

BAB 10

STRUKTUR PEMROGRAMAN R

Norman Matloff (2009:67) dalam bukunya “*The Art of R Programming*” menyatakan sebagai berikut.

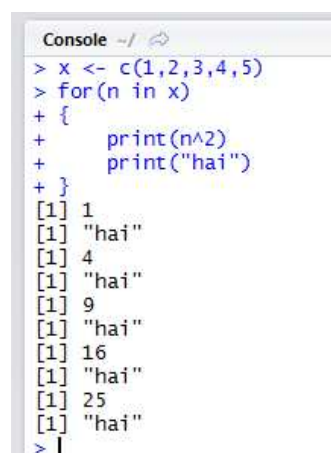
“R is a full programming language, similar to scripting languages such as Perl and Python. One can define functions, use constructs such as loops and conditionals, etc.

R is also block-structured, in a manner similar to those of the above languages, as well as C. Blocks are delineated by braces, though they are optional if the block consists of just a single statement.”

Jadi, pernyataan di atas memberikan informasi bahwa R merupakan bahasa pemrograman, sama seperti bahasa pemrograman *Perl* dan *Python*. Dalam R, dapat mendefinisikan suatu fungsi, menggunakan konstruk-konstruk seperti pengulangan (*loops*) dan kondisi (*conditionals*), dan sebagainya. R juga merupakan *block-structured*, mirip dengan bahasa pemrograman *Perl* dan *Python*, dan juga *C*. Suatu blok diawali dengan buka kurawal (*braces*) dan diakhiri tutup kurawal, meskipun hal tersebut bersifat opsional jika dalam blok tersebut hanya terdiri dari satu pernyataan.

10.1 Pengulangan (*Loop*) dengan *for* dan Kondisi (*Condition*) dengan *if*

Gambar 10.1 diberikan ilustrasi mengenai pengulangan dalam R.



```
Console -/ ↻
> x <- c(1,2,3,4,5)
> for(n in x)
+ {
+   print(n^2)
+   print("hai")
+ }
[1] 1
[1] "hai"
[1] 4
[1] "hai"
[1] 9
[1] "hai"
[1] 16
[1] "hai"
[1] 25
[1] "hai"
> |
```

Gambar 10.1

Berdasarkan Gambar 10.1, kode R

```
for (n in x)
{
print(n^2)
print("hai")
}
```

merupakan contoh dari **pengulangan** (*loop*). Diketahui vektor **x** menyimpan bilangan 1, 2, 3, 4, dan 5. Bilangan-bilangan tersebut kemudian dikuadratkan, sehingga menghasilkan 1, 4, 9, 16, dan 25. Kode R

print(n^2)

bertujuan untuk mengkuadratkan bilangan-bilangan dalam *n*.

- ⇒ Pada iterasi pertama, $n = 1$, maka $n^2 = 1$.
- ⇒ Pada iterasi kedua, $n = 2$, maka $n^2 = 4$.
- ⇒ Pada iterasi ketiga, $n = 3$, maka $n^2 = 9$.
- ⇒ Pada iterasi keempat, $n = 4$, maka $n^2 = 16$.
- ⇒ Pada iterasi kelima, $n = 5$, maka $n^2 = 25$.

Perhatikan juga bahwa tulisan “hai” tercetak sebanyak 5 kali (hal ini menandakan terdapat 5 kali pengulangan).

Informasi:

Fungsi **print()** digunakan untuk mencetak.

10.2 Contoh Ke-2: Pengulangan (Loop) dengan for dan Kondisi (Condition) dengan if

Gambar 10.2 diberikan contoh lagi mengenai pengulangan dalam R. Berdasarkan Gambar 10.2, kode R

