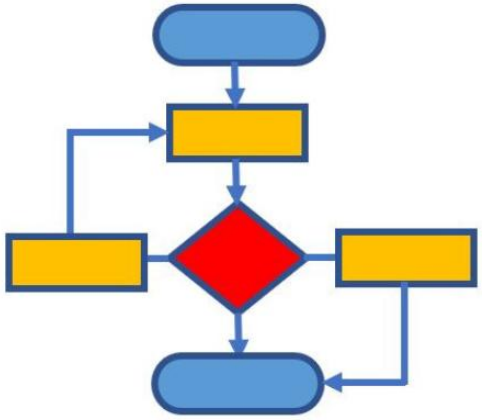
Topik 4 – Pernyataan

Algoritma dan Pemrograman



SCMA601004

Tim Dosen Algoritma dan Pemrograman

Departemen Matematika FMIPA Universitas Indonesia

Materi Pembahasan

- Pendahuluan Pernyataan
- *Assignment*
- *Branching*
- *Looping*
- Implementasi

Pendahuluan

Sebuah algoritma merupakan deskripsi langkah-langkah pelaksanaan suatu proses. Setiap langkah di dalam algoritma dinyatakan dalam sebuah **pernyataan** (*statement*) atau istilah lainnya **instruksi**. Sebuah pernyataan berisi **aksi** (*action*) yang dilakukan. Jika semua pernyataan dieksekusi oleh pemroses, maka aksi yang bersesuaian dengan pernyataan itu dikerjakan.

Di dalam algoritma terdapat beberapa jenis pernyataan, seperti pernyataan ekspresi, pernyataan pemilihan, pernyataan pengulangan, pernyataan prosedur, pernyataan gabungan, dan sebagainya.

Contoh:

Misalkan di dalam algoritma ada pernyataan berikut:

```
Cetak "Hello, World!"
```

Maka, pernyataan tersebut menggambarkan aksi mencetak pesan "Hello, World!"

Sumber: *Algoritma dan Pemrograman dalam Bahasa Pascal dan C,* Renaldi Munir.

Pendahuluan

Contoh:

Misalkan di dalam algoritma ada pernyataan berikut:

```
Kalikan x dengan 3
```

Maka, pernyataan tersebut menggambarkan aksi mengalikan `x` dengan 3 dan hasil perkalian disimpan di dalam peubah `x` lagi.

Contoh:

Misalkan di dalam algoritma ada pernyataan berikut:

```
Jika hasil = 100, maka cetak "Selamat, Anda lulus!"
```

Maka, pernyataan tersebut terdiri dari dua aksi, yaitu membandingkan nilai variabel `hasil` dengan 100 dan mencetak pesan "Selamat, Anda lulus!" jika perbandingan bernilai benar.

Sumber: *Algoritma dan Pemrograman dalam Bahasa Pascal dan C,* Renaldi Munir.

Pernyataan

Secara umum pada pemrograman, pernyataan dapat dibedakan menjadi:

- ***Assignment***

  Untuk “*assign*" variabel berarti mengasosiasikan secara simbolis informasi tertentu dengan nama. Operasi apa pun yang diterapkan ke "nama" (atau variabel) ini harus benar untuk setiap kemungkinan nilai.

- ***Branching***

  Ketika "Algoritma" membuat pilihan untuk melakukan salah satu dari dua (atau lebih banyak hal) ini disebut *branching* (percabangan). Yang paling umum pemrograman "pernyataan" yang digunakan untuk bercabang adalah pernyataan "IF“.

- ***Looping***

  Suatu loop digunakan untuk mengulang blok kode tertentu. Ada dua jenis pengulangan, “FOR” *loop* dan “WHILE” *loop*.

Variable Assignment

Simbol “=” adalah operator *assignment* yang tidak digunakan untuk persamaan (yang menggunakan double tanda sama dengan).

Walaupun operator assignment seperti tanda sama dengan di Matematika, tetapi dalam hal ini berbeda. Operator sama dengan pada pemrograman adalah “==”.

