BAB I
KONSEP DASAR PEMROGRAMAN

I.1. Program dan Pemrograman

Kata program dapat diartikan:

a. Untuk mendeskripsikan instruksi-instruksi tersendiri, yang biasanya disebut *source code*, yang dibuat oleh *programmer*.

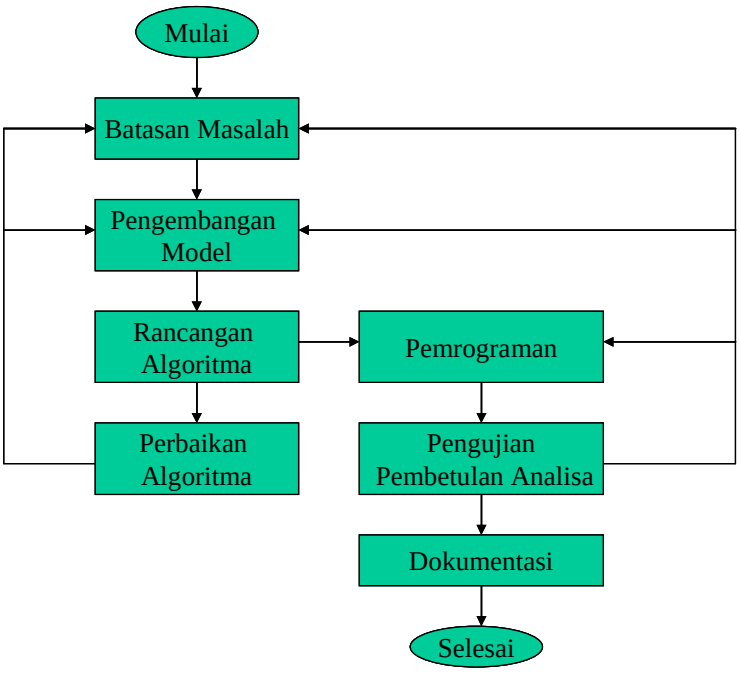
b. Untuk mendeskripsikan suatu keseluruhan bagian dari *software* yang *executable*.

Dapat juga dikatakan bahwa sebuah **program** merupakan himpunan atau kumpulan instruksi tertulis yang dibuat oleh programmer atau suatu bagian *executable* dari suatu *software*.

Kata **pemrograman** dapat diartikan sebagai cara membuat program; dalam konteks ini berarti membuat program komputer. Dapat juga dikatakan bahwa pemrograman merupakan suatu kumpulan urutan perintah ke komputer untuk mengerjakan sesuatu. Perintah-perintah ini membutuhkan suatu bahasa tersendiri yang dapat dimengerti oleh komputer.

Program adalah kata, ekspresi, pernyataan atau kombinasi yang disusun dan dirangkai menjadi satu kesatuan prosedur yang berupa urutan langkah untuk menyelesaikan masalah dan diimplementasikan dengan menggunakan bahasa pemrograman sehingga dapat dieksekusi oleh komputer

Pemrograman adalah proses mengimplementasikan urutan langkah untuk menyelesaikan suatu masalah dengan menggunakan suatu bahasa pemograman



Gambar 1.1. Tahap pengembangan program

Bahasa adalah suatu sistem untuk berkomunikasi. Bahasa tertulis menggunakan simbol (yaitu huruf) untuk membentuk kata. Bahasa merupakan suatu kumpulan simbol-simbol atomic yang terbatas. Kumpulan simbol ini disebut alphabet. Untaian simbol ditulis secara berurutan dari satu ke yang lain setelahnya. Satu untaian yang tidak terlihat, disebut untaian kosong (empty string), biasanya dilambangkan dengan "". Beberapa untaian menjadi bagian dari suatu bahasa, beberapa yang lain tidak. Untaian yang menjadi bagian dari suatu bahasa disebut kata atau kalimat.

Dalam ilmu komputer, bahasa manusia disebut bahasa alamiah (natural languages), dimana komputer tidak bisa memahaminya, sehingga diperlukan suatu bahasa komputer.

