



# TIB01 - ALGORITMA

# Capaian Pembelajaran Mata Kuliah

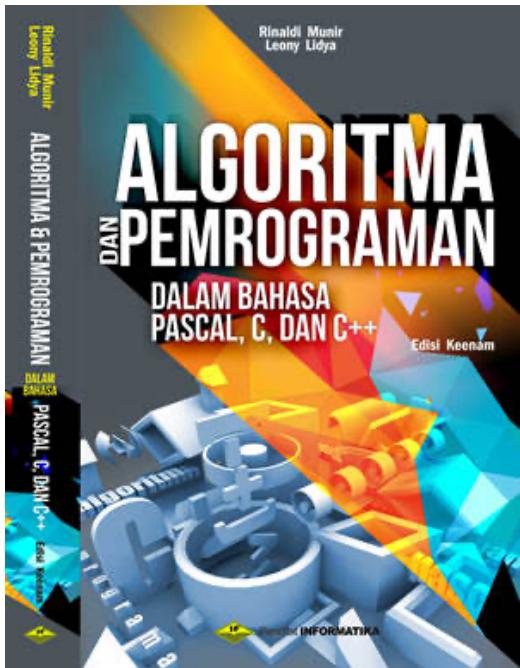
1. Mahasiswa mampu menjelaskan **konstruksi dasar algoritma, tipe data dan penulisan notasi** algoritma (C3, A3)
2. Mahasiswa mampu mengaplikasikan berbagai konsep dasar algoritma seperti **rekursif, array, berbagai metode pencarian dan pengurutan**, serta penyimpanan data di dalam **memori sekunder** (C3, A3)
3. Mahasiswa mampu menggunakan algoritma asymptotic untuk menunjukkan **kompleksitas algoritma** (C3, A3)



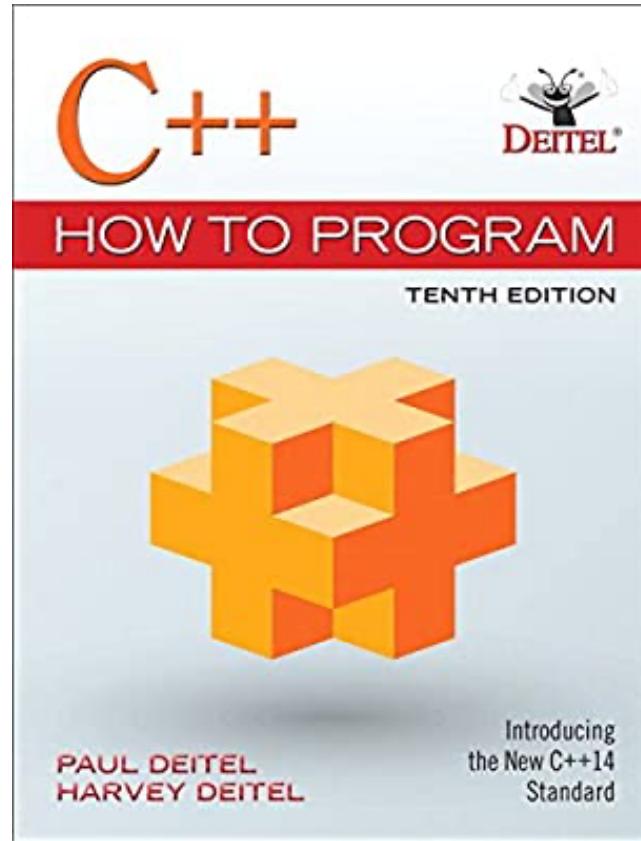
# KONSEP DASAR ALGORITMA DAN PEMROGRAMAN

Pertemuan ke-1 dan 2

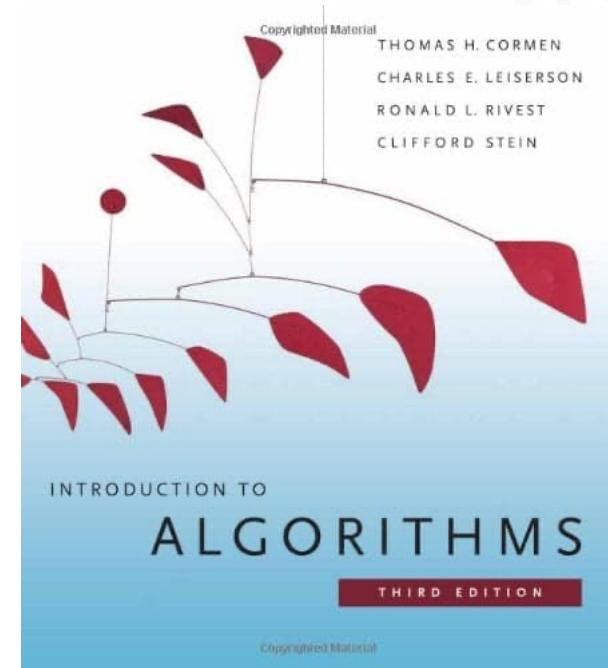
Diadopsi dari sumber :



1



2



3

## Diadopsi dari sumber :

No	Judul	Pengarang	Penerbit	Edisi	Kota	Tahun	Jenis
1	Algoritma dan Pemrograman dalam Bahasa Pascal, C, dan C++	Rinaldi Munir, Leorny Lidya	Penerbit Informatika	6	Bandung	2016	Utama
2	C++ How to Program	Paul Deitel, Harvey Deitel	Pearson Education, Inc.	10	Hoboken, New Jersey	2017	Pendukung
3	Introduction to Algorithms	Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein	MIT Press	3	London, England	2009	Pendukung

# Sub-CPMK

Mahasiswa mampu menjelaskan konsep algoritma dan pemrograman, konstruksi dasar algoritma, tipe data dasar, dan cara menuliskan algoritma dengan notasi tertentu (C2, A2)

## Materi

1. Definisi algoritma, program, pemrograman, dan contoh bahasa pemrograman
2. Persoalan algoritma dalam kehidupan sehari-hari
3. Gambaran umum struktur algoritma dengan program Scratch.
4. Notasi algoritma dalam bentuk narasi, flowchart dan pseudocode
5. Tipe dasar dan tipe bentukan serta berbagai macam ekspresi dalam algoritma
6. Penulisan algoritma untuk program sederhana