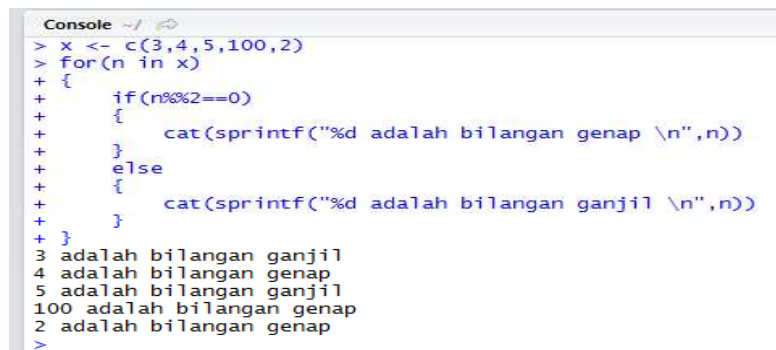
```
x <- c(3,4,5,100,2)
for(n in x)
{
  if(n%%2==0)
  {
    cat(sprintf("%d adalah bilangan genap \n",n))
  }
  else
  {
    cat(sprintf("%d adalah bilangan ganjil \n",n))
  }
}
```

merupakan contoh dari pengulangan yang juga melibatkan suatu kondisi.

- ⇒ Kondisi pertama, jika suatu bilangan dibagi 2, memiliki nilai sisa sama dengan 0, maka bilangan tersebut adalah bilangan genap, yakni dinyatakan dengan

if(n%%2==0)

- ⇒ Kondisi kedua, jika suatu bilangan dibagi 2, memiliki nilai sisa tidak sama 0, maka bilangan tersebut adalah bilangan ganjil.



```
Console ~/ / ↵
> x <- c(3,4,5,100,2)
> for(n in x)
+ {
+   if(n%%2==0)
+   {
+     cat(sprintf("%d adalah bilangan genap \n",n))
+   }
+   else
+   {
+     cat(sprintf("%d adalah bilangan ganjil \n",n))
+   }
+ }
3 adalah bilangan ganjil
4 adalah bilangan genap
5 adalah bilangan ganjil
100 adalah bilangan genap
2 adalah bilangan genap
>
```

Gambar 10.2

Diketahui vektor **x** menyimpan bilangan 3, 4, 5, 100, dan 2.

- ⇒ Pada iterasi pertama, $n = 3$. Nilai 3 dibagi 2 memiliki nilai sisa 1 (tidak sama dengan nol), maka 3 adalah bilangan ganjil.
- ⇒ Pada iterasi kedua, $n = 4$. Nilai 4 dibagi 2 memiliki nilai sisa 0, maka 4 adalah bilangan genap.
- ⇒ Pada iterasi ketiga, $n = 5$. Nilai 5 dibagi 2 memiliki nilai sisa 1 (tidak sama dengan nol), maka 5 adalah bilangan ganjil.
- ⇒ Pada iterasi keempat, $n = 100$. Nilai 100 dibagi 2 memiliki nilai sisa 0, maka 100 adalah bilangan genap.
- ⇒ Pada iterasi kelima, $n = 2$. Nilai 2 dibagi 2 memiliki nilai sisa 0, maka 2 adalah bilangan genap.

Informasi:

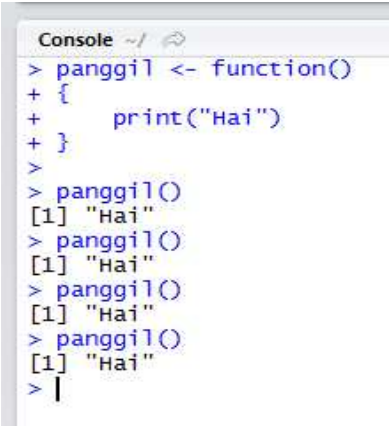
- ⇒ Fungsi **cat(sprintf())** digunakan untuk mencetak.
- ⇒ Perhatikan bahwa **%d** digunakan untuk menyatakan bilangan bulat.

