

# BAB I

## PENGANTAR ALGORITMA

### I.1. Pendahuluan

Di masa sekarang ini, peran serta komputer untuk membantu menyelesaikan masalah sudah sangat lekat dalam kehidupan manusia. **Komputer adalah** alat elektronik untuk mengolah data dengan menggunakan program tertentu untuk menghasilkan informasi. Sedang **fungsi dari komputer adalah** untuk pengolahan data (*data processing*), sehingga komputer juga disebut sebagai pengolah data elektronik. Proses dari pengolahan data dengan menggunakan komputer disebut sebagai Pengolahan Data Elektronik (EDP atau *Electronic Data Processing*).

Di dalam komputer, terdapat bagian yang disebut sebagai **Software** (Perangkat Lunak). Perangkat lunak ini merupakan program-program yang diperlukan untuk menjalankan bagian dari komputer yang disebut sebagai **Hardware** (Perangkat Keras). Macam-macam perangkat lunak yang dimaksud di sini adalah :

- ⇒ *Operating System* (Sistem Operasi), adalah program komputer yang diperlukan untuk mengatur semua yang kegiatan sistem komputer., sejak komputer mulai dihidupkan hingga komputer siap dimatikan. Komputer mengendalikan penggunaan semua perangkat yang dipasang di komputer. Contoh *operating system* adalah UNIX, ZENIX, MS DOS, Win NT.
- ⇒ *Language Software* (Bahasa Komputer), adalah bahasa yang digunakan untuk membuat program komputer. Karena digunakan untuk membuat program komputer, bahasa komputer sering disebut juga dengan bahasa pemrograman (*programming language*).
- ⇒ *Application Software* (Program Aplikasi), yaitu program yang diterapkan pada suatu aplikasi tertentu. Komputer diciptakan untuk memenuhi beberapa kebutuhan, sehingga program aplikasi yang dibuat pun bermacam-macam, di antaranya program aplikasi untuk mengolah kata, mengolah data, menggambar, mengaransir musik dan lain sebagainya.

Bahasa pemrograman komputer **digunakan sebagai** sarana komunikasi untuk menjembatani hubungan antara manusia dan komputer. Bahasa pemrograman merupakan suatu prosedur atau tata cara penulisan program, yang dalam hal ini adalah kata, ekspresi,

pernyataan atau kombinasi semuanya yang disusun dan dirangkai dan berupa urutan langkah-langkah untuk menyelesaikan masalah.

Terdapat **beberapa faktor** yang harus diperhatikan dalam bahasa pemrograman, yaitu:

- a. Sintaksis, yaitu tata bahasa yang digunakan dalam program  
Apabila terjadi kesalahan sintaksis, maka akan langsung terlihat, karena komputer akan menampilkan pesan salah.
- b. Semantik, yaitu maksud yang dikandung dalam setiap pernyataan yang ada dalam program.  
Kesalahan semantik biasanya terjadi karena kekurangpahaman terhadap setiap pernyataan yang dituliskan pada program, sehingga walaupun program berjalan tetapi tidak seperti yang dikehendaki.
- c. Kebenaran logika, yaitu berhubungan dengan benar atau tidaknya urutan pernyataan yang ada di dalam program.  
Bentuk kesalahan kebenaran logika merupakan kesalahan dalam mengimplementasikan masalah yang dihadapi, sehingga program yang ditulis tidak benar secara logika.

**Bahasa pemrograman itu sendiri dibagi menjadi dua kelompok, yaitu:**

- a. Bahasa pemrograman tingkat rendah (*low level programming language*)  
Biasanya sulit dipahami karena berhubungan dengan mesin komputer itu sendiri atau biasa disebut sebagai bahasa mesin.  
Contoh : bahasa Assembler.
- b. Bahasa pemrograman tingkat tinggi (*high level programming language*)  
Merupakan bahasa pemrograman yang memakai kata-kata dan pernyataan yang mudah dimengerti manusia, meskipun masih jauh berbeda dengan bahasa manusia sesungguhnya.  
Contoh : Bahasa Pascal, C, C++, **Phyton**

Penyusunan program komputer memerlukan keterangan-keterangan yang berkaitan dengan hal-hal sebagai berikut :

1. Data apa yang akan diproses/tersedia  
Data ini diperlukan untuk menentukan tipe data yang diperlukan oleh program komputer.

2. Bagaimana data dimasukkan

Keterangan ini diperlukan untuk menentukan jenis instruksi masukan yang akan digunakan.

3. Dimana data diletakkan

Keterangan ini diperlukan untuk menentukan variabel-variabel yang harus disediakan dalam program.

4. Operator apa saja yang dapat digunakan

Keterangan ini diperlukan untuk menentukan operator-operator yang diperlukan oleh program.