# Referensi

1. Referensi 1, Bab 1-5, hal 1-94
2. Referensi 2, Bab 1 poin 1.5-1.7 (hal 10-13), Bab 4 poin 4.1-4.3 (hal 104-105), Bab 5 poin 5.11-5.12 (hal 188-193)



# **1. Definisi algoritma, program, pemrograman, dan contoh bahasa pemrograman**

# 1.1 ALGORITMA

## 1.1.1 Sejarah



Sumber: <https://www.amust.com.au/wp-content/uploads/2018/01/AL-KHWARIZMI-painting-482366.jpg>

## 1.1.2 Pengertian

- Urutan langkah-langkah untuk menyelesaikan suatu persoalan.
- Deretan langkah-langkah komputasi yang mentransformasikan data masukan menjadi luaran.
- Deretan instruksi yang jelas untuk memecahkan persoalan, yaitu untuk memperoleh luaran yang diinginkan dari suatu masukan dalam jumlah waktu yang terbatas

## 1.1.3 Karakteristik

Menurut Donald E. Knuth di dalam Art of Computer Programming, sebuah algoritma harus mempunyai 5 ciri penting:

1. Algoritma harus berhenti setelah mengerjakan sejumlah langkah terbatas.
2. Setiap langkah harus didefinisikan dengan tepat dan tidak boleh berarti-dua (*ambiguous*).
3. Algoritma memiliki nol atau lebih masukan (*input*)
4. Algoritma memiliki nol atau lebih luaran (*output*).
5. Algoritma harus sangkil (*effective*).

## 1.2 PROGRAM DAN PEMROGRAMAN

### PROGRAM:

- Algoritma yang ditulis dalam bahasa komputer

### BAHASA PEMROGRAMAN (*programming language*):

- Bahasa komputer yang digunakan untuk menulis program

### PEMROGRAM (*programmer*):

- Orang yang menulis program komputer

### PEMROGRAMAN (*programming*):

- Kegiatan mulai dari mendesain hingga menulis program

## 1.2 PROGRAM DAN PEMROGRAMAN (Lanj..)

Belajar  
pemrograman

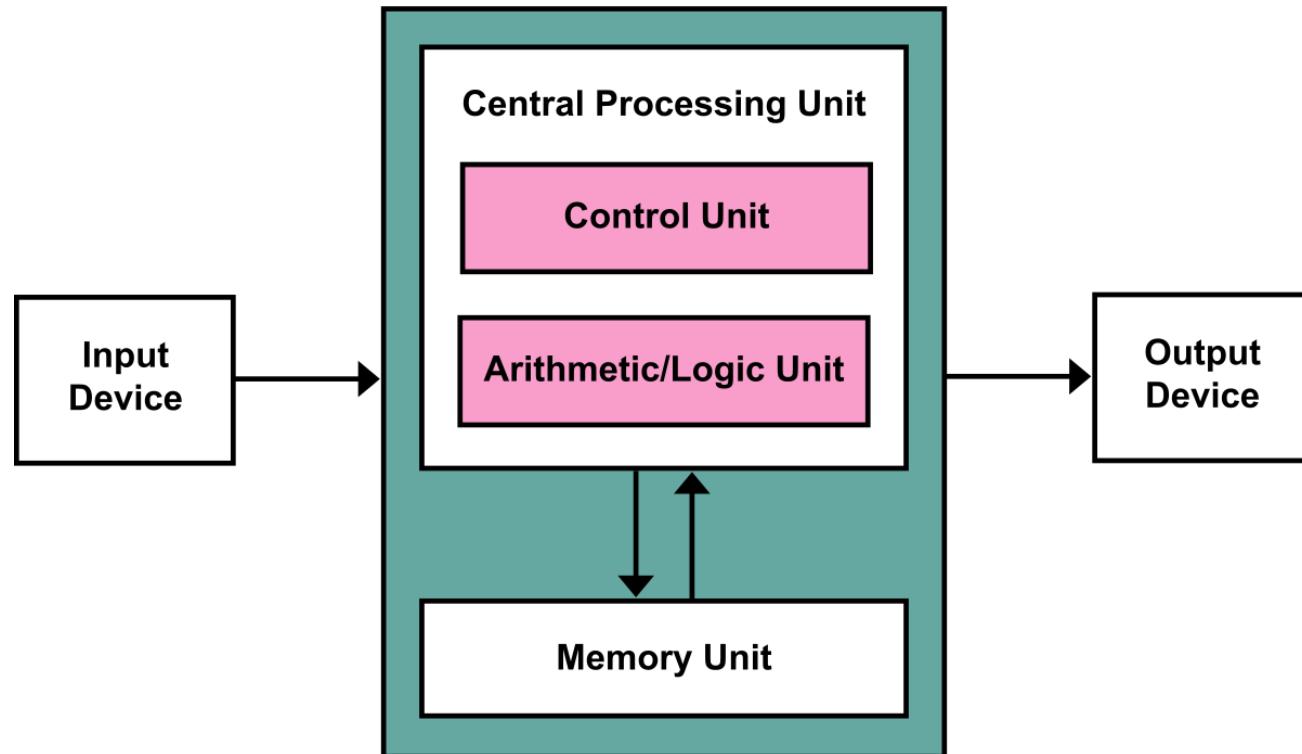


Belajar  
**Bahasa**  
pemrograman

- Belajar pemrograman berarti mempelajari metodologi pemecahan masalah, kemudian menuliskan algoritma pemecahan masalah dengan notasi tertentu.
- Sedangkan belajar bahasa pemrograman berarti belajar memakai suatu bahasa komputer, aturan tata bahasanya, instruksi-instruksinya, tata cara pengoperasian *compiler*-nya, dan memanfaatkan instruksi-instruksi tersebut untuk membuat program yang ditulis hanya dalam bahasa itu saja.

## 1.2 PROGRAM DAN PEMROGRAMAN (Lanj..)

Bagaimanakah komputer bisa menjalankan (*running*) sebuah program?



Arsitektur Von Neumann (Sumber: Wikipedia)

## 1.3 BAHASA PEMROGRAMAN

- Berdasarkan tujuan aplikasinya:
  1. Bahasa pemrograman bertujuan khusus (*specific purpose programming language*): Cobol, Fortran, assembly, dsb.
  2. Bahasa pemrograman bertujuan umum (*general purpose programming language*): Pascal, Basic, C, C++, C#, Java, dsb.