Syntax :	**nama_variabel = ekspresi**

Ekspresi dapat berupa bilangan : nama_variabel = 3
Ekspresi matematika : nama_variabel = 9 – 4/3 + 8
Function call : nama_variabel = sin (5)

Untuk mengevaluasi pernyataan *Assignment*:

1. Evaluasi “sisi kanan” ekspresi (sisi kanan tanda sama dengan)
2. Setelah nilainya diketahui, masukkan nilai tersebut dalam variabel (nama_variabel).

Sumber: *http://www.cs.utah.edu/~germain/PPS/Topics/*

Contoh

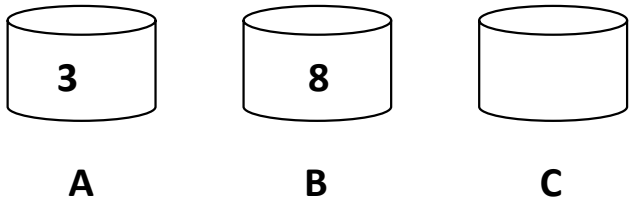
Algoritma	Program (Python)

Input A, B
 	
Masukkan nilai A ke dalam C
Masukkan nilai B ke dalam A
Masukkan nilai C ke dalam B
 	
Cetak A, B
	`A = 3`
	`B = 8`
	
	`C = A`
	`A = B`
	`B = C`
	
	`print(A)`
	`print(B)`

```
A = 3
B = 8
C = A
A = B
B = C
print(A)
print(B)
```

```
8
3
```

<u>Sebelum pertukaran</u>
Input A, B
Syntax: `A = 3`
`B = 8`



Contoh

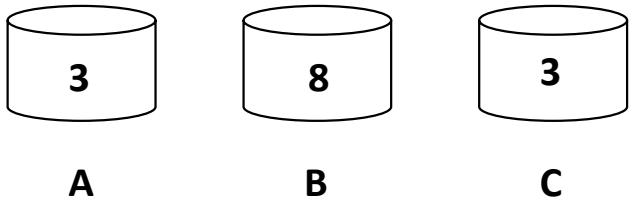
Algoritma	Program (Python)

Input A, B
 	
Masukkan nilai A ke dalam C
Masukkan nilai B ke dalam A
Masukkan nilai C ke dalam B
 	
Cetak A, B
	`A = 3`
	`B = 8`
	
	`C = A`
	`A = B`
	`B = C`
	
	`print(A)`
	`print(B)`

```
A = 3
B = 8
C = A
A = B
B = C
print(A)
print(B)
```

```
8
3
```

<u>Proses pertukaran</u>
Masukkan nilai A ke dalam C
Syntax: `C = A`



Contoh

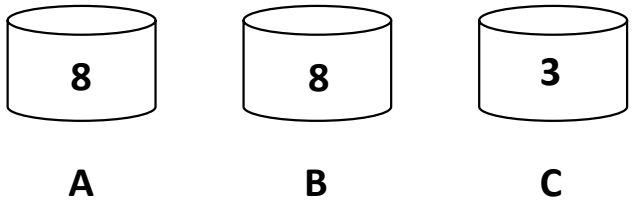
Algoritma	Program (Python)

Input A, B
 	
Masukkan nilai A ke dalam C
Masukkan nilai B ke dalam A
Masukkan nilai C ke dalam B
 	
Cetak A, B
	`A = 3`
	`B = 8`
	
	`C = A`
	`A = B`
	`B = C`
	
	`print(A)`
	`print(B)`

```
A = 3
B = 8
C = A
A = B
B = C
print(A)
print(B)
```

```
8
3
```

<u>Proses pertukaran</u>
Masukkan nilai B ke dalam A
Syntax: `A = B`



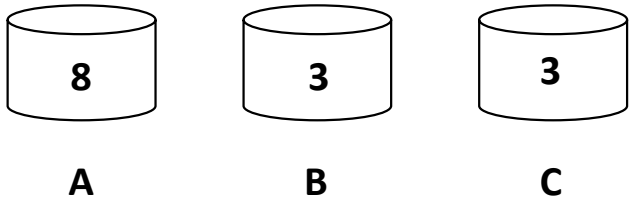
Contoh

Algoritma	Program (Python)