5. Bagaimana urutan instruksi disusun

Keterangan ini diperlukan untuk menentukan jenis proses yang diperlukan (urutan, kondisional/percabangan, pengulangan)

6. Bagaimana menyampaikan informasi hasil pengolahan

Keterangan ini diperlukan untuk menentukan jenis instruksi keluaran yang akan digunakan.

Suatu **program komputer dikatakan baik jika ia memenuhi** kriteria-kriteria sebagai berikut :

- ✓ Terintegrasi dan memiliki logika yang jelas
- ✓ Efisien terhadap waktu dan penggunaan memori
- ✓ Menerapkan prinsip modularitas
- ✓ Memiliki keluwesan dalam penggunaan
- ✓ Kesederhanaan

## I.2. Arti Penting Algoritma

Di dalam memecahkan suatu persoalan dengan menggunakan komputer, terdapat beberapa tahapan yang harus dilakukan sebelum diimplementasikan dalam sebuah program, yaitu:

1. Menganalisa dan memahami suatu permasalahan yang bertujuan untuk menemukan kemungkinan penyelesaian terhadap permasalahan.
2. Merancang algoritma yang merupakan pola pikir terstruktur yang berisi tahap-tahap penyelesaian suatu permasalahan

3. Membuat program komputer yaitu mengubah kode dari algoritma yang telah dibuat ke dalam pernyataan-pernyataan yang sesuai dengan bahasa pemrograman yang dipakai [*coding*]
4. Menjalankan program secara rutin untuk menemukan kesalahan-kesalahan dalam penulisan suatu pernyataan dalam program [*testing*] dan menemukan kesalahan-kesalahan dalam program dan kesalahan yang ditemukan diperbaiki sampai tidak muncul kesalahan lagi [*debugging*]
5. Melakukan dokumentasi terhadap setiap langkah yang dilakukan [*documentation*]

Tahap 1 dan 2 di atas merupakan fase penyelesaian masalah (*problem solving phase*) dan tahap 3, 4 dan 5 termasuk dalam fase implementasi (*implementation phase*).

Kriteria yang harus dipenuhi oleh **prosedur penyelesaian masalah** dengan algoritma adalah:

1. Setiap langkah harus bersifat pasti atau tertentu (*definite*)
2. Terdapat setidaknya satu keluaran (*output*)
3. Terstruktur dan sistematis
4. Memiliki kriteria untuk menghentikan proses

Algoritma yang baik harus bersifat efisien waktu dan penggunaan memori komputer. Hasil akhir fase penyelesaian masalah adalah penyelesaian dalam bentuk algoritma.

Algoritma merupakan pola pikir yang terstruktur untuk menyelesaikan suatu masalah. Dalam algoritma ini berisi langkah-langkah yang diperlukan oleh seorang pemrogram untuk membuat sebuah program. Jadi dapat dikatakan bahwa algoritma adalah himpunan berhingga langkah-langkah atau prosedur-prosedur logika yang harus dilaksanakan untuk menyelesaikan suatu masalah yang berorientasi pada pemrograman komputer.

**Tujuan algoritma adalah** memberikan petunjuk tentang langkah-langkah logika penyelesaian masalah dalam bentuk yang mudah dipahami nalar manusia sebagai acuan yang membantu dalam mengembangkan program komputer. Pemahaman terhadap algoritma akan mencegah sejak dini kemungkinan terjadinya kesalahan logika pada program komputer yang dikembangkan.

Untuk mendapatkan tujuan di atas, ada lima syarat yang harus dipenuhi oleh algoritma yang dibuat. **Kelima syarat tersebut adalah sebagai berikut:**

1. Logika prosedur pada algoritma harus cukup mudah dipahami nalar manusia
2. Validitas prosedur pada algoritma dapat ditelusuri dengan mudah
3. Tidak menimbulkan kerancuan interpretasi bagi orang lain
4. Prosedur pada algoritma harus cukup mudah dikonversi ke program komputer
5. Prosedur pada algoritma tidak terpengaruh atau tergantung pada bahasa pemrograman apa pun

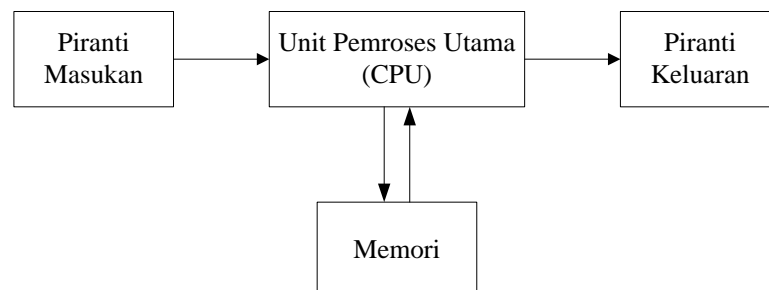
### I.3. Program Terstruktur dan Algoritma

Program terstruktur memberikan beberapa keuntungan, antara lain :

- ✓ Penulisan program menjadi lebih teratur
- ✓ Program tersusun secara sistematis
- ✓ Program tersusun secara terstruktur
- ✓ Lebih mudah dipahami
- ✓ Urutan atau alur proses dalam program menjadi sederhana dan mudah dipahami

Contoh bahasa pemrograman terstruktur adalah Pascal, C, C++.