BAB 11

FUNGSI

11.1 Membuat Fungsi Sederhana dalam R dan Memanggil Fungsi

Berikut diberikan contoh kode R terkait pembuatan fungsi.



```
Console ~/
> panggil <- function()
+ {
+   print("Hai")
+ }
>
> panggil()
[1] "Hai"
> panggil()
[1] "Hai"
> panggil()
[1] "Hai"
> panggil()
[1] "Hai"
> |
```

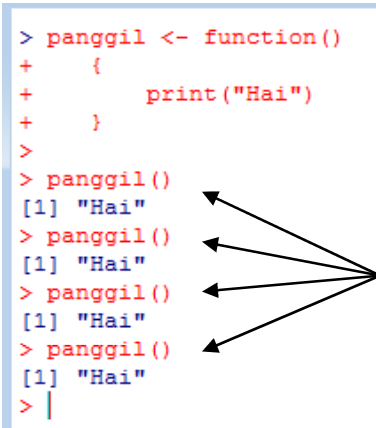
Membuat suatu fungsi bernama **panggil**, untuk mencetak tulisan **Hai**.

Gambar 11.1

Pada kode R Gambar 11.1, dibentuk suatu fungsi bernama **panggil**.

Suatu fungsi memiliki nama. Fungsi yang dibentuk pada Gambar 11.1 bernama **panggil**.

Berdasarkan kode R Gambar 11.1, fungsi bernama **panggil** dipanggil sebanyak 4 kali (Perhatikan Gambar 11.2).



```
> panggil <- function()
+ {
+   print("Hai")
+ }
>
> panggil()
[1] "Hai"
> panggil()
[1] "Hai"
> panggil()
[1] "Hai"
> panggil()
[1] "Hai"
> |
```

Fungsi bernama **panggil**, dipanggil sebanyak 4 kali. Setiap pemanggilan fungsi tersebut, akan mencetak tulisan **Hai**. Karena fungsi bernama **panggil** dipanggil sebanyak 4 kali, maka fungsi bernama **panggil** mencetak tulisan **Hai** sebanyak 4 kali.

Gambar 11.2

Fungsi bernama **panggil**, dipanggil sebanyak 4 kali. Setiap pemanggilan fungsi tersebut, akan mencetak tulisan **Hai**. Karena fungsi bernama **panggil** dipanggil sebanyak 4 kali, maka fungsi bernama **panggil** mencetak tulisan **Hai** sebanyak 4 kali. Dimulai dari buka kurawal "{", sampai dengan tutup kurawal "}" disebut **tubuh fungsi** (*body of function*) (Gambar 11.3).

```
+ {  
+ print("Hai")  
+ }  
~
```

Gambar 11.3 Tubuh Fungsi dari Fungsi Panggil

Tubuh fungsi dimulai dari buka kurawal "{", sampai dengan tutup kurawal "}".

Setelah suatu fungsi dibentuk/dibuat, maka suatu fungsi dapat dipanggil. Berikut sintaks untuk memanggil suatu fungsi yang telah dibentuk.

nama_fungsi()

Jadi untuk memanggil suatu fungsi yang telah dibentuk/dibuat, maka perlu disebutkan nama fungsi, kemudian ditambah buka kurung "(" dan tutup kurung ")".

Jadi untuk memanggil suatu fungsi yang telah dibentuk/dibuat, **maka perlu disebutkan nama fungsi**, kemudian ditambah buka kurung "(" dan tutup kurung ")".

11.2 Contoh Fungsi yang Melibatkan Dua Argumen

Berikut diberikan contoh kode R yang melibatkan dua argumen dalam suatu fungsi (Gambar 11.4).

```

Console ~/
> hitung <- function(x,y)
+ {
+   cat(sprintf("Penjumlahan %d + %d = %d\n",x,y,x+y))
+   cat(sprintf("Pengurangan %d - %d = %d\n",x,y,x-y))
+   cat(sprintf("Pembagian %d / %d = %f\n",x,y,x/y))
+   cat(sprintf("Perkalian %d * %d = %d\n",x,y,x*y))
+ }
>
> hitung(5,2)
Penjumlahan 5 + 2 = 7
Pengurangan 5 - 2 = 3
Pembagian 5 / 2 = 2.500000
Perkalian 5 * 2 = 10
> hitung(10,5)
Penjumlahan 10 + 5 = 15
Pengurangan 10 - 5 = 5
Pembagian 10 / 5 = 2.000000
Perkalian 10 * 5 = 50
> |

```

Fungsi bernama **hitung** dipanggil sebanyak 2 kali, yakni **hitung(5,2)** untuk pemanggilan pertama, dan **hitung(10,5)** untuk pemanggilan yang kedua.

