Universitas Esa Unggul Kampus Harapan Indah	**MODUL I-II** **Struktur Data**	
Judul	**TYPE DATA DAN TYPE DATA ABSTRAK**	
Penyusun	**Distribusi**	**Perkuliahan**
Nixon Erzed	Teknik Informatika Universitas Esa Unggul	Pertemuan – II online

Tujuan :

1. Mahasiswa memahami pengertian dan posisi struktur data dalam pemrograman
2. Mahasiswa mengenal dan memahami tipe data dan urgensi representasi data dalam pemrograman
3. Mahasiswa memahami representasi data dan dapat mendeklarasikan struktur data secara lengkap

Materi:

1. Pemrograman Komputer
2. Input output
3. Logika proses dan algoritma
4. Data dan deklarasi struktur data
5. Penyelesaian masalah dengan computer
6. Konstanta dan variabel
7. Definisi Formal Struktur Data
8. Pembuatan Struktur Data
9. Spesifikasi Non Formal
10. Pengertian Deklarasi
11. Deklarasi Type dan Function
12. Deklarasi Struktur Data

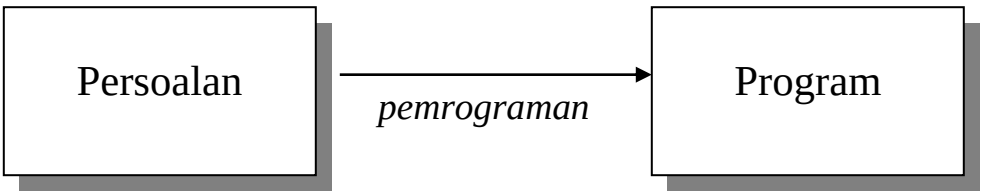
Referensi :

1. Algoritma dan Pemrograman dalam Bahasa Pascal, C, dan C++, Rinaldi Munir 2016
2. Struktur Data, Bambang Hariyanto, Infromatika, Bandung, 2003
3. Algoritma dan Struktur Data 1 & 2 , Moh. Syukani, Mitra Wacana Media, 2012
4. Fundamental of Data Structure, Ellis Horowitz, Pitman International Text, 1978

DASAR-DASAR STRUKTUR DATA

I. PEMROGRAMAN KOMPUTER

Komputer pada dasarnya adalah mesin yang tidak bisa apa-apa. Harus diberikan serangkaian instruksi kepada komputer agar dapat memecahkan suatu masalah. Langkah-langkah yang dilakukan untuk memberikan instruksi kepada komputer untuk memecahkan masalah dinamakan *pemrograman komputer*.



Hal-hal yang berhubungan dengan pemrograman komputer :

1. Bahasa pemrograman
2. Source program (program sumber)
3. Interpreter atau compiler
4. File program (execute file)

Bahasa Pemrograman

Aturan penulisan instruksi komputer dalam bahasa yang dimengerti oleh manusia.

Kelompok bahasa pemrograman : Bahasa Tingkat Rendah, Bahasa Generasi III, dan Bahasa Generasi IV

Umumnya bahasa pemrograman menggunakan kata atau singkatan kata dalam bahasa Inggris.

Bahasa Tingkat Rendah atau assembly adalah spesifik mikroprosesor, yang menggunakan singkatan-singkatan dalam bahasa Inggris, tapi dengan sintaks yang sangat kaku karena merupakan pemetaan satu-satu dari instruksi bahasa mesin.

Bahasa Generasi III (3GL / bahasa tingkat tinggi) menggunakan sintaks yang lebih dekat dengan bahasa manusia, sehingga lebih mudah digunakan.

Termasuk 3GL : Basic, Pascal, C, C++, Cobol, dan sebagainya.

Bahasa Generasi IV (4GL) adalah bahasa yang memiliki kemudahan untuk mengembangkan aplikasi basis data. Salah satu contohnya adalah SQL (Structured Query Language) yang banyak diadopsi oleh Sistem Manajemen Basis Data dan *aplikasi builder*nya.

Program Sumber (*Source Program)*

Program sumber adalah kumpulan instruksi yang ditulis menggunakan dan mengikuti aturan bahasa pemrograman untuk menyelesaikan suatu persoalan tertentu.