Secara garis besar, komputer tersusun dari empat buah komponen utama: piranti masukan, unit pemroses utama (CPU = *Central Processing Unit*), piranti keluaran, dan memori. Piranti masukan dan piranti keluaran (*I/O devices*) adalah peralatan-peralatan yang digunakan untuk memasukkan data atau program ke dalam memori dan peralatan-peralatan yang digunakan untuk menampilkan hasil dari suatu operasi. Unit pemroses utama (CPU) adalah unit yang mengerjakan seluruh operasi-operasi dasar seperti operasi perhitungan, operasi perbandingan, operasi membaca dan operasi menulis. Memori adalah komponen yang berfungsi sebagai penyimpan. Yang disimpan di dalam memori adalah program (berisi operasi-operasi yang akan dikerjakan oleh CPU) dan data atau informasi yang telah diolah oleh operasi-operasi.



Gambar 1.1. Komponen utama komputer

Secara umum, struktur proses dalam algoritma dikelompokkan menjadi tiga bagian, yaitu:

1. Proses urutan (*sequence*)

Prosedur proses dalam algoritma yang dilakukan secara urut langkah demi langkah.

Sebuah urutan terdiri dari satu atau lebih instruksi. Tiap instruksi dilaksanakan secara berurutan sesuai dengan urutan pelaksanaan, artinya suatu instruksi akan dilaksanakan setelah instruksi sebelumnya telah selesai dilaksanakan.

2. Proses pemilihan (*selection*)

Suatu instruksi dikerjakan jika suatu kondisi tertentu dipenuhi. Dengan adanya proses ini maka ada kemungkinan beberapa jalur aksi yang berbeda berdasarkan kondisi yang ada.

3. Proses pengulangan (*looping*)

Suatu proses melakukan eksekusi suatu program secara berulang-ulang pada suatu blok instruksi tertentu yang terkendali.

Struktur proses ini akan dijelaskan lebih detail pada bab-bab berikutnya yang ada di dalam buku ini.

#### **I.4. Notasi Algoritma**

Notasi algoritma bukan merupakan notasi bahasa pemrograman. Namun notasi ini dapat diterjemahkan ke dalam berbagai bahasa pemrograman. Penotasian algoritma harus dilakukan sedemikian hingga mudah dibaca dan dimengerti. Meskipun notasi algoritma tidak berbentuk baku seperti notasi bahasa pemrograman, namun konsistensi terhadap notasi perlu diperhatikan untuk menghindari terjadinya kekeliruan.

Pada umumnya, algoritma dapat dinyatakan atau dinotasikan dalam tiga bentuk, yaitu:

1. Uraian deskriptif
2. Diagram-alir (*flowchart*)
3. *Pseudocode*

##### **I.4.1. Uraian Deskriptif**

Dengan notasi bergaya uraian, deskripsi setiap langkah dijelaskan dengan bahasa yang gamblang. Proses diawali dengan kata kerja seperti 'baca' atau 'membaca', 'hitung'

atau ‘menghitung’, ‘bagi’ atau ‘membagi’, ‘ganti’ atau ‘membagi’, dan sebagainya, sedangkan pernyataan kondisional dinyatakan dengan ‘jika ... maka ...’.

Sebagai contoh, misalkan untuk menyelesaikan permasalahan menghitung luas dan keliling suatu lingkaran. Maka, algoritma dalam bentuk uraian deskriptifnya adalah sebagai berikut:

#### **Algoritma Hitung\_Luas\_dan\_Keliling\_Lingkaran**

DESKRIPSI:

- (1) Masukkan jari-jari lingkaran ( $r$ )
- (2) Hitung luas lingkaran dengan rumus  $L = \pi * r^2$
- (3) Hitung keliling lingkaran dengan rumus  $K = 2 * \pi * r$
- (4) Tampilkan luas lingkaran
- (5) Tampilkan keliling lingkaran

Notasi ini bagus jika digunakan dalam algoritma yang pendek, akan tetapi jika digunakan dalam algoritma yang besar, notasi ini tidak efisien. Selain itu, perkonversian notasi algoritma ke dalam notasi bahasa pemrograman akan relatif sukar.