Gambar 11.4

Berdasarkan Gambar 11.4, dibentuk suatu fungsi bernama **hitung** yang memiliki argumen sebanyak 2, yakni **x** dan **y**.

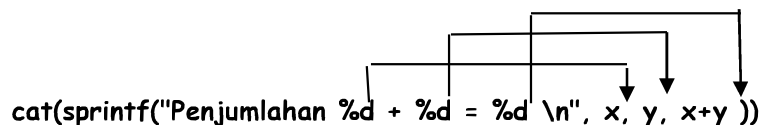
Suatu fungsi **memiliki nama**. Fungsi yang dibentuk pada Gambar 11.4 bernama **hitung**, memiliki argumen sebanyak 2, yakni **x** dan **y**.

Berdasarkan kode R pada Gambar 11.4, fungsi bernama **hitung** dipanggil sebanyak 2 kali (Perhatikan Gambar 11.4).

Pada fungsi bernama **hitung**, kode R

```
cat(sprintf("Penjumlahan %d + %d = %d \n", x, y, x+y ))
```

bertujuan untuk menjumlahkan dua buah bilangan. Perhatikan bahwa dalam sintaks tersebut terdapat “%d” yang berarti untuk menyatakan bilangan bulat.



Pada fungsi bernama **hitung**, kode R

```
cat(sprintf("Pengurangan %d - %d = %d \n", x, y, x-y ))
```

bertujuan untuk mengurangkan dua buah bilangan. Perhatikan bahwa dalam sintaks tersebut terdapat “%d” yang berarti untuk menyatakan bilangan bulat.

```
cat(sprintf("Penjumlahan %d - %d = %d \n", x, y, x-y))
```

Pada fungsi bernama hitung, kode R

```
cat(sprintf("Pembagian %d / %d = %f \n", x, y, x/y))
```

bertujuan untuk pembagian dua buah bilangan. Perhatikan bahwa dalam sintaks tersebut terdapat “%d” yang berarti untuk menyatakan bilangan bulat, sementara “%f” untuk menyatakan bilangan pecahan.

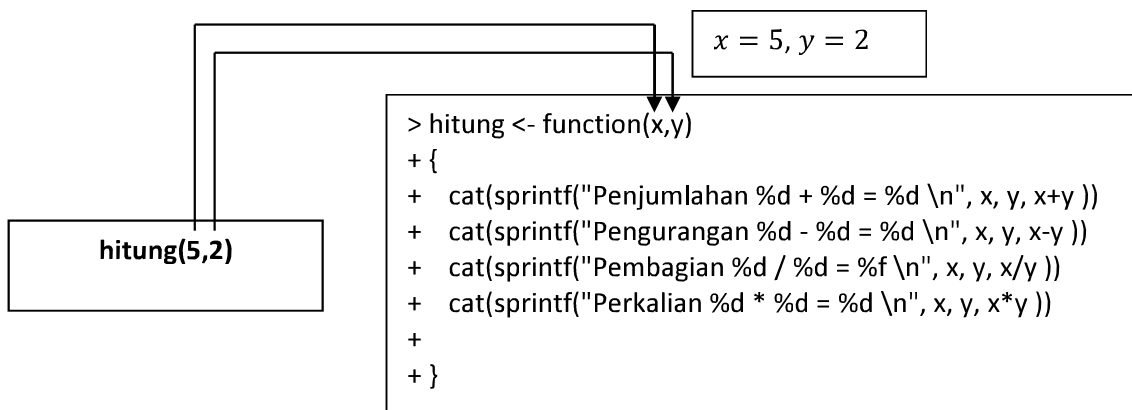
```
cat(sprintf("Penjumlahan %d / %d = %f \n", x, y, x/y))
```

Perhatikan bahwa %f digunakan karena hasil pembagian dari dua buah bilangan, yakni x/y dapat berupa bilangan pecahan.