Assembler, Interpreter dan Compiler

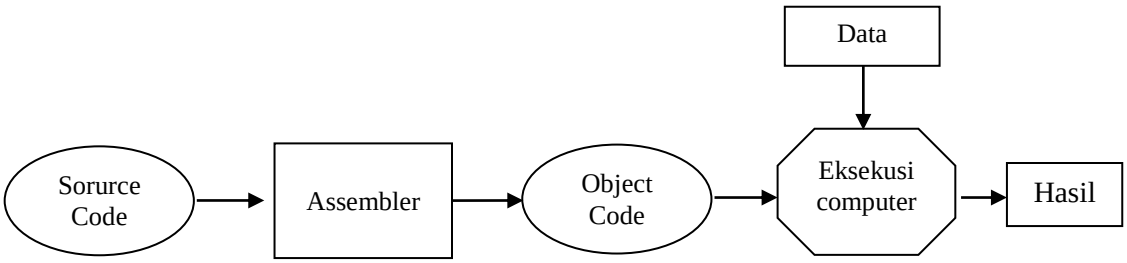
Assembler, interpreter dan compiler adalah perangkat lunak untuk menterjemahkan program sumber ke bahasa mesin.

Assembler adalah perangkat lunak yang mengkonversikan perintah-perintah assembly ke bahasa mesin.

Interpreter menterjemahkan program baris per baris ke bahasa mesin sebelum dieksekusi, artinya interpretasi dilakukan ketika program akan dieksekusi dan hasil interpretasi tidak disimpan secara permanen dalam memory sekunder.

Compiler menterjemahkan semua perintah program sumber kedalam bahasa mesin, hasil kompilasi disimpan secara permanen dalam memory sekunder.

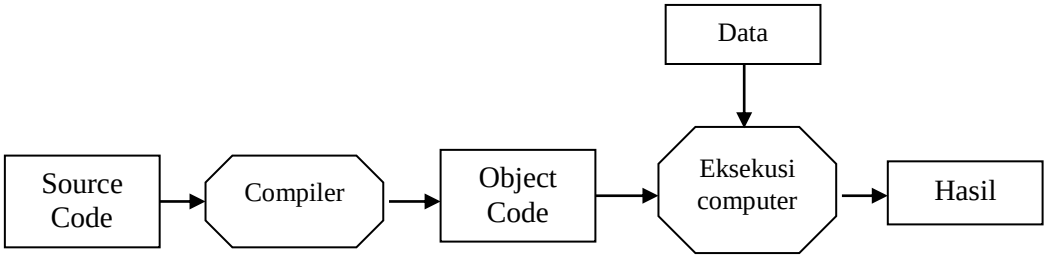
a. Struktur Sistem Assembler :



Source code adalah bahasa assembly, object code adalah bahasa mesin.

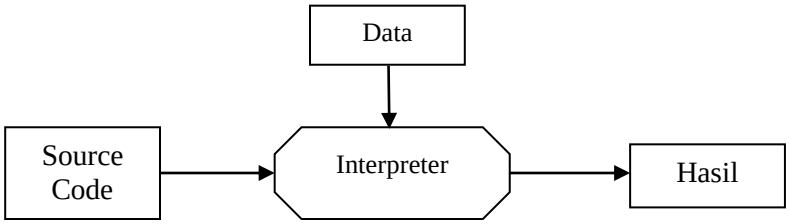
Contohnya : Turbo Assembler dan Macro Assembler

b. Struktur Sistem Kompilator :



Source code adalah bahasa tingkat tinggi, object kode adalah bahasa mesin atau bahasa assembler. Source code dan data diproses pada saat yang berbeda

c. Struktur Sistem Interpreter :



Interpreter tidak membangkitkan object code, hasil translasi hanya dalam bentuk internal.

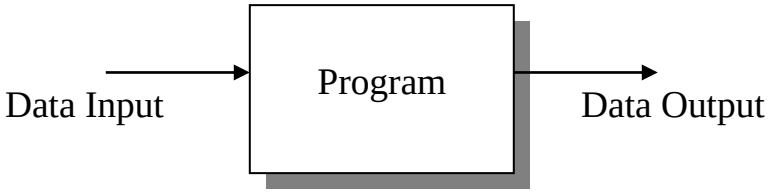
Execute File/Program

Execute file atau *execute program* adalah file hasil kompilasi yang siap untuk dijalankan oleh mesin (komputer).

II. INPUT OUTPUT

Program komputer adalah sebuah sistem. Sesuai dengan konsep sistem program akan bekerja jika mendapat masukan tertentu. Hasil kerjanya diidentifikasi berdasarkan output yang dihasilkannya.