## 1.3 BAHASA PEMROGRAMAN (Lanj...)

- Berdasarkan “kedekatan” bahasa pemrograman dengan bahasa alami (bahasa manusia):
  1. Bahasa tingkat rendah (*low level language*): bahasa mesin dan bahasa *assembly*.
  2. Bahasa tingkat tinggi (*high level language*): Pascal, Ada, Cobol, Basic, Fortran, C, C++, Java

## 1.3 BAHASA PEMROGRAMAN (Lanj...)

High-level Language

```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
TEMP = V(K)
V(K) = V(K+1)
V(K+1) = TEMP
```

C/Java Compiler

Fortran Compiler

```
lw $t0, 0($2)
lw $t1, 4($2)
sw $t1, 0($2)
sw $t0, 4($2)
```

Assembly Language

MIPS Assembler

Machine Language

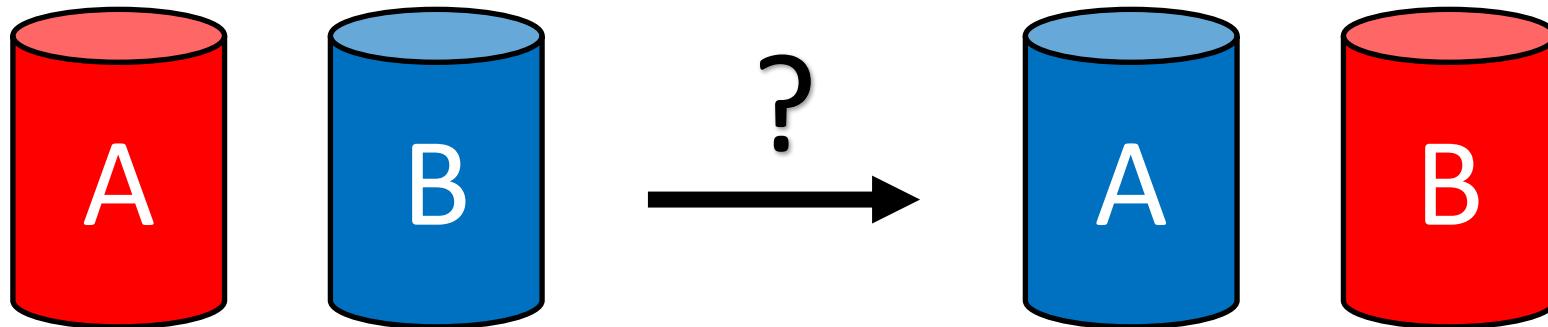
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111

Sumber: <https://www.dictio.id/uploads/db3342/original/3X/e/7/e7b6cfe29f3615c55f74eb9419bb6e87c161dbe1.gif>



## 2. Contoh Persoalan algoritma dalam kehidupan sehari-hari

## 2.1 Persoalan Mempertukarkan Isi Gelas



Misalkan terdapat dua buah gelas, gelas A dan gelas B, yang berisi larutan berwarna. Gelas A berisi larutan berwarna merah, sedangkan gelas B berisi larutan berwarna biru. Volume air di dalam kedua gelas sama banyaknya.

Bagaimana cara mempertukarkan isi kedua gelas tersebut sedemikian sehingga nantinya gelas A berisi larutan berwarna biru dan gelas B berisi larutan berwarna merah?

## Penyelesaian:

Agar pertukaran larutan di kedua gelas dapat dilakukan, kita memerlukan sebuah gelas ketiga sebagai tempat penampungan sementara. Misalkan gelas ketiga tersebut adalah gelas C. Dengan menggunakan gelas C ini, algoritma mempertukarkan isi kedua gelas adalah seperti algoritma berikut ini:

**ALGORITMA** mempertukarkan isi dari dua buah gelas,

A dan B:

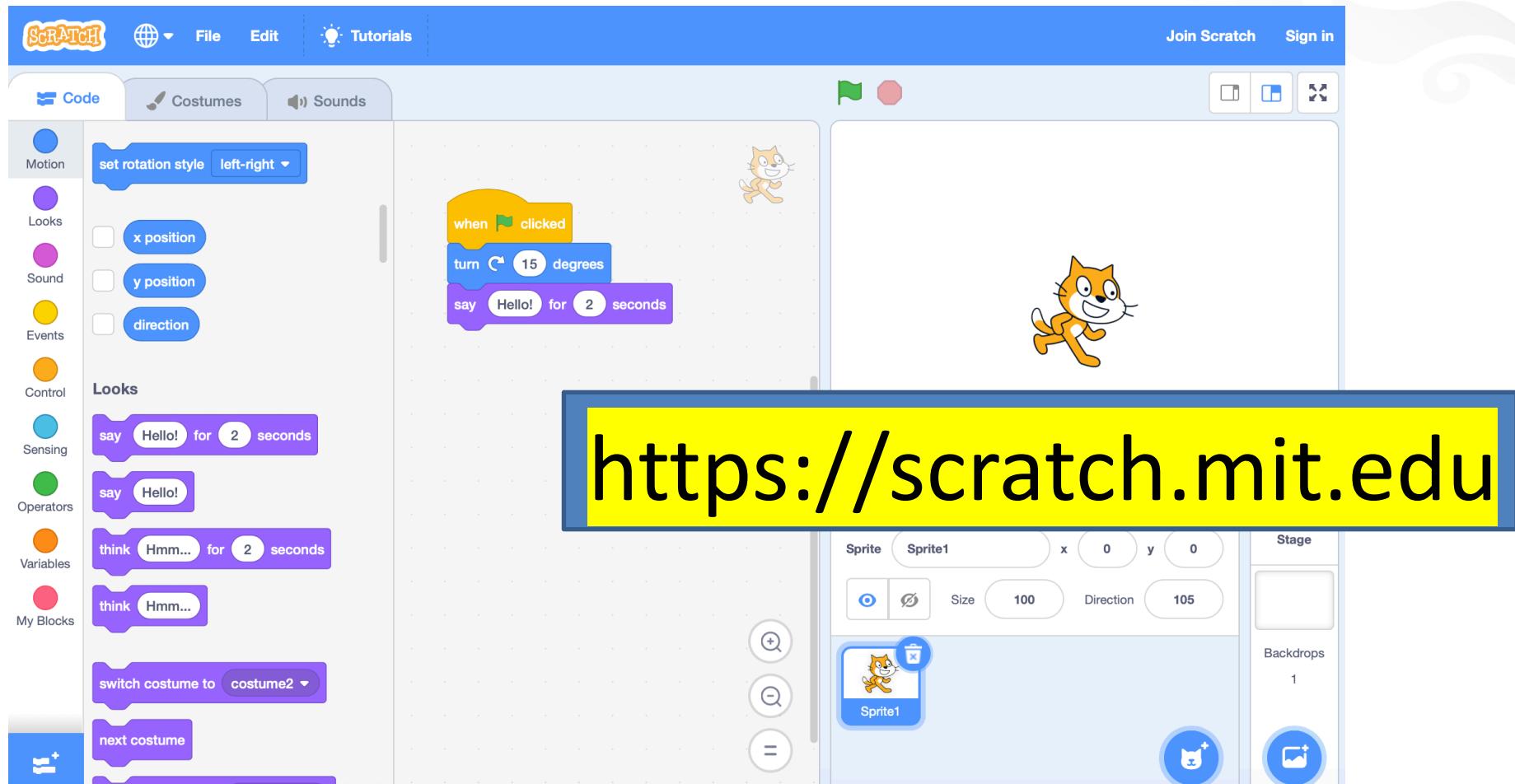
1. Tuangkan larutan dari **gelas A** ke **gelas C**
2. Tuangkan larutan dari **gelas B** ke **gelas A**
3. Tuangkan larutan dari **gelas C** ke **gelas B**

