
7 PENGULANGAN

Komputer memiliki kemampuan untuk mengerjakan suatu instruksi (aksi) secara berulang-ulang dengan performansi yang sama. Kemampuan tersebut menjadi salah satu keunggulan komputer dibandingkan manusia karena manusia biasanya tidak menyukai tugas-tugas monoton yang dikerjakan secara berulang-ulang (karena lelah atau bosan).

Notasi pengulangan adalah salah satu notasi dasar dalam penulisan algoritma selain pemilihan. Terdapat beberapa macam struktur pengulangan pada algoritma dan beberapa diantaranya yang paling banyak digunakan antara lain struktur FOR, WHILE-DO, dan REPEAT-UNTIL.. Masing-masing struktur digunakan pada jenis permasalahan yang berbeda meskipun untuk beberapa kasus sebuah struktur pengulangan dapat diganti dengan struktur pengulangan yang lain.

1. Struktur FOR

Struktur ini digunakan bila sudah diketahui berapa kali akan mengulang satu atau beberapa aksi dalam badan pengulangan.

Bentuk Umum :

```
for variabel ← nilai_awal to nilai_akhir do  
aksi1  
aksi2  
...  
aksin  
endfor
```

Catatan :

- variabel adalah nama variabel kontrol bertipe karakter atau integer yang berfungsi sebagai pencacah pengulangan
- aksi1, aksi2, ..., aksin merupakan satu atau lebih instruksi yang dikerjakan secara berulang-ulang
- nilai_awal dan nilai_akhir bisa berupa konstanta atau ekspresi
- nilai_awal harus lebih kecil atau sama dengan nilai_akhir
- tipe data variabel harus sama dengan tipe data nilai_awal dan nilai_akhir
- Pengulangan akan dilakukan sebanyak nilai_akhir - nilai_awal + 1

Contoh : Membuat sebuah algoritma untuk menampilkan bilangan dari 1 sampai 10

```
Algoritma Show_Numeric  
{ menampilkan bilangan integer dari 1 sampai 10}  
Kamus  
i : integer  
Deskripsi  
for i ← 1 to 10 do  
write(i)  
endfor
```

For juga dapat digunakan pada pengulangan yang mencacah dari bilangan lebih besar ke bilangan yang lebih kecil

Bentuk Umum :

```
for variabel ← nilai_awal downto nilai_akhir do  
aksi1  
aksi2  
...  
aksin  
endfor
```

Catatan :

- variabel adalah nama variabel kontrol bertipe karakter atau integer yang berfungsi sebagai pencacah pengulangan
- aksi1, aksi2, ..., aksin merupakan satu atau lebih instruksi yang dikerjakan secara berulang-ulang
- nilai_awal dan nilai_akhir bisa berupa konstanta atau ekspresi
- nilai_awal harus lebih besar atau sama dengan nilai_akhir
- tipe data variabel harus sama dengan tipe data nilai_awal dan nilai_akhir
- Pengulangan akan dilakukan sebanyak nilai_awal - nilai_akhir + 1

Contoh : Membuat sebuah algoritma untuk menampilkan bilangan dari N sampai 1 dimana N diinputkan dari user

```
Algoritma Show_Numeric2  
{ menampilkan bilangan integer dari N sampai 1 }
```

Kamus

```
i, N : integer
```

Deskripsi

```
read(N)  
for i ← N downto 1 do  
  write(i)  
endfor
```

2. Struktur WHILE-DO (Pengulangan berdasarkan kondisi ulang)

Struktur pengulangan ini biasanya digunakan pada kasus yang belum pasti berapa kali aksi/instruksinya akan diulang. Struktur While mirip struktur IF yang melakukan pemeriksaan ekspresi boolean sebelum satu atau lebih aksi dikerjakan.