#### **I.4.2. Diagram-alir (*flowchart*)**

*Flowchart* adalah gambaran dalam bentuk diagram alir dari algoritma-algoritma dalam suatu program, yang menyatakan arah alur program tersebut.

Diagram alir lebih menggambarkan aliran instruksi di dalam program secara visual daripada memperlihatkan struktur program.

Gambar 1.2 adalah contoh permasalahan menghitung luas dan keliling lingkaran yang algoritmanya dinotasikan dalam bentuk diagram alir (*flowchart*).

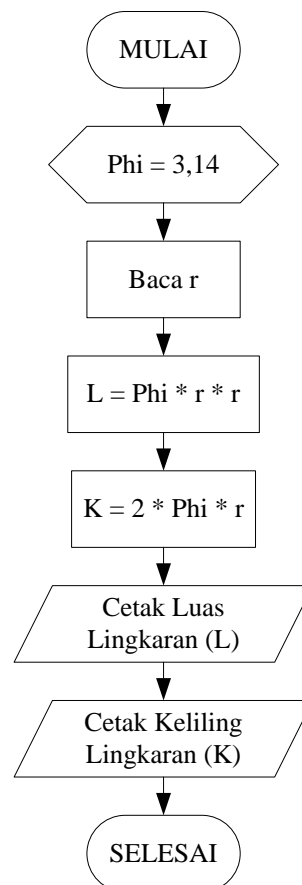
Pada Tabel 1.1, berisi macam-macam simbol yang digunakan dalam diagram alir (*flowchart*) beserta fungsinya.

#### **I.4.3. Pseudocode**

*Pseudocode* adalah notasi yang menyerupai notasi bahasa pemrograman tingkat tinggi, khususnya Bahasa Pascal dan C. Dalam beberapa bahasa pemrograman umumnya mempunyai notasi yang hampir mirip untuk beberapa instruksi, seperti notasi *if-then-else*, *while-do*, *repeat-until*, *read*, *write*, dan sebagainya. Tidak seperti bahasa pemrograman yang direpotkan dengan tanda titik koma (*semicolon*), indeks, format keluaran, kata-kata

husus dan sebagainya, sembarang versi *pseudocode* dapat digunakan asalkan perintahnya tidak membingungkan pembaca.







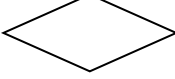


Keuntungan menggunakan notasi *pseudocode* adalah kemudahan mengkonversinya atau mentranslasinya ke notasi bahasa pemrograman, karena terdapat korespondensi antara setiap *pseudocode* dengan notasi bahasa pemrograman. Korespondensi ini dapat diwujudkan dengan tabel translasi dari notasi algoritma ke notasi bahasa pemrograman apapun.



Gambar 1.2. Diagram alir untuk menyelesaikan algoritma menghitung luas dan keliling lingkaran



Tabel 1.1. Simbol-simbol dalam diagram alir

SIMBOL	NAMA	FUNGSI
	<b>TERMINATOR</b>	Permulaan/ akhir program
	<b>GARIS ALIR (FLOW LINE)</b>	Arah aliran program
	<b>PREPARATION</b>	Proses inisialisasi/ pemberian harga awal
	<b>PROSES</b>	Proses perhitungan/ proses pengolahan data
	<b>INPUT/ OUTPUT DATA</b>	Proses input/ output data, parameter, informasi
	<b>PREDEFINED PROCESS (SUB PROGRAM)</b>	Permulaan sub program/ proses menjalankan sub program
	<b>DECISION</b>	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	<b>ON PAGE CONNECTOR</b>	Penghubung bagian-bagian flowchart yang berada pada satu halaman
	<b>OFF PAGE CONNECTOR</b>	Penghubung bagian-bagian flowchart yang berada pada halaman berbeda

Contoh penyelesaian dengan menggunakan bentuk pseudocode untuk menyelesaikan permasalahan menghitung luas dan keliling suatu lingkaran adalah sebagai berikut:

**Algoritma Hitung\_Luas\_dan\_Keliling\_Lingkaran**

{ Dimasukkan nilai jari-jari lingkaran (phi). Carilah dan cetak luas dan keliling lingkaran tersebut dengan menggunakan rumus  $L=\phi*r*r$  dan  $K=2*\phi*r$  }

DEKLARASI:

const phi = 3.14

r, L, K = real

DESKRIPSI:

read(r)

$L \leftarrow \phi * r * r$

$K \leftarrow 2 * \phi * r$

write('Luas lingkaran = ',L)

write('Keliling lingkaran = ',K)