

# BAB I

## PENGANTAR ALGORITMA

### 1. PENDAHULUAN

Bab ini memiliki kompetensi dasar untuk memahami konsep dasar algoritma, pemrograman dan bahasa pemrograman.

Komputer atau Hardware dibuat sebagai alat bantu untuk menyelesaikan masalah. Permasalahan apapun dapat diselesaikan dengan komputer asalkan dengan langkah-langkah yang tepat dan jelas yang disediakan oleh manusia. Bagaimana cara mendeskripsikan masalah agar dapat diselesaikan dengan komputer?

**Caranya** adalah:

1. menjabarkan masalah
2. merinci langkah untuk menyelesaikan masalah
3. membuat sarana interaksi manusia-komputer

Langkah untuk menyelesaikan suatu masalah disebut dengan program komputer. **Program komputer** adalah urutan langkah kerja dalam bahasa pemrograman komputer.

**Tranformasi masalah** menjadi program komputer diperlukan:

1. bentuk urutan masalah
2. bahasa yang dipakai
3. konsep mesin computer

Bagaimana cara mengembangkan dan menganalisa langkah-langkah penyelesaian masalah tanpa tergantung pada karakteristik bahasa yang dipakai ataupun sifat mesin yang digunakan? Hal inilah yang melatarbelakangi mengapa diperlukannya sebuah Algoritma.

### 2. PENYAJIAN

#### 2.1. Algoritma

**Apakah Algoritma itu?** Asal mula kata Algoritma adalah *Algorism* yang berasal dari nama penulis buku Arab yaitu Abu Ja'far Muhammad ibnu Musa Al-Khuwarizmi.

**Keuntungan pemakaian algoritma** adalah: logika pemecahan masalah dapat dibuat bertingkat (mulai dari global menuju terperinci), algoritma merupakan bentuk fleksibel untuk diterapkan ke berbagai bahasa pemrograman.

Jadi **Algoritma** adalah:

1. penyusunan aspek proses logika dari suatu pemecahan masalah tanpa melihat karakteristik bahasa pemrograman yang akan digunakan
2. urutan notasi logika yang merupakan hasil analisis dan rancangan sistematis dari strategi pemecahan masalah, untuk menggambarkan urutan langkah kerja yang jika dikerjakan akan membawa ke tujuannya.
3. urutan logika langkah kerja untuk menyelesaikan suatu masalah.

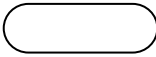


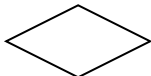
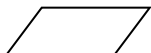

Contoh algoritma dalam kehidupan sehari-hari, misalnya: menjahit pakaian, membuat kue, jadwal harian, panduan merakit komputer, dan lain-lain.

## 2.2. Notasi Algoritma independen dengan bahasa pemrograman dan mesin komputer

Beberapa notasi yang digunakan dalam penulisan algoritma :

1. Notasi I : untaian kalimat deskriptif
2. Notasi II : diagram alir (flow chart)
3. Notasi III : pseudo-code

Simbol-simbol program flowchart

No.	Simbol	Gambar	Keterangan
1	Terminal		Digunakan untuk menunjukkan awal dan akhir dari program
2	Persiapan		Digunakan untuk memberikan nilai awal pada suatu variabel
3	Pengolahan/Proses		Digunakan untuk pengolahan aritmatika dan pemindahan data
4	Keputusan		Digunakan untuk mewakili operasi perbandingan logika
5	Input/Output		Digunakan untuk menyatakan proses input/baca dan output/tulis
6	Garis		Digunakan untuk menyatakan urutan pelaksanaan, atau alur proses

Contoh masalah : menghitung luas segiempat.

**Notasi I :**

Algoritma Luas\_Segiempat

*Menghitung luas segiempat dengan memasukkan nilai lebar dan panjang segiempat*

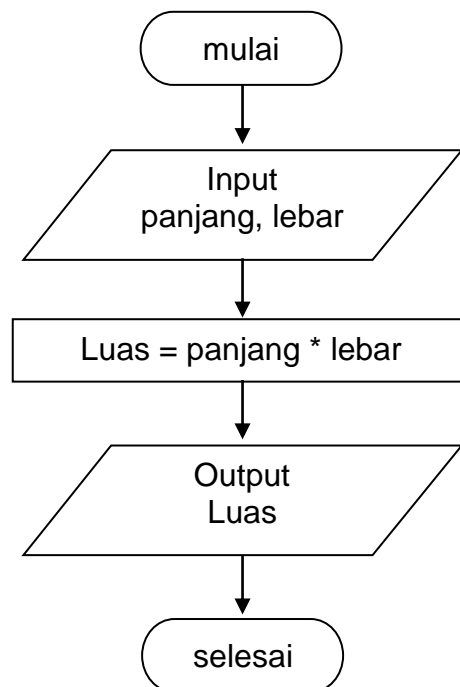
Deklarasi

Luas,panjang,lebar : bilangan bulat

Deskripsi

1. Masukkan nilai lebar dan panjang
2. Hitung luas sama dengan panjang kali lebar
3. Tampilkan Luas

**Notasi II :**



**Notasi III :**

Algoritma Luas\_Segiempat

*Menghitung luas segiempat dengan memasukkan nilai lebar dan panjang segiempat*

Deklarasi

Luas,panjang,lebar : integer

Deskripsi

Input(n)

Luas = panjang \* lebar

Output(Luas)