Bentuk Umum :

```
while (kondisi) do  
aksi1  
aksi2  
...  
aksin  
endwhile
```

Catatan :

- Kondisi adalah kondisi pengulangan berupa ekspresi boolean yang dapat menghasilkan nilai True atau False
- Aksi pada badan pengulangan dilaksanakan selama (kondisi) menghasilkan nilai True
- Pengulangan berhenti jika (kondisi) menghasilkan nilai False

Contoh : Membuat sebuah algoritma untuk menampilkan bilangan dari 1 sampai 10

```
Algoritma Show_Numeric3
{ menampilkan bilangan integer dari 1 sampai 10}
Kamus
  i : integer
Deskripsi
  i ← 1
  while i ≤ 10 do
    write(i)
    i ← i + 1
  endwhile
```

Badan pengulangan (aksi) pada struktur While-Do mungkin tidak akan pernah dilakukan karena sebelum aksi pertama dieksekusi, dilakukan test terhadap kondisi pengulangan. Pengulangan ini berpotensi untuk menimbulkan aksi “kosong” (tidak pernah melakukan apa-apa) jika pada test pertama kondisi menghasilkan nilai False

Contoh :

```
Kamus
  f : boolean
Deskripsi
  f ← true
  while (not f) do
    f ← not f
  enwhile
```

3. Struktur REPEAT-UNTIL (Pengulangan berdasarkan kondisi berhenti)

Struktur ini hampir sama dengan struktur While dan biasanya digunakan bila jumlah pengulangan belum dapat ditentukan saat program ditulis

Bentuk Umum :

```
Repeat
  aksi1
  aksi2
  ...
  aksin
Until (kondisi berhenti)
```

Catatan :

- Kondisi berhenti berupa ekspresi boolean yang dapat menghasilkan nilai True atau False
- Aksi pada badan pengulangan akan dikerjakan sampai kondisi berhenti bernilai True
- Badan pengulangan (aksi) pada struktur ini minimal dikerjakan satu kali karena pada waktu eksekusi pengulangan yang pertama tidak dilakukan test terhadap kondisi berhenti. Test terhadap kondisi berhenti dilakukan setelah aksi dikerjakan

Contoh : Membuat sebuah algoritma untuk menampilkan bilangan dari 1 sampai 10

```
Algoritma Show_Numeric4
{ menampilkan bilangan integer dari 1 sampai 10}
Kamus
  i : integer
Deskripsi
  i ← 1
  repeat
    write(i)
    i ← i + 1
  until i > 10
```

Perbedaan struktur Repeat-Until dan While-Do terletak pada pengecekan kondisi. Jika pada struktur While, kondisi dicek pada awal badan pengulangan, sedangkan pada struktur Repeat kondisi dicek pada akhir badan pengulangan.

Perbedaan yang lain, bila struktur While mengulang pernyataan selama kondisi masih terpenuhi (kondisi = True), struktur Repeat mengulang pernyataan selama kondisi belum terpenuhi (kondisi = False)

STUDI KASUS

1. Membuat algoritma untuk menampilkan semua bilangan faktor dari n dimana n diinputkan oleh user

```
Algoritma Faktor
Kamus
  n, i : integer
Deskripsi
  read(n)
  for i ← 1 to n do
    if (n mod i = 0) then
      write(i)
    endif
  endfor
```

2. Buat algoritma untuk membaca dan menghitung nilai mahasiswa kemudian menghitung nilai rata-rata dari nilai mahasiswa tersebut. Proses pembacaan dan perhitungan dilakukan sampai user tidak ingin menginputkan lagi

```
Algoritma Rerata_Nilai_Mahasiswa
Kamus
  n : integer
  jawab : char
  nilai, jumlah, rata : real
Deskripsi
  jumlah ← 0
  n ← 0
  repeat
    read(nilai)
    jumlah ← jumlah + nilai
    n ← n + 1
    write('Apakah anda ingin input data lagi (y/t) ? ')
    read(jawab)
  until (jawab = 'T') or (jawab = 't')
  rata ← jumlah / n
  write(rata)
```