Input dan output dalam program komputer berbentuk data.



Dalam sistem komputer, data input akan disimpan dalam suatu variabel yang merepresentasikan data tersebut dalam proses-proses yang dilakukan program.

Data input dapat diterima melalui masukan langsung dengan keyboard atau perangkat masukan lainnya, atau masukan yang berasal dari file data yang tersimpan dalam memory sekunder.

Hasil pengolahan oleh program dapat dikeluarkan langsung ke perangkat keluaran atau disimpan/dituliskan ke file data untuk disimpan dalam memory sekunder.

Pengorganisasian data yang merepresentasikan persoalan dalam program dikenal sebagai struktur data.

III. LOGIKA PROSES DAN ALGORITMA

Hasil analisis terhadap persoalan akan memberikan ***metoda-tepat*** bagaimana masalah akan diselesaikan dengan sistem komputer, yang merupakan logika proses/penyelesaian masalah.

Logika proses tersebut akan dituangkan dalam sekumpulan langkah-langkah terbatas. Setiap langkah mungkin memerlukan satu oprasi atau lebih.

Ciri-ciri algoritma :

1. Input
2. Output
3. Definite
4. Effective
5. Terminate

Input : Terdapat nol atau lebih masukan yang diberikan secara eksternal

Output : Sedikitnya terdapat satu keluaran yang harus dihasilkan

Definite : Harus secara sempurna menyatakan apa yang akan dikerjakan

Contoh :

a. Hitung 5/0
b. Tambahkan 6 atau 7 ke x

Contoh instruksi tersebut tidak dijinkan karena :

Kasus a : tidak jelas hasil operasinya.
Kasus b : tidak jelas yang harus dilakukan.

Effective : Setiap instruksi harus dapat dilakukan secara manual menggunakan pensil dan kertas dalam jumlah waktu yang berhingga.

Terminate : Harus berhenti setelah sejumlah terbatas operasi.

IV. DATA DAN DEKLARASI STRUKTUR DATA

Penganalisisan persoalan untuk dapat diselesaikan dengan komputer diwujudkan melalui penelitian terhadap data-data yang mewakili persoalan tersebut. Pemrosesan oleh komputer hanya dapat dilakukan terhadap data kuantitatif. Jika dalam analisis persoalan terdapat data kualitatif, maka data tersebut mesti dikuantifisir, artinya harus didapatkan padanan angka eksak dari setiap nilai kualitatif.

Contoh :

Persoalan posisi untung rugi warung makan. Analisis terhadap persoalan diketahui bahwa untung rugi usaha dapat dihitung berdasarkan data nilai penjualan, biaya bahan baku dan biaya operasional.

Setiap data yang akan diolah oleh program, memerlukan variabel sebagai wadah baik sebagai penampung data awal maupun untuk menyimpan data hasil proses. Untuk memberikan identitas dan batasan dalam algoritma/program, setiap variabel yang terlibat dalam proses harus didefinisikan/dideklarasikan diawal algoritma atau program.

Deklarasi data meliputi :

- penamaan variabel
- penetapan type variabel
- panjang/ukuran data

Untuk contoh diatas misalnya dideklarasikan sebagai berikut :