Kode R

```
hitung(5,2)
```

berarti memanggil fungsi bernama **hitung**, kemudian nilai x diisi dengan 5, sementara nilai y diisi dengan 2.



$5 + 2 = 7$



```
cat(sprintf("Penjumlahan %d + %d = %d \n", x, y, x+y))
```

$5 - 2 = 3$

```
cat(sprintf("Pengurangan %d - %d = %d \n", x, y, x-y))
```

$5 / 2 = 2.5$

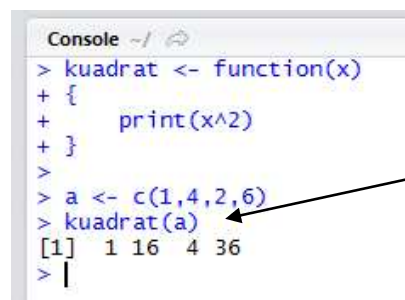
```
cat(sprintf("Pembagian %d / %d = %f \n", x, y, x/y))
```

$5 * 2 = 10$

```
cat(sprintf("Perkalian %d * %d = %d \n", x, y, x*y))
```

11.3 Contoh Fungsi yang Melibatkan Argumen Vektor

Berikut diberikan contoh kode R yang melibatkan argumen vektor dalam suatu fungsi (Gambar 11.5).



```
Console -1
> kuadrat <- function(x)
+ {
+   print(x^2)
+ }
>
> a <- c(1,4,2,6)
> kuadrat(a)
[1] 1 16 4 36
> |
```

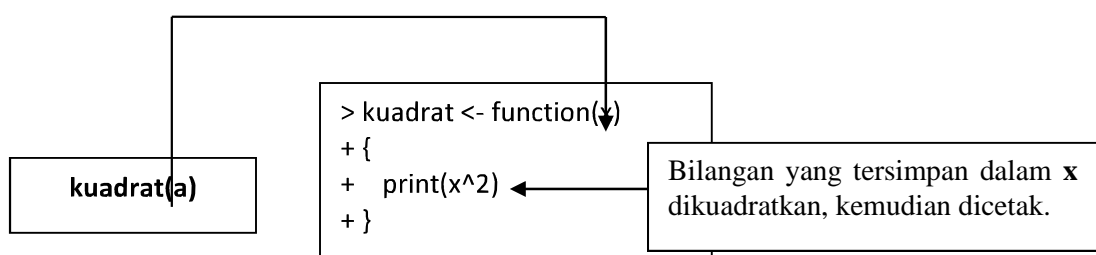
Fungsi bernama **kuadrat** dipanggil sebanyak 1 kali, yakni **hitung(a)**.

Gambar 11.5

Pada kode R Gambar 11.5, dibentuk suatu fungsi bernama **kuadrat** yang memiliki argumen sebanyak 1, yakni **x**.

Suatu fungsi **memiliki nama**. Fungsi yang dibentuk pada Gambar 11.5 bernama **kuadrat**, memiliki argumen sebanyak 1, yakni **x**.

Berdasarkan kode R Gambar 11.5, fungsi bernama **kuadrat** dipanggil sebanyak 1 kali.



11.4 Contoh Fungsi yang Melibatkan Pengembalian Nilai (*Return Value*)

Berikut diberikan contoh kode R yang melibatkan fungsi dengan pengembalian nilai (Gambar 11.6).

```
Console -1 ↻
> kuadrat <- function(x)
+ {
+   return(x^2)
+ }
>
> a <- c(1,4,2,6)
> a
[1] 1 4 2 6
> a <- kuadrat(a)
> a
[1] 1 16 4 36
> |
```

Gambar 11.6

Pada kode R Gambar 11.6, dibentuk suatu fungsi bernama **kuadrat** yang memiliki argumen sebanyak 1, yakni **x**.

Suatu fungsi memiliki nama. Fungsi yang dibentuk pada Gambar 11.6 bernama **kuadrat**, memiliki argumen sebanyak 1, yakni **x**.

Berdasarkan kode R Gambar 11.6, fungsi bernama **kuadrat** dipanggil sebanyak 1 kali.

