

# Latihan5\_123190138

rega agassi

10/28/2021

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

## Modul 5

1. Fungsi nchar dapat digunakan untuk menghitung jumlah karakter dari suatu vektor karakter. Buatlah satu baris kode yang akan menyimpan hasil komputasi pada variabel 'new\_names' dan berisi singkatan nama negara ketika jumlah karakternya lebih dari 8 karakter. .

```
new_names <- ifelse(nchar(murders$state) > 8, murders$abb, murders$state)
new_names
```

```
## [1] "Alabama" "Alaska" "Arizona" "Arkansas" "CA" "Colorado"
## [7] "CT" "Delaware" "DC" "Florida" "Georgia" "Hawaii"
## [13] "Idaho" "Illinois" "Indiana" "Iowa" "Kansas" "Kentucky"
## [19] "LA" "Maine" "Maryland" "MA" "Michigan" "MN"
## [25] "MS" "Missouri" "Montana" "Nebraska" "Nevada" "NH"
## [31] "NJ" "NM" "New York" "NC" "ND" "Ohio"
## [37] "Oklahoma" "Oregon" "PA" "RI" "SC" "SD"
## [43] "TN" "Texas" "Utah" "Vermont" "Virginia" "WA"
## [49] "WV" "WI" "Wyoming"
```

2. Buat fungsi sum\_n yang dapat digunakan untuk menghitung jumlah bilangan bulat dari 1 hingga n. Gunakan pula fungsi ini untuk menentukan jumlah bilangan bulat dari 1 hingga 5.000. .

```
sum_n <- function(n){
  x<- 1:n
  sum(x)
}
n<-5000
sum_n(n)
```

```
## [1] 12502500
```

3. Buat fungsi compute\_s\_n yang dapat digunakan untuk menghitung jumlah  $S_n = 1^2 + 2^2 + 3^2 + \dots + n^2$ . Tampilkan hasil penjumlahan ketika  $n = 10$ .

```
compute_s_n <- function(n){
  x <- 1:n
  sum(x*x)
}
n<-10
compute_s_n(n)
```

```
## [1] 385
```

4. Buat vektor numerik kosong dengan nama: s\_n dengan ukuran:25 menggunakan s\_n <- vector ("numeric", 25). Simpan di hasil komputasi S1, S2,. . . S25 menggunakan FOR-LOOP. .

```
len <- 25
s_n <- vector("numeric", length = len)
for(n in 1:len){
  s_n[n] <- compute_s_n(n)
}
s_n
```

```
## [1] 1 5 14 30 55 91 140 204 285 385 506 650 819 1015 1240
## [16] 1496 1785 2109 2470 2870 3311 3795 4324 4900 5525
```

5. Ulangi langkah pada soal no. 4 dan gunakan fungsi sapply.

```
n <- 1:25
s_n <- sapply(n, compute_s_n)
s_n
```

```
## [1] 1 5 14 30 55 91 140 204 285 385 506 650 819 1015 1240
## [16] 1496 1785 2109 2470 2870 3311 3795 4324 4900 5525
```