Bahasa yang dapat dimengerti oleh komputer disebut bahasa pemrograman. Bahasa pemrograman mempengaruhi cara dan teknik pemrograman.

Komputer mengerjakan transformasi data berdasarkan kumpulan perintah - program - yang telah dibuat oleh pemrogram. Kumpulan perintah ini harus dimengerti oleh komputer, berstruktur tertentu (sintaks) dan bermakna. Bahasa pemrograman merupakan notasi untuk memberikan secara tepat program komputer. Berbeda dengan bahasa alamiah, mis. Bahasa Indonesia, Inggris dsb. yang merupakan bahasa alamiah (*natural language*), sintaks dan semantik bahasa pemrograman (komputer) ditentukan secara kaku, sehingga bahasa pemrograman juga disebut sebagai bahasa formal (*formal language*). Jadi, dalam bahasa pemrograman yang digunakan sebagai alat komunikasi untuk memberikan perintah kepada komputer tidak berlaku kebebasan berekspresi seperti laiknya dalam bahasa alamiah.

Pemrograman dalam pengertian luas meliputi seluruh kegiatan yang tercakup dalam pembuatan program, termasuk analisis kebutuhan (*requirement's analysis*) dan keseluruhan tahapan dalam perencanaan (*planning*) , perancangan (*design*) dan pewujudannya (*implementation*). Dalam pengertian yang lebih sempit, pemrograman merupakan pengkodean (*coding* atau *program writing* = penulisan program) dan pengujiannya (*testing*) berdasarkan rancangan tertentu. Pemahaman yang lebih sempit ini sering digunakan dalam pembuatan program-program terapan komersial yang membedakan antara *system analyst* yang bertanggung jawab dalam menganalisa kebutuhan, perencanaan dan perancangan program dengan pemrogram (*programmer*) yang bertugas membuat kode program dan menguji kebenaran program.

Secara umum terdapat 4 kelompok Bahasa Pemrograman, yaitu :

1. Object Oriented Language (Visual dBase, Visual FoxPro, Delphi, Visual C)
2. High Level Language (seperti Pascal dan Basic)
3. Middle Level Language (seperti bahasa C), dan
4. Low Level Language (seperti bahasa Assembly)

Tipe Pemrograman ada 7 macam, yaitu :

1. **Pemrograman Prosedural**
   Algoritma berisi urutan langkah-langkah penyelesaian masalah. Ini berarti algoritma adalah proses yang prosedural.

   Definisi prosedural adalah :
   a. Tahap-tahap kegiatan untuk menyelesaikan suatu aktivitas
   b. Metode langkah demi langkah secara eksak dalam memecahkan suatu masalah.

Bahasa tingkat tinggi seperti Cobol, Basic, Pascal, Fortran dan C mendukung kegiatan pemrograman prosedural, karena itu mereka dinamakan juga bahasa prosedural.

2. Pemrograman Terstruktur

Pemrograman terstruktur adalah bahasa pemrograman yang mendukung pembuatan program sebagai kumpulan prosedur. Prosedur-prosedur ini dapat saling memanggil dan dipanggil dari manapun dalam program dan dapat mengunakan parameter yang berbeda-beda untuk setiap pemanggilan. Bahasa pemrograman terstruktur adalah pemrograman yang mendukung abstraksi data, pengkodean terstruktur dan kontrol program terstruktur.
Contoh bahasa pemrograman terstruktur : Pascal, Cobol, RPG, ADA, C.

3. Pemrograman Modular

Dalam pemrograman modular, program dipecah-pecah ke dalam modul-modul, dimana setiap modul menunjukkan fungsi dan tugas tunggal. Dengan membagi masalah ke dalam modul-modul, maka masalah akan menjadi sederhana sehingga program dapat lebih mudah disusun dan dipahami.
Pemrograman modular diterapkan dengan menggunakan sub-routine, yaitu sebuah kumpulan perintah yang melakukan tugas pemrosesan yang terbatas. Pemrograman ini banyak dimanfaatkan oleh Bahasa Pemrograman Berbasis Obyek.

