

1-Pengenalan R

Ashari Ramadhan

11/19/2020

1. Pengenalan

Pengenalan R

R merupakan bahasa yang digunakan dalam komputasi statistik.



Figure 1: Logo R

R bisa digunakan untuk membuat model linier dan nonlinier, hipotesis test, visualisasi, time series, klasifikasi, web-apps, pemetaan dan lain-lain.

Kenapa harus pakai R? 1. Gratis / Open Source 2. Packages / Library yang banyak 3. Digunakan oleh industri 4. Komunitas Besar

Case sensitif

Pada bahasa R, huruf kapital dan non kapital dianggap berbeda. Contoh angka berbeda dengan Angka berbeda pula dengan ANGKA

```
"angka" == "Angka"
```

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

```
"angka" == "angka"
```

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

Hasil FALSE menandakan objek berbeda, sedangkan TRUE menandakan objek sama

Komentar

Komentar digunakan untuk memberikan penjelasan pada program. Komentar tidak akan mempengaruhi jalannya program. Pada bahasa R semua text yang berada di di belakang `#` akan dianggap komentar dan tidak akan dieksekusi

```
# ini adalah komentar  
# 1 + 1
```

Kode diatas tidak menghasilkan output.

```
# ini adalah komentar  
1 + 1
```

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

baris `1 + 1` dieksekusi dan menghasilkan output 2

help

Setiap fungsi di R memiliki dokumentasi yang berisi cara penulisan, parameter dan penjelasan-penjelasan lainnya. misal `mean` adalah fungsi untuk mencari nilai rata-rata. Jika ingin menampilkan dokumentasi `mean` ketikkan

```
help("mean")
```

atau

```
?mean
```

Say Hello World

```
print('Hello World')
```

```
## [1] "Hello World"
```

2. Operator pada R

Operator aritmatika

- tanda `+` adalah penjumlahan
- tanda `-` adalah pengurangan
- tanda `/` adalah pembagian
- tanda `*` perkalian
- tanda `^` pangkat
- tanda `%%` modulus, sisa bagi

```
print('1 + 1 sama dengan')
```

```
## [1] "1 + 1 sama dengan"
```

```
1 + 1
```

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

```
print('2 - 3 sama dengan')
```

```
## [1] "2 - 3 sama dengan"
```

```
2 - 3
```

```
## [1] -1
```

```
print('8/3 sama dengan')
```

```
## [1] "8/3 sama dengan"
```

```
8/3
```

```
## [1] 2.666667
```

```
print('9 * 0.5 sama dengan')
```

```
## [1] "9 * 0.5 sama dengan"
```

```
9*0.5
```

```
## [1] 4.5
```

```
print('6 ^ 9 sama dengan')
```

```
## [1] "6 ^ 9 sama dengan"
```

```
6^9
```

```
## [1] 10077696
```

```
print('9 %% 3')
```

```
## [1] "9 %% 3"
```

```
9%%3
```

```
## [1] 0
```

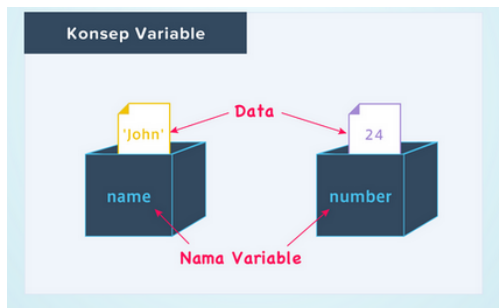
tambahan operator

```
numbers = c(1:7)
numbers
```

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

3. Variabel

Variabel tempat untuk menyimpan sebuah nilai



Membuat variabel kita memerlukan operator assignment

```
number <- 24
print(number)
```

```
## [1] 24
```

```
print(number + 2)
```

```
## [1] 26
```

```
name <- 'John'
print(paste('Nama saya', name, 'umur saya', number - 2, 'tahun'))
```

```
## [1] "Nama saya John umur saya 22 tahun"
```

Operator assignment

```
name1 <- 'John' # tapi ini yang disarankan untuk di pake, RStyle
name2 = 'John'
name3 <<- 'John'
'John' -> name4
```

Penamanaan Variabel

- Diberi nama yang jelas, sesuai dengan isinya.

Sebagai contoh akan lebih mudah dibaca jika menulis:

```
umurku = 22
```

dibanding

```
x = 22
```

Meskipun maksud x adalah umurku

Penamanaan Variabel

- Tidak diawali angka atau __

```
_x = 3
```

Error: unexpected input in "_"

Penamaan variabel

- jangan menggunakan tanda operasi untuk penghubungan atau tanda spasi. Sebagai pengganti gunakan tanda __ sebagai penghubungan
- disarankan menggunakan huruf kecil, menggunakan Snake case (stylized as snake_case)

```
umur-ku = 20
```

Error in umur - ku = 20 : object 'umur' not found

Memasukkan banyak nilai pada variabel

Tentu tidak efektif jika menulis kode secara berulang seperti ini

```
food1 <- 'banana'
food2 <- 'orange'
food3 <- 'grape'
```

```
print(food1)
```

```
## [1] "banana"
```

```
print(food2)
```

```
## [1] "orange"
```

```
print(food3)
```

```
## [1] "grape"
```

Lebih baik di tulis seperti ini

```
# lebih baik di tulis seperti ini
foods = c('banana', 'orange', 'grape')
print(foods)
```

```
## [1] "banana" "orange" "grape"
```

4. Percabangan

Percabangan adalah fitur dari bahasa pemrograman yang melakukan perhitungan atau tindakan yang berbeda tergantung pada apakah kondisi boolean yang ditentukan pemrogram mengevaluasi benar atau salah.

Kondisi boolean: + TRUE, kondisi benar + FALSE, kondisi salah

Operator	Description
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
!=	not equal to
!x	Not x
x y	x OR y
x & y	x AND y
isTRUE(x)	test if X is TRUE

Figure 2: Operator Logika

```
x = 4
y = 9
x < y
```

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

```
y <= x
```

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

```
x > y
```

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

```
y >= x
```

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

```
x == y
```

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

```
y != x
```

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

```
## Percabangan
```

Percabangan adalah cara yang digunakan untuk mengambil keputusan apabila di dalam program dihadapkan pada kondisi tertentu. Contoh kita ingin mencetak 'Anda lulus ujian' jika nilai yang di peroleh >= 70

```
if ( __kondisi__ ){  
    __statment__  
}
```

Logika nya statement akan di eksekusi jika berniali TRUE (benar)

IF

```
nilai <- 70  
if(nilai > 70){  
    print('Anda lulus ujian')  
}
```

if else

```
nilai <- 70  
if(nilai > 70){  
    print('Anda lulus ujian')  
} else{  
    print('anda tidak lulus ujian')  
}
```

```
## [1] "anda tidak lulus ujian"
```

if elseif else

```
nilai <- 70
if(nilai > 70){
  print('anda lulus ujian')
} else if( nilai == 70){
  print('anda tidak lulus ujian, tapi boleh remedian')
} else{
  print('anda tidak lulus ujian')
}
```

```
## [1] "anda tidak lulus ujian, tapi boleh remedian"
```

Latihan

Buat program percabangan dengan kondisi + jika nilai > 80 cetak “A” + jika 70 < nilai <= 80 cetak “B” + jika nilai 50 < nilai <=70 cetak “C” + Selain itu cetak “D”

5. Perulangan

6. Fungsi

7.Packages