## Pertanyaan Review

Coba tuliskan beberapa contoh algoritma yang lain dalam kehidupn sehari-hari. Tuliskan juga beberapa contoh langkah di dalam algoritmanya.



### 3. Gambaran umum struktur algoritma dengan program Scratch.



The image shows the Scratch programming environment. A yellow banner at the bottom displays the URL <https://scratch.mit.edu>. The stage features a cat sprite running towards the right. In the script editor, a script for the cat sprite is shown:

```
when green flag clicked
  turn (15) degrees
  say [Hello! for 2 seconds]
```

The script uses the "Events" category blocks: "when green flag clicked", "turn (15) degrees", and "say [Hello! for 2 seconds]". The stage has a single backdrop. The sprite properties panel shows the cat sprite is named "Sprite1", has a size of 100, and a direction of 105. The costumers panel shows the cat costume.



## 4. Notasi algoritma dalam bentuk narasi, flowchart dan pseudocode

## 4.1 Notasi Algoritma: Narasi

- Notasi yang digunakan dalam bentuk narasi menyatakan langkah-langkah algoritma dalam deretan kalimat deskriptif.
- Contoh kasus: mencari pembagi bersama terbesar dengan menggunakan algoritma Euclidean.

## 4.1 Notasi Algoritma: Narasi (Lanj..)

### PROGRAM Euclidean

{Diberikan dua buah bilangan bulat tak-negatif m dan n ( $m \geq n$ ). Algoritma Euclidean mencari pembagi bersama terbesar, gcd, dari kedua bilangan tersebut, yaitu bilangan bulat positif terbesar yang habis membagi m dan n}

### ALGORITMA:

1. Jika  $n=0$  maka m adalah jawabannya; stop. Namun jika  $n \neq 0$ , lanjutkan ke langkah 2.
2. **Bagilah** m dengan n dan misalkan r adalah sisa baginya.
3. **Ganti** nilai m dengan nilai n dan **ganti** nilai n dengan nilai r, lalu ulang kembali ke langkah 1.

# 4.2 Notasi Algoritma: Flowchart

## The Definitive Flow Chart Cheat Sheet



**Start/End**  
The beginning or end of any flow



**Process**  
Any action or moment in the flow



**Predefined Process**  
Pre-existing subprocess that isn't described in this diagram



**Decision**  
Branching point, followed by two or more paths



**Document**  
Any brief, form, or other document



**Multiple Documents**  
Multiple briefs, forms, or other documents



**Preparation**  
Set-up necessary for the rest of the flow



**Delay**  
Any waiting period planned into the flow



**Merge**  
The point where separate processes come together



**Connector**  
A jump between separated sections of the flow on one page



**Off-page Connector**  
Flow continues on the next page/from the previous page



**Arrows**  
Bold alternatives to standard connectors

### Data Symbols

Specific to technical flows, these shapes indicate data flow, storage, and display.