4. Pemrograman Fungsional

Disebut bahasa pemrograman fungsional karena memang pada program seluruh kodenya berupa fungsi-fungsi. Bahasa pemrograman fungsional merupakan salah satu bahasa pemrograman yang memperlakukan proses komputasi sebagai evaluasi fungsi-fungsi matematika.
Contoh : Lisp, Scheme, ML, Haskell.

5. Pemrograman Berorientasi Obyek

Obyek : elemen yang memiliki fungsi, metode, karakteristik tertentu yang dapat dibedakan dalam dunia nyata.
Class : kumpulan obyek-obyek yang memiliki kesamaan karakteristik.

- Merupakan bahasa pemrograman yang mampu memanfaatkan obyek-obyek yang tersedia atau membuat suatu obyek tertentu dengan menggunakan bahasa pemrograman.
- Mampu merefleksikan kebutuhan-kebutuhan user sebagaimana layaknya yang ada di dunia nyata
- Relatif lebih fleksibel dan mudah diadaptasikan terhadap perubahan suatu program
- Memiliki feature yang memperkuat dan meningkatkan fleksibilitas suatu obyek dengan adanya class, instance, encapsulation, inheritance, reusability, dan polymorphism.

Contoh : C++, SmallTalks, Java.

6. Pemrograman Visual

- Penggunaan ekspresi visual (seperti grafik, gambar, atau ikon) dalam proses pemrograman
- Mengacu pada aktivitas yang memungkinkan pengguna untuk membuat program dalam dua (atau lebih) dimensi

7. Pemrograman Even-Driven

Menggunakan konsep "jika sebuah aksi/perintah dilakukan terhadap sebuah obyek, apa yang akan terjadi/dilakukan oleh obyek tersebut selanjutnya".

Sangat fleksibel dalam pembuatan koding program, karena sudah mengunakan konsep OOP dimana pemrograman dapat dimulai dari obyek yang diinginkan tanpa harus terurut. Biasanya meruapakan jenis bahasa pemrograman visual.
Contoh : Visual Basic, Visual C++, Delphi.

I.2. Kriteria Bahasa Pemrograman

Seorang *programmer* akan memilih bahasa pemrogramannya jika merasa bahwa bahasa pemrograman tersebut bagus dan mudah untuk digunakan. Ada beberapa kriteria untuk penilaian suatu bahasa pemrograman, yaitu :

a. Clarity, simplicity dan unity

Bahasa pemrograman harus dapat menolong *programmer* untuk membuat suatu desain program jauh sebelum programmer melakukan *coding*. Kemudahan, kesederhanaan, dan kesatuan merupakan suatu kombinasi yang membantu *programmer* mengembangkan suatu algoritma sehingga algoritma yang dihasilkan mempunyai kompleksitas yang rendah.

Sintaks bahasa pemrograman mempengaruhi kemudahan ketika program mulai ditulis, dites, dan dimodifikasi. Program yang mudah dibaca adalah kunci dari hal tersebut.

b. Orthogonality

Orthogonality menunjuk kepada suatu atribut yang dapat dikombinasikan dengan beragam fitur bahasa pemrograman sehingga setiap kombinasinya mempunyai arti dan dapat digunakan. Contohnya, suatu bahasa pemrograman mendukung suatu ekspresi yang dapat menghasilkan suatu nilai, dan bahasa pemrograman tersebut juga mendukung statemen kondisi yang mengevaluasi suatu ekspresi untuk mendapatkan nilai **`true`** dan **`false`**. Dua fitur dari bahasa pemrograman tersebut, yaitu ekspresi dan statemen kondisi, adalah orthogonal jika sembarang ekspresi dapat digunakan (dan dievaluasi) di dalam statemen kondisi.

Ketika fitur bahasa pemrograman adalah orthogonal, maka bahasa pemrograman tersebut akan mudah dipelajari dan program akan mudah ditulis karena hanya ada sedikit *exception* dan *case* yang harus diingat.

c. Kewajaran untuk aplikasi

Bahasa pemrograman membutuhkan sintaks yang tepat/cocok yang digunakan pada struktur program untuk merefleksikan struktur logika yang melandasi suatu algoritma.

