integer()` atau dengan menambahkan L pada akhir data angka, contoh: 1L. Untuk melihat perbedaannya, silahkan gunakan `class(1L)`.

*Vector* juga dapat digunakan untuk menyimpan *string* dengan tipe *character*, Sebagai contoh: nama negara pada dataset "*murders*":

```
class(murders$state)
```

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

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

Jenis *vector* penting lainnya adalah *logical* yang nilainya berupa TRUE atau FALSE.

```
z <- 3 == 2
z
```

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

```
#> [1] FALSE
class(z)
```

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

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

***Factors***

Dalam *dataset* "*murders*", variabel *state* yang berisi data karakter bukan bertipe *vector*: *character*, namun, tipe datanya adalah *factor*:

```
class(murders$region)
```

```
## [1] "factor"
```

```
#> [1] "factor"
```

Faktor berguna untuk menyimpan data kategorikal. Dapat dilihat, bahwa hanya terdapat 4 wilayah pada variabel *state.* Untuk melihat jumlah kategori yang dimiliki oleh variabel dengan tipe data *factor* dapat digunakan fungsi `level`:

```
levels(murders$region)
```

```
## [1] "Northeast"     "South"         "North Central" "West"
```

```
#> [1] "Northeast" "South" "North Central" "West"
```

Pada *background process*, R menyimpan level sebagai bilangan bulat yang memiliki peta tersendiri untuk melacak arti label dari bilangan tersebut. Hal ini dimaksudkan untuk penghematan memori, terutama apabila karakter dari tiap level cukup panjang. Standarnya, level akan ditampilkan sesuai urutan abjad.

***Lists***

*Data frame* merupakan sekumpulan *list* yang memiliki kelas yang berbeda-beda. Sama halnya dengan *data frame*, analisis *list* dapat dilakukan dengan menggunakan operator aksesor ($) dan dua kurung siku ([[).

**Matriks**

Matriks merupakan tipe data yang mirip dengan *data frame* karena keduanya memiliki dua dimensi, yaitu: baris dan kolom. Namun, sama halnya dengan tipe data *vector* numerik, karakter dan logis, entri dalam matriks harus terdiri dari jenis *vector* yang sama. Dalam hal ini, *data frame* dapat dikatakan sebagai tipe data yang paling cocok untuk menyimpan data, karena kita dapat memiliki karakter, faktor, dan angka sekaligus dalam satu *data frame.* Namun matriks memiliki satu keunggulan yang tidak dimiliki oleh tipe *data frame*: pada matriks dapat dilakukan operasi aljabar. Untuk mendefinisikan matriks, dapat digunakan fungsi `matrix` dengan mendefinisikan pula argumen berupa jumlah baris dan kolom yang diinginkan.

```
mat <- matrix(1:12, 4, 3)
mat
```

```
##      [,1] [,2] [,3]
## [1,]    1    5    9
## [2,]    2    6   10
## [3,]    3    7   11
## [4,]    4    8   12
```

```
#> [,1] [,2] [,3]
#> [1,] 1 5 9
#> [2,] 2 6 10
#> [3,] 3 7 11
#> [4,] 4 8 12
```

Untuk mengakses entri tertentu dalam matriks, dapat digunakan tanda kurung siku ([). Sebagai contoh, kita akan menampilkan data pada baris kedua, kolom ketiga, menggunakan:

```
mat[2, 3]
```

```
## [1] 10
```

```
#> [1] 10
```