Input A, B
 	
Masukkan nilai A ke dalam C
Masukkan nilai B ke dalam A
Masukkan nilai C ke dalam B
 	
Cetak A, B
	`A = 3`
	`B = 8`
	
	`C = A`
	`A = B`
	`B = C`
	
	`print(A)`
	`print(B)`

```
A = 3
B = 8
C = A
A = B
B = C
print(A)
print(B)
```

```
8
3
```

<u>Proses pertukaran</u> Masukkan nilai C ke dalam B *Syntax*: `B = C`	

Contoh

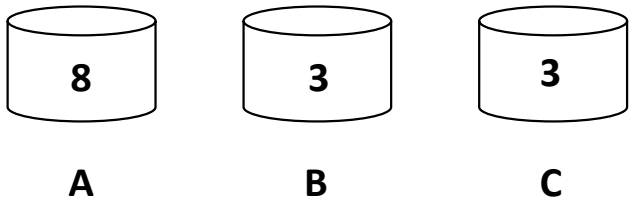
Algoritma	Program (Python)

Input A, B
 	
Masukkan nilai A ke dalam C
Masukkan nilai B ke dalam A
Masukkan nilai C ke dalam B
 	
Cetak A, B
	`A = 3`
	`B = 8`
	
	`C = A`
	`A = B`
	`B = C`
	
	`print(A)`
	`print(B)`

```
A = 3
B = 8
C = A
A = B
B = C
print(A)
print(B)
```

```
8
3
```

Setelah pertukaran
Cetak A, B
Syntax: `print(A)`
`print(B)`



Branching

Dalam program komputer, algoritma sering kali harus memilih untuk melakukan salah satu dari dua hal tergantung pada kondisinya.

Contoh:

```
Jika nilai > 55, maka lulus ujian
```

Pernyataan di atas dapat ditulis dalam pernyataan-pemilihan (selection-statement), atau disebut juga pernyataan-kondisional, sebagai berikut:

```
if kondisi then
   aksi
```

Dalam bahasa Indonesia, **`if`** berarti "jika" dan **`then`** artinya "maka". `Kondisi` adalah persyaratan yang dapat bernilai benar atau salah. `Aksi` sesudah kata **`then`** hanya dilaksanakan apabila `kondisi` bernilai benar. Sebaliknya, apabila `kondisi` bernilai salah, maka `aksi` tidak dilaksanakan.

Branching

Bentuk pemilihan yang lebih umum adalah memilih satu dari dua buah aksi bergantung pada nilai kondisinya.

```
if kondisi then
    aksi 1
else
    aksi 2
```

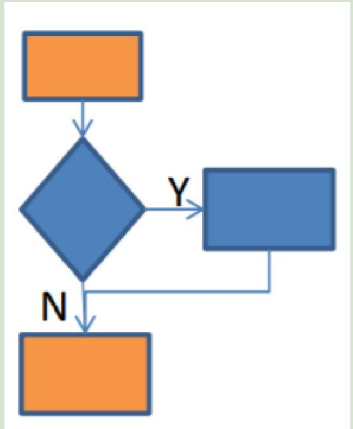
else artinya “kalau tidak”. Bila kondisi terpenuhi, `aksi 1` akan dikerjakan, tetapi kalau tidak (yaitu kondisi salah), `aksi 2` yang akan dikerjakan.