Bahasa pemrograman harus mempunyai struktur data, operasi-operasi, struktur kontrol, dan sintaks alami yang tepat/cocok untuk memecahkan suatu masalah. Suatu bahasa pemrograman didesain secara khusus untuk kebutuhan tertentu, misalnya PROLOG digunakan untuk keperluan deduksi atau C++ untuk pemrograman berorientasi objek.

d. Medukung abstraksi

Abstraksi merupakan suatu hal yang substansial bagi *programmer* untuk membuat suatu solusi dari masalah yang dihadapi, sehingga abstraksi tersebut dapat dengan mudah diimplementasikan menggunakan fitur-fitur yang ada dalam bahasa pemrograman.

e. Kemudahan untuk verifikasi program

Verifikasi program merupakan hal penting bagi sebuah program karena dengan verifikasi yang mudah maka suatu program akan dengan mudah dibangun dan

dikembangkan. Kesederhaan struktur semantik dan sintaks merupakan aspek primer yang mempengaruhi kesederhanaan verifikasi program.

f. Lingkungan pemrograman

Bahasa pemrograman yang mempunyai lingkungan pemrograman yang baik dan lengkap akan memudahkan *programmer* untuk mengimplementasikan abstraksi yang sudah disusunnya. Lingkungan pemrograman di sini dapat berarti editor yang digunakan, dokumentasi yang baik dari bahasa pemrograman, fasilitas *debugging*, *user interface* yang baik, ataupun *tool* lain yang dapat digunakan untuk memudahkan pekerjaan *programmer*.

SmallTalk merupakan salah satu bahasa pemrograman yang didesain khusus untuk lingkungan pemrogramannya, terdiri dari Windows, menu, input mouse, dan sekumpulan tool untuk digunakan dalam program.

g. Portabilitas program

Salah satu kriteria penting untuk proyek pemrograman adalah kemudahan program yang sudah jadi untuk dipindah-pindahkan dari komputer yang digunakan untuk membuat dan mengembangkan, ke komputer lain yang akan menggunakannya.

h. Biaya penggunaan

Biaya merupakan elemen penting dalam mengevaluasi suatu bahasa pemrograman. Ada beberapa biaya yang dapat diukur, yaitu:

- Biaya eksekusi program

  Program yang sering dieksekusi akan membutuhkan suatu kode executable yang efisien sehingga cepat untuk dieksekusi. Semakin cepat suatu program dieksekusi maka akan semakin murah biaya eksekusi program.

- Biaya translasi/kompilasi program

  Untuk pembelajaran, kecepatan translasi lebih diutamakan daripada kecepatan eksekusi karena pada pembelajaran lebih sering dilakukan translasi/kompilasi daripada eksekusi program yang dihasilkan. Oleh karena itu, lebih dibutuhkan compiler yang efisien dibandingkan kode executable yang efisien.

- Biaya penciptaan, testing dan penggunaan program

  Semakin baik dan lengkap lingkunga pemrograman pada bahasa pemrograman maka ketiga biaya ini akan menjadi rendah. Hal ini disebabkan tidak banyak waktu dan tenaga serta pikiran yang dicurahkan ke pembuatan program.

- Biaya pemeliharaan program

  Pemeliharaan program termasuk perbaikan error yang muncul ketika program sudah digunakan, perubahan yang dibutuhkan pada program ketika hardware atau sistem operasi berubah, dan penyesuaian kebutuhan dengan kebutuhan yang baru.

  Pemeliharaan merupakan salah satu biaya terbesar dari life cycle cost dan merupakan suatu hal yang membosankan bagi programmer.

I.3. Data

I.3.1. Objek data

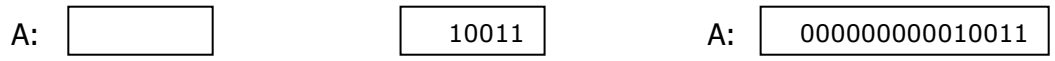
Pengelompokan satu atau lebih potongan data di dalam virtual komputer pada saat *run-time* disebut objek data. Selama eksekusi suatu program, banyak

objek data berbeda dengan tipe yang berbeda-beda muncul sehingga objek data ini bersifat dynamic.