### 2.3. Program

**Program** adalah logika pemecahan masalah dalam bahasa pemrograman tertentu untuk diproses oleh computer.

**Program yang baik** mempunyai syarat:

1. benar, yaitu bersih dari syntak error, run time error, ataupun logic error
2. berlaku umum untuk beragam data (valid)
3. mudah dibaca (dilengkapi dengan komentar dan keterangan)
4. mudah dimodifikasi dan dikembangkan
5. efisiensi dalam penggunaan ruang dan waktu (kompleksitas rendah)

### Belajar Memogram Vs Belajar Bahasa Pemograman

<u>Belajar</u> strategi pemecahan masalah metodologi dan sistematika pemecahan masalah.	<u>Belajar</u> memakai suatu bahasa pemrograman tertentu.
<u>Bersifat</u> pemahaman persoalan, analisis dan sistesis.	<u>Bersifat</u> ketrampilan.
<u>Membentuk</u> seorang desainer sekaligus programmer.	<u>Membentuk</u> seorang programmer murni.
<u>Tujuannya</u> mencegah memprogram dengan cara <i>trial and error</i> .	<u>Tujuannya</u> mempunyai keterampilan menggunakan suatu bahasa pemograman.
<u>Produknya</u> sebuah program dari hasil rancangan yang metodologis sistimatis.	<u>Produknya</u> sebuah program yang belum dapat dipastikan bersih dari salah logika.

### Paradigma Pemrograman

Komputer digunakan sebagai alat bantu penyelesaian suatu persoalan. Masalahnya, problematika itu tidak dapat “disodorkan” begitu saja ke depan komputer, lalu komputer dapat memberikan solusi dan jawaban.

Ada “jarak” antara persoalan dengan komputer. Jadi, strategi pemecahan masalahnya harus lebih dahulu ditanamkan ke dalam komputer dalam bentuk program. Ilmu memprogram berkembang sehingga memprogram dengan cara *trial and error* harus diganti dengan “seni” memprogram yang baik (lihat syarat program yang baik).

Program harus dihasilkan dari proses pemahaman masalah dan analisis, untuk dituangkan menjadi kode bahasa pemrograman secara sistematis dan metodologis. Untuk menghasilkan suatu program, seorang *programmer* dapat memakai berbagai pendekatan.

## **Beberapa Paradigma dalam Pemrograman**

### **a. *Prosedural / Terstruktur***

Paradigma ini mengkonstruksi program dari struktur data dan algoritma. Paradigma ini berdasarkan konsep Von Newman : ada sekelompok memori yang dibedakan menjadi memori instruksi dan memori data. Instruksi akan dieksekusi satu per satu secara sekuensial oleh prosesor tunggal. Data diperiksa dan dimodifikasi secara sekuensial pula. Pemrograman dalam paradigma ini tidak “alamiah” karena *programmer* diharuskan berpikir dalam batasan mesin/komputer. Karena dekat dengan mesin maka keuntungan menggunakan paradigma ini yaitu program yang dihasilkan dapat berjalan secara efisiensi.

### **b. *Paradigma Fungsional***

Paradigma ini didasari oleh konsep pemetaan dan fungsi pada matematika. Pemrogram mengasumsikan bahwa ada fungsi-fungsi dasar yang dapat dilakukan, penyelesaian masalah didasari atas aplikasi dari fungsi-fungsi tersebut. Semua tingkah laku program merupakan suatu rantai transformasi dari sebuah keadaan awal menuju ke suatu rantai keadaan akhir, yang mungkin melalui keadaan antara.

Paradigma ini tidak lagi memperlakukan memorisasi dan struktur data, tidak ada pemilahan antara data dan program, tidak ada lagi pengertian tentang “variable”. *Programmer* tidak perlu lagi mengetahui bagaimana mesin mengeksekusi dan bagaimana data disimpan.

### **c. *Paradigma Deklaratif / Logika***

Paradigma ini mengkonstruksi program dari kumpulan fakta dan aturannya. Paradigma ini didasari oleh bagaimana membuat relasi antar individu yang dinyatakan dengan predikat.

Karena program logika terdiri atas kumpulan fakta dan kumpulan aturan, maka tugas *programmer* mengendalikan pencocokan *goal* yang akan dituju, melewati fakta-fakta yang ada.

#### **d. Paradigma Object-Oriented**

Paradigma ini mengkonstruksi program dari objek-objek dalam ruang lingkup masalahnya. Sekumpulan objek yang mempunyai sifat yang sama. Dapat menjadi sebuah kelas. Sebuah kelas mempunyai *attribute* (sekumpulan sifat/ciri). Paradigma ini menawarkan konsep modularitas, penggunaan ulang, dan kemudahan modifikasi.

#### **e. Paradigma Konkruen**

Paradigma ini dipengaruhi oleh arsitektur perangkat keras yang memungkinkan proses dijalankan secara parallel.

Berdasarkan paradigma tersebut maka bahasa pemrograman dapat dikelompokkan :

<b>Paradigma</b>	<b>Bahasa Pemrograman</b>
Prosedural / terstruktur	Pascal, C, basic, Java, dll
Fungsional	Logo, Apl, LISP, dll
Deklaratif / logika	Prolog
<i>Object-Oriented</i>	Smalltalk, Java, C++, dll

Ada lagi kelompok bahasa pemrograman yang menyembunyikan kode pemrograman saat merancang sarana interaksi manusia-komputer. Kelompok ini juga menyediakan sarana *object-oriented*. Kelompok bahasa ini disebut *event-programming*.

### **3. PENUTUP**

Latihan :

1. Apakah Algoritma itu ?
2. Buatlah algoritma dengan mengambil contoh dalam kehidupan sehari-hari !