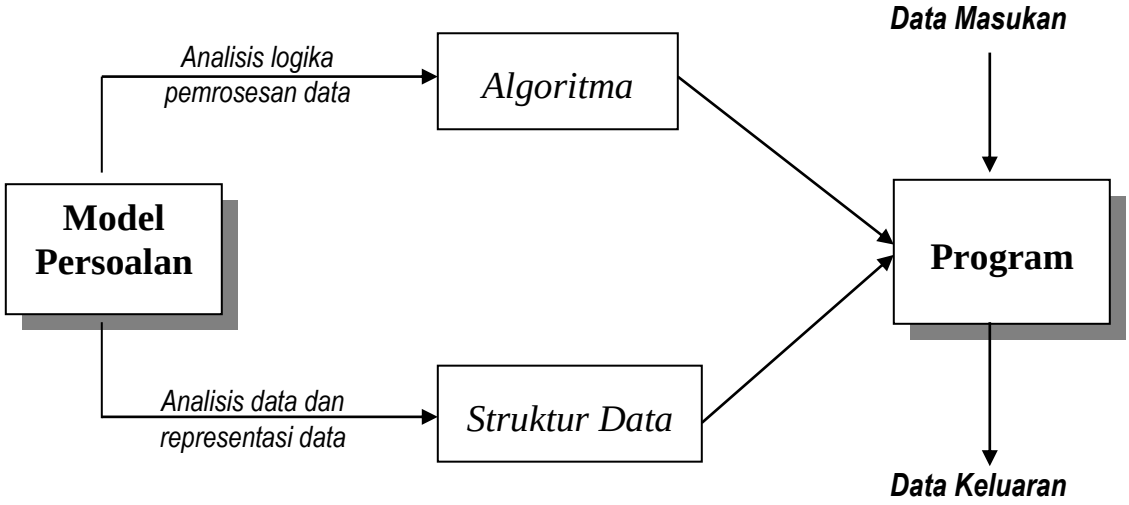
Deklarasi

N_jual	*variabel larik bertype real dengan panjang 100 data*
B_bahan	*variabel larik bertype real dengan panjang 100 data*
B_Opers	*variabel larik bertype real dengan panjang 100 data*

Aturan pendeklarasian struktur data tergantung pada aturan tata bahasa pemrograman. Contoh diatas adalah model untuk bahasa algoritma.

V. PENYELESAIAN MASALAH DENGAN KOMPUTER

Model penyelesaian persoalan dengan komputer :



Pada penyelesaian persoalan yang besar dan kompleks, akan sangat membantu jika persoalan dipecah menjadi sub-persoalan *(clustering dan decoupling)*. Dalam melakukan analisis logika pemrosesan data, clustering/decoupling akan menghasilkan modulisasi program, sedangkan pada analisis data dan representasi data akan menghasilkan sub-sub struktur data. Untuk penyelesaian persoalan komputasi bisnis, sub-sub struktur data tersebut tercermin dalam rancangan basis data dan sistem file.

Beberapa hal yang harus diperhatikan dalam penyelesaian persoalan dengan komputer:

1. Pemahaman secara menyeluruh keterhubungan elemen-elemen data yang relevan terhadap solusi persoalan
2. Operasi-operasi yang akan dilakukan terhadap elemen data
3. Perancangan metoda representasi elemen-elemen data di memori sehingga memenuhi kriteria berikut :
   a. Memenuhi keterhubungan logika antara elemen-elemen data
   b. Operasi-operasi terhadap elemen data dapat dilakukan dengan mudah dan efisien
4. Memilih bahasa pemrograman yang paling sesuai untuk menterjemahkan solusi persoalan menjadi program

TYPE DATA & REPRESENTASI DATA

I. KONSTANTA DAN VARIABEL

Konstanta :

Konstanta (literal) adalah suatu nilai yang tetap berada di dalam program. Jika dalam membuat suatu program sering menggunakan bilangan numerik atau suatu kalimat string sama berkali-kali, akan lebih baik jika bilangan atau string tersebut dideklarasikan sebagai *konstanta.*

Manfaatnya : memudahkan dalam pemahaman dan perbaikan program.
Misalnya untuk sebuah bilangan yang kompleks, akan lebih baik jika diberi sebuah nama yang mudah. Jika ingin dilakukan perubahan terhadap konstanta, maka hanya diperlukan penggantian satu kali saja.

Contoh *(dalam sintaks Pascal) :*

```
Const    Pi  =  3.1415926536
```

Didalam blok program jika akan digunakan, cukup ditulis *Pi.*
Jika ingin diganti nilai *Pi* menjadi 3.14 maka cukup diganti pada deklarasi konstanta.

Deklarasi Konstanta :
Sesuai aturan penulisan bahasa pemrograman. Pada Pascal, dideklarasikan pada awal program sebelum blok (*begin-end*)

Jenis jenis konstanta :

a. Konstanta biasa :
mendeklarasikan suatu nilai numerik, character, atau string

b. Konstanta bertype :
selain nilai, pada konstanta juga dapat didefinisikan type tertentu

Contoh (dalam sintaks Pascal) :

```
Pi  = 3.14;                                  (konstanta biasa)
Indeks = 'A';                                (konstanta biasa)
Nama  = 'Esa Unggul'";                       (konstanta biasa)
JumlahMahasiswa : integer = 15000;           (konstanta bertype)
CekNilai  :  boolean=true;                   (konstanta bertype)
JdlHal : string[25]='Pendapatan Tahunan';    (konstanta bertype)
```

VARIABEL

Variabel adalah suatu lokasi dimemori yang disiapkan oleh programmer dan diberi nama khas untuk menampung suatu nilai dan atau mengambil kembali nilai tersebut.