Beberapa objek data yang muncul selama program dieksekusi merupakan objek data buatan *programmer* dimana *programmer* secara eksplisit membuat dan memanipulasi melalui deklarasi dan statemen di dalam program, contohnya variabel, konstanta, array, file, dan seterusnya. Objek data yang lainnya merupakan objek data yang dibuat oleh sistem. Contohnya adalah objek data yang ada di virtual komputer untuk ”perawatan” selama program dieksekusi dan tidak dapat diakses oleh programmer secara langsung, *misalnya file buffer*, *list* ruang kosong, *stack* penyimpanan pada waktu *run-time*, dan lainnya yang berhubungan dengan sistem.

Objek data menggambarkan suatu *container* untuk nilai data, suatu tempat dimana nilai-nilai data disimpan dan dipanggil kembali. Objek data dikarakteristikkan oleh suatu kumpulan *attribute*. *Attribute* menentukan jumlah dan tipe nilai objek data. Nilai data dapat berupa angka tunggal, karakter, atau ada kemungkinan *pointer* ke objek data yang lain.

Perbedaan objek data dengan nilai data dapat membingungkan. Perbedaan yang mudah dilihat adalah suatu objek data selalu direpresentasikan sebagai penyimpanan di memori komputer. Suatu nilai data direpresentasikan sebagai pola dari bit. Suatu objek data A berisi nilai B, yang berarti bahwa suatu blok penyimpanan yang direpresentasikan oleh A diset berisi pola bit khusus yang merepresentasikan B, seperti pada gambar 3.2.



(a) Objek data: suatu lokasi di memori komputer dengan nama A

(b) Nilai data: pola bit yang digunakan oleh translator sewaktu-waktu angka 19 digunakan di program

(c) variabel terikat: objek data terikat ke nilai data 19

Gambar 3.2. Objek data sederhana dengan nilai 19

I.3.2. Tipe Data

Tipe data merupakan suatu kelas objek data dengan kumpulan operasi untuk membentuk dan memanipulasinya. Setiap bahasa pemrograman mempunyai kumpulan tipe data primitif yang sudah *built-in*. Sebagai tambahan, bahasa pemrograman mempunyai fasilitas supaya *programmer* dapat mendefinisikan tipe data baru. Salah satu perbedaan pokok antara bahasa pemrograman lama (FORTRAN, COBOL) dengan bahasa pemrograman baru (C, C++, ADA) adalah pembuatan tipe data baru (*programmer-defined data types*).

Objek data dasar terdiri dari nilai data tunggal. Suatu kelas dari objek data tersebut, dimana banyak operasi yang terlibat didalamnya, disebut tipe dasar. Contohnya adalah integer, real, character, boolean, enumerasi dan pointer, walaupun setiap bahasa pemrograman agak berbeda dalam pengelompokkannya.

Elemen dasar suatu tipe data mempunyai syarat-syarat sebagai berikut:

a. Attribute

   Membedakan objek-objek data dari tipe tersebut, misalnya nama objek data.

b. Nilai

Dimiliki oleh obejk data dari tipe tersebut. Nilai ini dipengaruhi oleh *hardware* komputer yang melandasinya. Contohnya dalam bahasa C, dikenal kelas tipe data integer **`short`**, **`long`** dan **`int`**.

c. Operasi

Mendefinisikan manipulasi-manipulasi yang dimungkinkan oleh objek data tipe tersebut. Operasi ini biasanya merupakan operasi secara matematika. Suatu operasi yang mempunyai dua argumen dan menghasilkan satu hasil disebut operasi *binary*. Jika hanya ada satu argumen dengan satu hasil maka disebut operasi *unary*. Jumlah argumen untuk suatu operasi disebut *arity*.

I.3.2.1. Tipe data dasar

Merupakan tipe data primitif yang tidak terstruktur yang didefinisikan oleh bahasa pemrograman. Tipe data dasar dibagi menjadi lima bagian besar yang terdiri dari bagian-bagian kecil, yaitu tipe data numerik, enumerasi, boolean, character, dan internatinalization.

a. Tipe data Numerik

Di setiap bahasa pemrograman, dapat dipastikan ada tipe data numerik. Paling umum adalah integer dan riil karena keduanya didukung secara langsung oleh *hardware* komputer.