**Loop Limit**  
The beginning (or end) of a looped process



**Display**  
Information displayed to a user



**Input/Output**  
Data added to the flow or resulting from the flow



**Manual Input**  
Data that must be entered at a prompt, e.g., filling out a form



**Manual Operation**  
Any adjustment to the flow that must be made manually



**Paper Tape**  
Input into an older computer system



**Card**  
Job control card for mainframe batch processing flows



**Data Storage**  
Data stored in any format



**Internal Storage**  
Data stored locally, not as a remote file



**Database**  
Data that can be accessed in any order

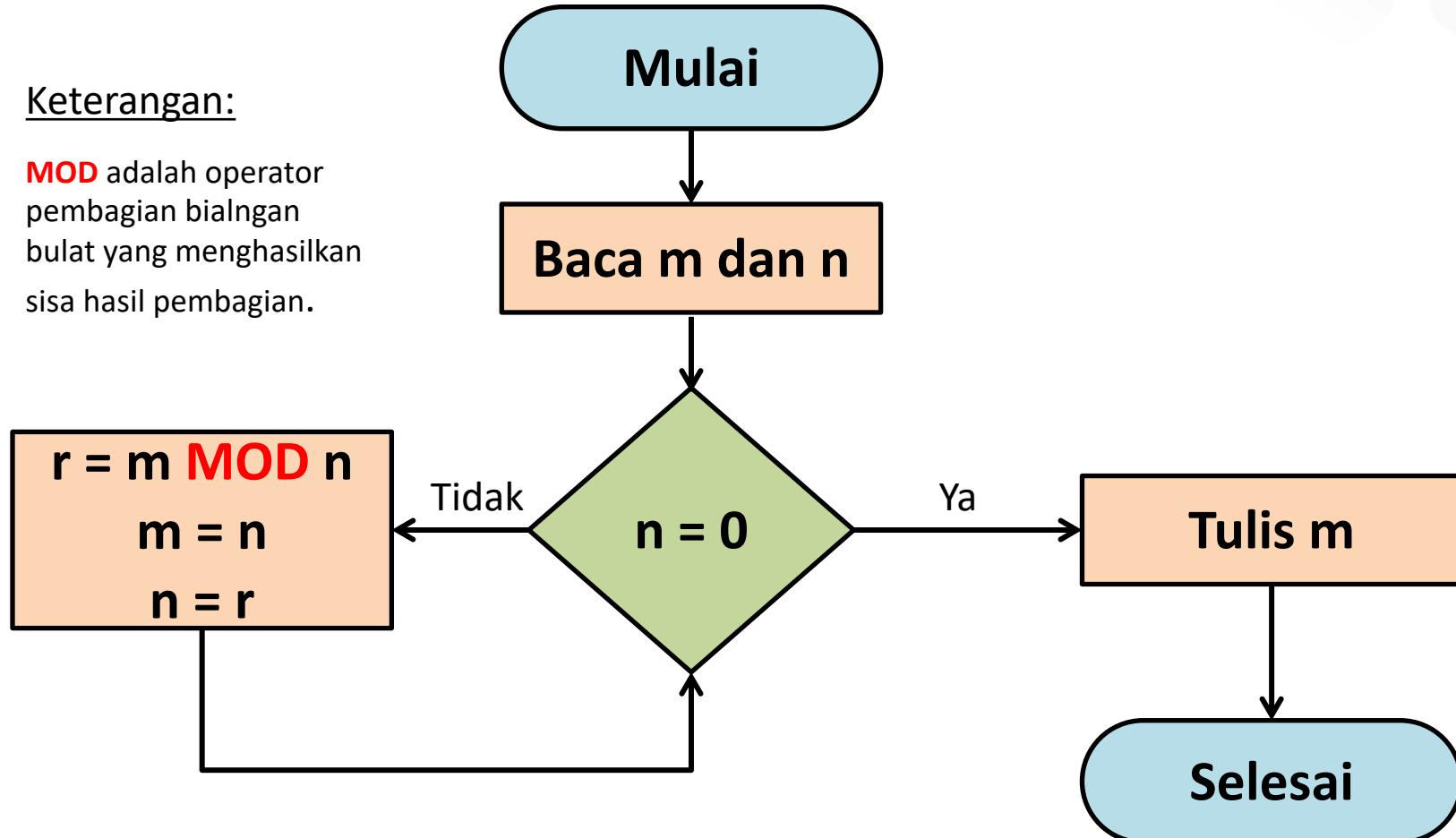


**Tape Data**  
Data that must be accessed sequentially

## 4.2 Notasi Algoritma: Flowchart (Lanj..)

Keterangan:

**MOD** adalah operator pembagian bialngan bulat yang menghasilkan sisa hasil pembagian.



## 4.3 Notasi Algoritma: Pseudo-code

- Bila bahasa pemrograman sangat rigid dalam urusan sintaks seperti tanda titik koma (*semicolon*), cara menulis indeks, format luaran, aturan khusus, dsb, maka sebaliknya, notasi pseudo-code mengabaikannya.
- Keuntungan menggunakan notasi pseudo-code adalah kemudahan mentranslasinya ke dalam notasi bahasa pemrograman, karena terdapat korespondensi antara setiap pseudo-code dengan notasi bahasa pemrograman

## 4.3 Notasi Algoritma: Pseudo-code (Lanj..)

### PROGRAM Euclidean

{Program untuk mencari pembagi bersama terbesar, gcd, dari kedua bilangan tersebut, yaitu bilangan bulan positif terbesar yang habis membagi m dan n ( $m \geq n$ ) .}

#### DEKLARASI:

<b>m, n: integer</b>	{bilangan bulat yang akan dicari gcd-nya}
<b>r: integer</b>	{sisa hasil bagi}

#### ALGORITMA:

<b>read</b> (m, n)	{asumsi: $m \geq n$ }
<b>while</b> $n \neq 0$ <b>do</b>	
$r \leftarrow m \text{ MOD } n$	{bagi m dengan n dan simpan sisanya di r}
$m \leftarrow n$	{ganti nilai m dengan nilai n}
$n \leftarrow r$	{ganti nilai n dengan nilai r}
<b>end while</b>	
{kondisi akhir pengulangan: $n = 0$ }	
<b>write</b> (m)	{m berisi gcd dari kedua bilangan yang dicari}

## 4.3.1 Struktur Teks Algoritma

### 1. JUDUL ALGORITMA

- Judul algoritma adalah bagian yang terdiri atas nama algoritma dan penjelasan (spesifikasi) tentang algoritma tersebut. Nama algoritma sebaiknya singkat, namun cukup menggambarkan apa yang dilakukan algoritma tersebut.
- Di bawah algoritma disertai dengan penjelasan singkat atau intisari tentang apa yang dilakukan oleh algoritma. Penjelasan di bawah nama algoritma sering dinamakan juga **spesifikasi program**.

**PROGRAM** Euclidean

**JUDUL ALGORITMA**

{Program untuk mencari pembagi bersama terbesar, gcd, dari kedua bilangan tersebut, yaitu bilangan bulan positif terbesar yang habis membagi m dan n ( $m \geq n$ ) .}

**SPESIFIKASI PROGRAM**

## 4.3.1 Struktur Teks Algoritma (Lanj..)

### 2. DEKLARASI

- Deklarasi nama adalah bagian untuk mendefinisikan semua nama yang dipakai di dalam algoritma.
- Nama tersebut dapat berupa nama konstanta (tetapan), nama variable (peubah), nama tipe, nama prosedur dan nama fungsi.

## 4.3.1 Struktur Teks Algoritma (Lanj..)

DEKLARASI :

{nama konstanta}

const phi = 3.14 {nilai phi}

{nama tipe}

**type** Titik : **record** {koordinat titik di bidang kartesian}  
<x: **real**,  
y: **real**>

{nama variable (peubah) }

m, n: **integer** {tipe data integer}

P : Titik {titik dalam koordinat kartesian}

.

## 4.3.1 Struktur Teks Algoritma (Lanj..)

### 3. DESKRIPSI

- Deskripsi merupakan bagian inti dari suatu program. Bagian ini berisi langkah-langkah penyelesaian masalah. Misalnya notasi write digunakan untuk mencetak data atau informasi, notasi read untuk membaca data, dsb
- Setiap langkah algoritma, dibaca dari atas ke bawah. Urutan penulisan menentukan urutan perintah.
- Deskripsi pada algoritma, mempunyai kemiripan fungsi pada bagian “cara membuat” dalam sebuah resep masakan.