Contoh:

```
if x > y then
    cetak nilai x
else
    cetak nilai y
```

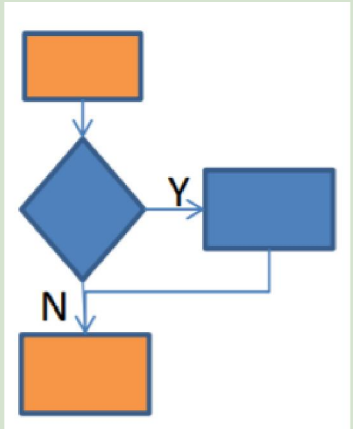
Contoh

Algoritma	Program (Python)

input x
 	
if x > 55 then
  cetak “Selamat, Anda lulus!”
end
 
	`x = 90`
	
	`if x > 55:`
	`  print('Selamat, Anda lulus!')`
	 

Contoh

Algoritma	Program (Python)

input x
 	
if x > 55 then
  cetak “Selamat, Anda lulus!”
end
 
	`x = 90`
	
	`if x > 55:`
	`  print('Selamat, Anda lulus!')`
	 

Looping

Loops – or repeating yourself

Loops memungkinkan untuk mengulangi satu (atau beberapa) baris kode berulang kali. Hal ini memungkinkan kita untuk "menulis sekali" dan kemudian "mengeksekusi berkali-kali“

Terdapat dua *loop* yang perlu diingat:

- ***For** loop*: yang digunakan ketika diketahui berapa kali *loop* akan menjalankan.

```
for i = 1 to 10 do
    print i
```

- ***While** loops*: yang digunakan ketika tidak diketahui berapa kali *loop*, tapi ingin dilanjutkan sampai kondisi tertentu bernilai tidak benar.

```
while i < 10 do
    print i
    i = i + 1
```

Contoh

Algoritma	Program (Python)

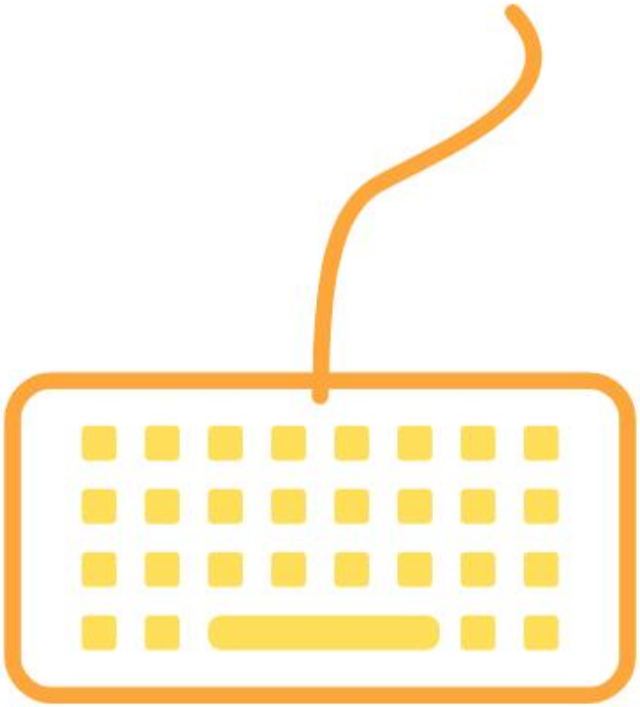
for i = 1 to 5 **do**
  print i
	`for i in range (5):`
	`  print(i)`
	`0 1 2 3 4`
	
	`for i in range(1,6):`
	`  print(i)`
	`1 2 3 4 5`

while i < 6 **do**
  print i
  i = i + 1
	`i = 1`
	`while i < 6:`
	`  print(i)`
	`  i = i + 1`
	`1 2 3 4 5`
	
	`i = 1`
	`while i < 6:`
	`  print(i)`
	`  i += 1`
	`1 2 3 4 5`

Contoh

```
number = int(input('Input any number: '))
count = 0
while (number > 1):
 print('urutan ke',count,'adalah', number)
 number = number / 2
 count = count + 1
```

```
Input any number: 10
urutan ke 0 adalah 10
urutan ke 1 adalah 5.0
urutan ke 2 adalah 2.5
urutan ke 3 adalah 1.25
```



Terima Kasih.

***Credit**: Slide ini merupakan adaptasi dari slide Minggu-8 Pernyataan pada Algoritma dan Pemrograman Saintifik oleh Gatot F. Hertono, Ph.D.*