- Integer

  Objek data integer tidak mempunyai *attribute* yang lain selain dirinya sendiri. Kumpulan nilai integer dalam suatu bahasa pemrograman merupakan bilangan terbatas dan merupakan himpunan bagian dari suatu integer yang tidak terbatas dalam perhitungan matematika.

- Subrange

  Suatu subrange dari tipe data integer merupakan sebuah *subtype* dari tipe data integer dalam range yang terbatas. Contohnya adalah integer dalam range 1 sampai 10. Subrange dapat juga disebut *subtype* dari basis tipe integer.

  Keuntungan menggunakan subrange:

  - Penyimpanan yang lebih kecil.
  - Pemeriksaan tipe data yang lebih baik.

- Floating-point real

  Tipe data floating-point real biasa disingkat dan disebut tipe data riil di FORTRAN dan Pascal, atau float di C dan C++.

- Fixed-point real

  Bilangan fixed-point dipresentasikan dengan urutan digit yang mempunyai panjang tetap, dengan titik desimal diposisikan di tempat yang diberikan antara dua digit.

- Lain-lain

  - Bilangan kompleks
  - Bilangan rational

b. Enumerasi

Enumerasi adalah suatu urutan list dari nilai-nilai yang berbeda. Programmer mendefinisikan nama literal yang akan digunakan sebagai nilai dan urutannya.

Enumerasi digunakan ketika suatu variabel hanya membutuhkan nilai-nilai tertentu dan terbatas, contohnya adalah variabel **`KelasMhs`** yang mempunyai tiga nilai yaitu ”Baru”, ”Yunior”, ”Senior” dan variabel **`JenisKel`** yang mempunyai dua nilai yaitu ”Laki-laki” dan ”Perempuan”.

c. Boolean

Tipe data yang merepresentasikan TRUE atau FALSE.

d. Character

Sebagian besar data yang di-input ke dalam komputer merupakan bentuk character. Oleh karena itu, bahasa pemrograman menyediakan fasilitas konversi character ke tipe data lain selama proses input dan output berlangsung, tetapi dalam pemrosesan data secara langsung dalam bentuk character tetap diperlukan.

e. Internationalization.

Penggunaan character ASCII yang 8-bit kurang mencukupi sehingga berkembanglah internationalization yang disebut I18N.

I.3.2.2. Tipe data terstruktur

a. Vector dan Array

Vector adalah struktur data yang tersusun dari komponen-komponen bertipe sama dengan jumlah tetap dan terbatas sebagai suatu urutan linier yang sederhana. Komponen suatu *vector* dipilih dengan pemberian integer *subscript* (nilai enumerasi) yang menandakan posisi komponen tersebut di dalam urutan. Vector disebut juga *array* satu dimensi atau *array linier*.

b. Record

Merupakan suatu struktur data yang tersusun dari komponen-komponen berbeda tipe dengan jumlah tetap dan terbatas sebagai suatu urutan linier yang sederhana.

Record berbeda dengan vector dalam dua hal, yaitu:

- Komponen record adalah *heterogeneous*, yang merupakan campuran banyak tipe.
- Komponen record dinamai dengan *symbolic name* (identifier), bukan *subscript* yang terurut.

c. List

Merupakan struktur data yang mirip dengan *vector* dan tersusun atas urutan yang terurut dari suatu struktur data. Anggota pertama dari list disebut *head*, anggota kedua dan seterusnya disebut *tail*.