### ALGORITMA:

```
read(m, n)           {asumsi: m >= n}
while n ≠ 0 do
    r ← m MOD n      {bagi m dengan n dan simpan sisanya di r}
    m ← n             {ganti nilai m dengan nilai n}
    n ← r             {ganti nilai n dengan nilai r}
end while
{kondisi akhir pengulangan: n = 0}

write(m)           {m berisi gcd dari kedua bilangan yang dicari}
```

### Catatan:

Penulisan write dan read dengan garis bawah atau tidak tergantung dari buku referensi atau kesepakatan saja. Biasa digunakan jika menuliskan pseudocode dengan tulisan tangan, karena lebih memudahkan dalam pembacaan.

## 4.3.2 Translasi Teks ke dalam Bahasa Pemrograman

### Program NAMA\_ALGORITMA

{penjelasan tentang algoritma yang berisi uraian singkat mengenai apa yang dilakukan oleh algoritma}

### DEKLARASI:

{semua nama yang dipakai, meliputi nama tipe, nama konstanta, nama variabel, nama prosedur dan nama fungsi didefinisikan di sini}

### DESKRIPSI/ALGORITMA:

{semua langkah/aksi algoritma dituliskan di sini}

# Pertanyaan Review

Buatlah program sederhana dalam pseudocode,  
disertai dengan flowchart/bagan alir dari  
program yang Saudara buat.



## 5. Tipe dasar dan tipe bentukan serta berbagai macam ekspresi dalam algoritma

- Tipe data dikelompokkan atas dua macam, yaitu tipe dasar dan tipe bentukan.
- Tipe dasar adalah tipe yang dapat langsung dipakai, sedangkan tipe bentukan dibentuk dari tipe dasar atau dari tipe bentukan lain yang sudah didefinisikan.
- Suatu tipe diacu dari namanya. Nilai yang dicakup dari tipe tersebut dinyatakan di dalam ranah (domain) nilai.

- Operasi-operasi beserta operator yang dapat dilakukan terhadap tipe tersebut juga didefinisikan.
- Dengan kata lain, suatu tipe dinyatakan dengan namanya, ranah nilai yang dikandungnya, cara menuliskan konstantanya dan operasi yang dapat dilakukan kepadanya.

- Program Komputer memanipulasi data (variabel dan konstanta) di dalam memori.
- **TIPE DATA:** Untuk menyatakan tipe data dari sebuah variabel (peubah) pada Deklarasi.
- **OPERATOR:** Menspesifikasikan operasi apa yang dapat dilakukan terhadap peubah (variabel) dan konstanta.
- **EKSPRESI:** Mengkombinasikan peubah-peubah dan konstanta untuk menghasilkan hasil baru.

## 5.1 Tipe Dasar

Dalam dunia pemrograman, yang termasuk ke dalam tipe dasar adalah:

- Bilangan logika
- Bilangan bulat
- Bilangan riil
- Karakter, dan
- String

# 5.1 Tipe Dasar

## 5.1.1 Bilangan Logika

- NAMA TIPE :
  - Boolean
- RANAH NILAI :
  - Dua buah nilai : Benar (*true*) dan Salah (*false*)
  - bilangan logik :benar → 1, salah → 0
- KONSTANTA :
  - True dan False
- OPERASI :
  - Operasi Logika atau operasi boolean
  - Operasi logika menghasilkan nilai : *true* atau *false*
  - Operator logika : AND, OR dan XOR

## 5.1.1 Bilangan Logika (Lanj..)

a	b	not a	a and b	a or b	a xor b
True	True	False	True	True	False
True	False	False	False	True	True
False	True	True	False	True	True
False	false	True	false	False	False

DEKLARASI :

a, b: **boolean**

## 5.1.1 Bilangan Logika (Lanj..)

Contoh operasi logika : Misalkan X, Y, dan Z adalah peubah (*variabel*) bertipe *Boolean*, dimana: X bernilai *true*, Y bernilai *false*, dan Z bernilai *true*

Maka:

Operasi Logika	Hasil
-----	
( x and y ) or z	true
A and ( y or z )	true
Not (x and z)	false
(y xor z) and y	false

## 5.1.2 Bilangan Bulat

- Merupakan bilangan yang tidak mengandung pecahan desimal, misalnya: 34, 8, 0, -17, 45678901, dsb
- NAMA TIPE :
  - Integer
- RANAH NILAI :

Tipe	Rentang nilai	Format
Byte	0 .. 255	8 bit
Shortint	-128 .. 127	8 bit
Word	0 .. 65535	16 bit
Integer	-32768 .. 32767	16 bit
Longint	-2147483648 .. 2147483647	32 bit

## 5.1.2 Bilangan Bulat (Lanj..)

DEKLARASI :

x : byte

y : **integer**

Maka :

- Peubah X tidak dapat dioperasikan untuk nilai-nilai di atas 255
- Peubah Y tidak dapat dioperasikan untuk nilai-nilai di atas 32767

## 5.1.2 Bilangan Bulat (Lanj..)

KONSTANTA :

- Harus ditulis tanpa mengandung titik desimal:
- Contoh : 78, -14, 7654, 0, 5, 9999, dsb

OPERASI :

1. Operasi aritmetika :

+ (tambah)

- (kurang)

\* (kali)

Div (hasil bagi bilangan bulat)

Mod (sisa hasil bagi)

Contoh :

$3 + 10 \rightarrow$  hasil : 13

$10 \text{ DIV } 3 \rightarrow$  hasil : 3

$10 \text{ MOD } 3 \rightarrow$  hasil : 1

## 5.1.2 Bilangan Bulat (Lanj..)

2. Operasi perbandingan :

<	Lebih kecil
$\leq$	Lebih kecil atau sama dengan
>	Lebih besar
$\geq$	Lebih besar atau sama dengan
=	Sama dengan
$\neq$	Tidak sama dengan

Contoh :

$74 > 101$	False
$17 = 17$	True
$(24 \text{ div } 3) \neq 8$	false

## 5.1.3 Bilangan Riil

- Bilangan yang mengandung pecahan desimal:
- Contoh: 3.65, 0.003, 29.0, .24, dsb
- NAMA TIPE :
  - Real
- RANAH NILAI :

Tipe	Rentang nilai	Format
Real	$2.9 \times 10^{-39}$ .. $1.7 \times 10^{38}$	6 byte
Single	$1.5 \times 10^{-45}$ .. $3.4 \times 10^{38}$	4 byte
Double	$5.0 \times 10^{-324}$ .. $1.7 \times 10^{308}$	8 byte
extended	$3.4 \times 10^{-4932}$ .. $1.1 \times 10^{4932}$	10 byte

## 5.1.3 Bilangan Riil (Lanj..)

- **OPERASI :**

1. **Operasi Aritmatika:** + - \* /

Contoh:

$$6.4 + 5.7 \rightarrow \text{hasil} : 12.1$$

$$8.0 - 2.8 \rightarrow \text{hasil}: 5.2$$

$$10/2.5 \rightarrow \text{hasil}: 4.0 \text{ (operasi bilangan campuran)}$$

$$7.2 * 0.5 \rightarrow \text{hasil} : 3.6$$

## 5.1.3 Bilangan Riil (Lanj..)

### 2. Operasi Perbandingan :

Menghasilkan nilai *boolean* (*true* dan *false*)

<	Lebih kecil
$\leq$	Lebih kecil atau sama dengan
>	Lebih besar
$\geq$	Lebih besar atau sama dengan
$\neq$	Tidak sama dengan

Tipe bilangan riil tidak mengenal operator sama dengan (=), karena bilangan riil tidak bisa disajikan secara tepat oleh komputer.

**Misal:**  $1/3$  tidak sama dengan  $0.33333$ , sebab  $1/3 = 0.33333\dots$ (dengan angka 3 yang tidak pernah berhenti).

## 5.1.4 Karakter

- Mencakup semua huruf abjad, tanda baca, karakter khusus, karakter kosong (*null*).
- **NAMA TIPE:**
  - Char
- **RANAH NILAI:**
  - Adalah semua huruf di dalam alfabet ('a'..'z', 'A'..'Z', angka desimal ('0'..'9'), tanda baca('., ':, '!', dll), operator aritmatika('+' , '-' , dll), karakter khusus('\$', '#', '@', dll)
- **KONSTANTA:**
  - Karakter harus diapit oleh tanda petik tunggal. Contoh : 'h', 'y', ' ', ' ', '9', '\$'

## 5.1.4 Karakter (Lanj..)

- **OPERASI :**
  - Hanya Operasi Perbandingan:

<	Lebih kecil
$\leq$	Lebih kecil atau sama dengan
>	Lebih besar
$\geq$	Lebih besar atau sama dengan
=	Sama dengan
$\neq$	Tidak sama dengan

Contoh :

‘a’ = ‘a’ → hasil: true

‘T’ = ‘t’ → hasil: false

‘y’  $\neq$  ‘y’ → hasil: false

‘m’ < ‘z’ → hasil: true

‘q’ > ‘z’ → hasil : false

## 5.1.5 String

- Adalah untaian karakter dengan panjang tertentu.
- **NAMA TIPE :**
  - String
- **RANAH NILAI :**
  - Deretan karakter yang telah didefinisikan pada ranah karakter.
- **KONSTANTA :**
  - Semua konstanta string harus diapit oleh tanda petik tunggal.

Contoh: ‘BANDUNG’, ‘ganesha’, ‘Jl. Pahlawan No. 76’,  
‘.....’, ‘k768532’, dll.

## 5.1.5 String (Lanj..)

- **OPERASI :**
  1. **Operasi Penyambungan (*Concatenation*):**
    - Operator : + (penyambungan, bukan tambah)

### Contoh:

‘Teknik’ + ‘ Informatika’ → hasil : ‘Teknik Informatika’

‘aaa’ + ‘ bbb’ → hasil: ‘aaa bbb’

‘1’ + ‘2’ → hasil: ‘12’

## 5.1.5 String (Lanj..)

### 2. Operasi Perbandingan menghasilkan nilai *boolean* (*true* dan *false*)

<	Lebih kecil
$\leq$	Lebih kecil atau sama dengan
>	Lebih besar
$\geq$	Lebih besar atau sama dengan
=	Sama dengan
$\neq$	Tidak sama dengan

Contoh:

‘abcd’ = ‘abc’ → hasil: *false*

‘aku’ < ‘AKU’ → hasil: *true*

## 5.2 Tipe Bentukan

- Tipe bentukan adalah tipe yang didefinisikan oleh pemrogram (*user-defined data type*).
- Tipe bentukan disusun oleh satu atau lebih tipe dasar.
- Ada dua macam tipe bentukan:
  1. Tipe dasar yang diberi nama dengan nama tipe baru
  2. Tipe terstruktur

## 5.2.1 Tipe Dasar yang Diberi Nama Tipe Baru

- Contoh:

### **DEKLARASI :**

```
type BilanganBulat : integer  
P : BilanganBulat
```

- Keterangan:

BilanganBulat adalah tipe bilangan bulat yang sama saja dengan tipe integer. Apabila kita mempunyai sebuah peubah (*variable*) yang bernama *P* dan bertipe BilanganBulat, peubah *P* tersebut sama saja bertipe integer.

## 5.2.2 Tipe Terstruktur

field 1

field 2

field 3

...

field N

- **Tipe terstruktur** adalah tipe yang berupa rekaman (*record*). Rekaman disusun oleh satu atau lebih *field*. Tiap *field* menyimpan data dari tipe dasar tertentu atau dari tipe bentukan lain yang sudah didefinisikan sebelumnya.
- Nama rekaman ditentukan sendiri oleh pemrogram, selanjutnya nama rekaman tersebut menjadi nama tipe baru.

## 5.2.2 Tipe Terstruktur (Lanj..)

### Contoh 1:

- Titik dalam koordinat kartesian (x,y)

#### DEKLARASI :

```
type Titik : record <x: real, y: real>
P : Titik
```

x	y
---	---

- Cara mengacu tiap field pada P adalah:

P.x

P.y

## 5.2.2 Tipe Terstruktur (Lanj..)

### Contoh 2:

- Tipe tanggal dalam kalender masehi

#### DEKLARASI :

```
type Tanggal : record <dd: integer,  
                      mm: integer,  
                      yy: integer  
>
```

D : Tanggal

dd      mm      yy

- Cara mengacu tiap field pada D adalah:

D.dd; D.mm; D.yy

## 5.2.2 Tipe Terstruktur (Lanj..)

### Contoh 3:

- Tipe terstruktur yang menyatakan nilai ujian seorang mahasiswa untuk suatu mata kuliah (MK) yang ia ambil.