List mempunyai perbedaan dengan *vector* dalam beberapa hal, yaitu:

- *List* jarang yang mempunyai panjang yang tetap (*fixed-length*)
- List jarang yang *homogeneous*. Tipe data setiap member dapat berbeda.
- Bahasa yang menggunakan *list* biasanya mendeklarasikan data secara implisit tanpa *attribute* yang eksplisit untuk member list.

d. Character String

Merupakan objek data yang tersusun atas urutan *character*. *Character string* merupakan tipe data yang penting di banyak bahasa pemrograman karena kerap digunakan untuk input dan output.

e. Pointer dan Objek Data Programmer-Constructed

Tipe data *pointer* mendefinisikan suatu kelas dari objek data yang mempunyai nilai di lokasi objek data yang lain.

f. Himpunan

Merupakan suatu objek data yang tersusun dari suatu kumpulan yang tidak terurut dari nilai-nilai yang berbeda.

I.3.2.3. Tipe data abstrak

Merupakan suatu fasilitas dari bahasa pemrograman yang menampung aspirasi para programmer supaya dapat mempunyai tipe data yang didefinisikan sendiri. Fasilitas ADT (*Abstract Data Type*) pertama kali muncul di Simula 67. Implementasinya disebut *class*. ADT adalah model matematika dengan sekumpulan operasi-operasi yang absah yang didefinisikan pada model ini. ADT merupakan generalisasi dari tipe data primitif (integer, real, float, dsb), sementara operasi-operasi yang berada di ADT yang berupa prosedur dan fungsi merupakan generalisasi operasi-operasi primitif (seperti +, -, dsb).

Model Komputasi

Ada tiga model dasar komputasional-- fungsional, logika, dan imperatif. Sebagai tambahan terhadap satuan nilai-nilai dan operasi yang berhubungan, masing-masing model komputasional mempunyai satu set operasi yang digunakan untuk menggambarkan komputasi.

a. *Model Fungsional* : terdiri dari satu set nilai-nilai, fungsi-fungsi dan operasi aplikasi fungsi dan komposisi fungsi. Fungsi dapat mengambil fungsi lain sebagai argumentasi dan mengembalikan fungsi sebagai hasil (higher-order function). Suatu program adalah koleksi definisi fungsi-fungsi dan suatu komputasi adalah aplikasi fungsi.

b. *Model Logika* : terdiri dari satu set nilai-nilai, definisi hubungan dan kesimpulan logis. Program terdiri dari definisi hubungan dan suatu komputasi adalah suatu bukti(suatu urutan kesimpulan).

c. *Model Imperatif* : terdiri dari satu set nilai-nilai yang mencakup suatu keadaan dan operasi tugas untuk memodifikasi pernyataan. Pernyataan adalah set pasangan nilai-nama dari konstanta dan variabel. Program terdiri dari urutan tugas dan suatu komputasi terdiri dari urutan pernyataan.

Definisi Sintaks, Sematik, dan Pragmatis

Sintaks : aturan gramatikal atau komposisi suatu program yang mengatur tata cara penulisan huruf, angka dan karakter lain.
Contoh : pada pembuatan program Pascal antara dua statement dipisahkan oleh titik koma (;).

```
X := 1 ; X := X + 1;
```

Semantik : mendefinisikan arti dari dari program yang benar secara sintaks dari bahasa pemrograman tersebut.
Contoh : Pada pembuatan program C

Int vector [10]

Arti semantiknya : akan menyebabkan ruang sebanyak 10 elemen integer diberikan kepada variabel bernama vector (0 – 9 untuk array dalam C)

Pragmatis : memperhatikan tentang pemakaian bahasa, area aplikasi, kemudahan implementasi dan penggunaan, dan sukses bahasa didalam desain pelaksanaan tujuannya. Kekuatan yang membentuk suatu bahasa pemrograman meliputi arsitektur komputer, praktek rancang-bangun perangkat lunak (terutama daur hidup perangkat lunak), model komputasional, dan daerah aplikasi (contoh: alat penghubung pemakai, sistem pemprograman, dan sistem ahli).

Tujuan umum bahasa pemrograman berpegang pada prinsip desain bahasa pemrograman yang berikut.

Prinsip Kelengkapan Komputasional
Model komputasional untuk tujuan umum suatu bahasa pemrograman harus universal.

Prinsip Implementasi
Implementasi harus efisien dalam penggunaan waktu dan ruangnya. Prinsip Memprogram harus ditulis dalam suatu bahasa yang mencerminkan daerah masalah.