#### DEKLARASI :

```
type Nilai : record <NIM: integer,  
                      NamaMhs: string,  
                      KodeMK: string,  
                      Nilai: char  
                    >
```

M : Nilai

NIM	NamaMhs	KodeMK	Nilai
-----	---------	--------	-------

- Cara mengacu tiap field pada D adalah:

M.NIM; M>NamaMhs; M.KodeMK; M.Nilai

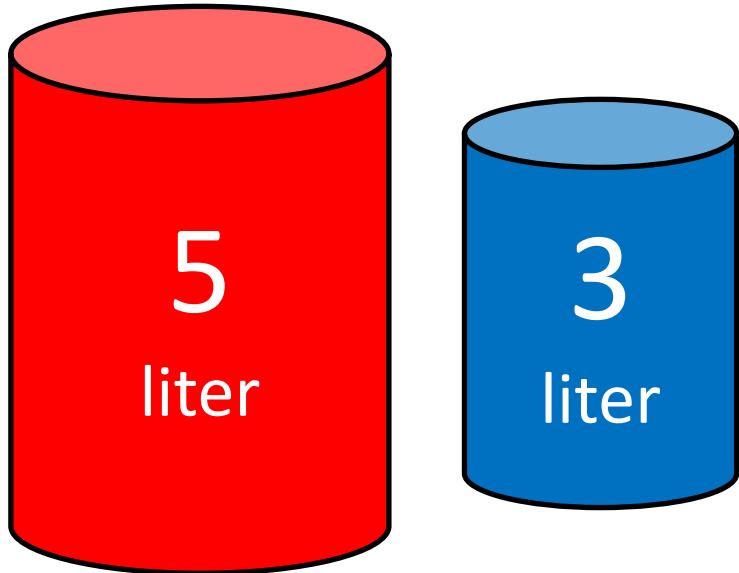
## Pertanyaan Review

Definisikan sebuah tipe terstruktur untuk menyatakan data penerangan di sebuah bandara. Data penerbangan terdiri atas: nomor penerbangan (misal GA101), bandara (kota) asal, bandara tujuan, tanggal keberangkatan, jam keberangkatan (*departure time*), jam datang (*arrival time*). Untuk setiap *field*, definisikan tipe data yang cocok.



## 6. Penulisan algoritma untuk program sederhana

## 6.1 Water Jug Problem



Misalkan Anda mempunyai dua buah ember, masing-masing bervolume 5-liter dan 3-liter. Anda diminta mendapatkan air sebanyak 4 liter (sumber air dari sebuah danau) dengan menggunakan hanya kedua ember tersebut (tidak ada peralatan lain yang tersedia), terserah bagaimana caranya.

Anda boleh memindahkan air dari satu ember, dan sebagainya.  
Bagaimanakan algoritmanya?

## 6.2 Persoalan Petani, Kambing, Serigala, dan Sayur Kubis

Misalkan seorang petani tiba di tepi sungai. Petani tersebut membawa seekor kambing, seekor serigala, dan sekeranjang sayur kubis. Mereka hendak menyeberangi sungai.



Petani itu menemukan sebuah perahu kecil di pinggir sungai tetapi sayang hanya dapat memuat satu bawaan saja setiap kali menyeberang. Situasinya dipersulit dengan kenyataan bahwa serigala tidak bisa ditinggal berdua dengan kambing (karena serigala akan memangsa kambing) atau kambing tidak dapat ditinggal berdua dengan sayur kubis (karena kambing akan memakan sayur).

Bagaimana algoritma si petani menyeberangkan seluruh bawaannya itu sehingga mereka sampai ke seberang sungai dengan selamat?

# 6.3 Pseudocode untuk Program Sederhana

# **PROGRAM** Sapaan

{ Program yang menerima input <nama> dan mencetak sapaan 'Halo <nama>' }

## DEKLARASI:

nama: **string** {variable nama dengan tipe data string}

## ALGORITMA:

```
read(nama)           { membaca <nama> dari input user}
write('Halo', nama) { menampilkan 'Halo <nama>' }
```

## 6.3 Pseudocode untuk Program Sederhana (lanj..)

### **PROGRAM** LuasSegiempat

{Program membaca Panjang (p) dan lebar (l) sebuah segiempat yang berbentuk persegi panjang, menghitung luas segiempat, lalu mencetak hasilnya ke layer.}

#### DEKLARASI:

```
p: real      {panjang segiempat, dalam satuan cm}  
l: real      {lebar segiempat, dalam satuan cm}  
Luas: real   {luas segiempat, dalam satuan cm2}
```

#### ALGORITMA:

```
read(p,l)      {inputkan panjang dan lebar segiempat}  
Luas ← p*l     {hitung luas segiempat}  
write(Luas)    {tampilkan luas segiempat ke layer}
```

## 6.3 Pseudocode untuk Program Sederhana (lanj..)

### **PROGRAM** LuasLingkaran

{Program membaca Panjang jari-jari(r) sebuah lingkaran, menghitung luas lingkaran, lalu mencetak luas tersebut ke layar.}

#### DEKLARASI:

```
const pi=3.14          {konstanta pi}
r: real                {jari-jari linkgaran, dalam satuan cm}
L: real                {luas lingkaran, dalam satuan cm2}
```

#### ALGORITMA:

```
read(r)              {inputkan jari-jari lingkaran}
L ← pi*r*r          {hitung luas segiempat}
write(L)             {tampilkan luas lingkaran ke layar}
```

# Ringkasan

- Program scratch yang dikembangkan oleh MIT dapat membantu saudara untuk memahami algoritma dengan cara yang lebih menyenangkan.
- Notasi algoritma dalam bentuk narasi, flowchart, dan terutama pseudocode dapat membantu saudara untuk menyelesaikan suatu persoalan secara sistematis, sebelum diimplementasikan ke dalam bahasa pemrograman tertentu.
- Tipe data dasar dan bentuknya sangat penting untuk dikuasai, termasuk operasi-operasi yang menyertainya.



# Terima Kasih

UNIVERSITAS BUNDA MULIA