Pendeklarasian variabel merupakan bagian langkah pembangunan struktur data. Penamaan variabel yang baik merepresentasikan data yang diwadahinya. Setiap bahasa pemrograman memiliki aturan penamaan.

Contoh :

JmlMahasiswa : untuk merepresentasikan data jumlah mahasiswa
Luas : untuk merepresentasikan data luas
NilaiUTS [i] : untuk merepresentasikan data nilai mahasiswa

Deklarasi variabel mencakup :
pemberian nama identifier
pedefinisian type
pendefinisian jangkauan (khususnya untuk variabel larik)

Contoh :

Pascal

```
JmlPesertaMK    : integer;
NilaiUTS        : array [ 1..  100 ]  of  real;
```

Dimana :

JmlPesertaMK dan *NilaiUTS* adalah nama variabel

Integer dan *array..of real* adalah type variabel

[1 .. 100] adalah jangkauan

C

```
int   jmlpeserta
float   nilaiUTS [20]
```

Dimana :

jmlPeserta dan *nilaiUTS* adalah nama variabel
Int dan *float* adalah type variabel
[20] adalah jangkauan

II. STRUKTUR DATA

Type Data

Type data adalah jenis data yang ditangani oleh bahasa pemrograman. Tiap bahasa pemrograman memiliki kumpulan type data *built-in,* sehingga :

- Memungkinkan deklarasi variabel bertype tersebut.
- Menyediakan kumpulan operasi untuk manipulasi variabel bertype tersebut.

Beberapa type data dasar sudah disediakan oleh pemroses, sehingga secara mudah bahasa pemrograman mengadopsinya, yaitu : *boolean, integer dan real.* Sedangkan type lain merupakan type bentukan yang memerlukan rutin khusus untuk mengimplementasikannya dalam bahasa pemrograman, misalnya : *string, character dan lain-lain.*

Contoh

Bahasa Pascal	:	*boolean, char, integer, real, string*
Bahasa C	:	*int, short int, long int, float, double, character, string*
Bahasa Fortran	:	*character, integer, real, logical, double precision*
Bahasa Snobol	:	*string, character*
Bahasa Lisp	:	*list (ekspresi S)*

Ukuran dan jangkauan data dari suatu tipe data tergantung dari kompilator dan komputer yang digunakan. Berikut adalah jangkauan dan ukuran tipe integer pada beberapa bahasa pemrorgraman.

Bahasa Pemrograman C

Tipe integer standar yang digunakan dalam bahasa C adalah tipe ***int.*** Ukuran dan jangkauan data dari tipe integer adalah sebagai berikut :

- **char :** tipe data ini digunakan untuk menyimpan karakter dalam kode ASCII, tapi dapat juga digunakan untuk menyimpan integer dari 0 sampai 255
- **short int :** ukuran 2 byte, jangkauan -32,768 sampai 32,767
- **int** : ukuran 4 byte, jangkauan -2,147,483,648 hingga 2,147,483,647

Tipe-tipe data di atas dapat menyimpan integer negatif dan positif.

Untuk menyimpan bilangan positif dan nol saja, dapat digunakan kata kunci unsigned sebelum tipe data. Sebagai contoh:

- **unsigned short int**, ukuran 2 byte, jangkauan 0 sampai 65,535
- **unsigned int**, ukuran 4 byte, jangkauan 0 sampai 4,294,967,295

Bahasa Pemrograman Pascal

Dalam bahasa Pascal, integer mampu menampung 16-bit. Walaupun memiliki ukuran 2 byte (16 bit) tetapi karena integer adalah type data signed maka hanya mampu di-assign nilai antara -215 hingga 215-1 yaitu -32768 sampai 32767. Ini disebabkan karena 1 bit digunakan sebagai penanda positif/negatif.

Meskipun memiliki istilah yang sama, tetapi tipe data integer pada bahasa pemrograman **Visual Basic.NET** dan **Borland Delphi** memiliki ukuran 4 byte atau 32 bit signed sehingga dapat di-assign nilai antara -2,147,483,648 hingga 2,147,483,647.

Selain tipe integer, bahasa Pascal juga memiliki beberapa tipe lain:

- **byte**, ukuran 1 byte, jangkauan dari 0 sampai 255
- **smallint**, ukuran 1 byte, jangkauan dari -128 sampai 127
- **word**, ukuran 2 byte, jangkauan dari 0 sampai 65,535

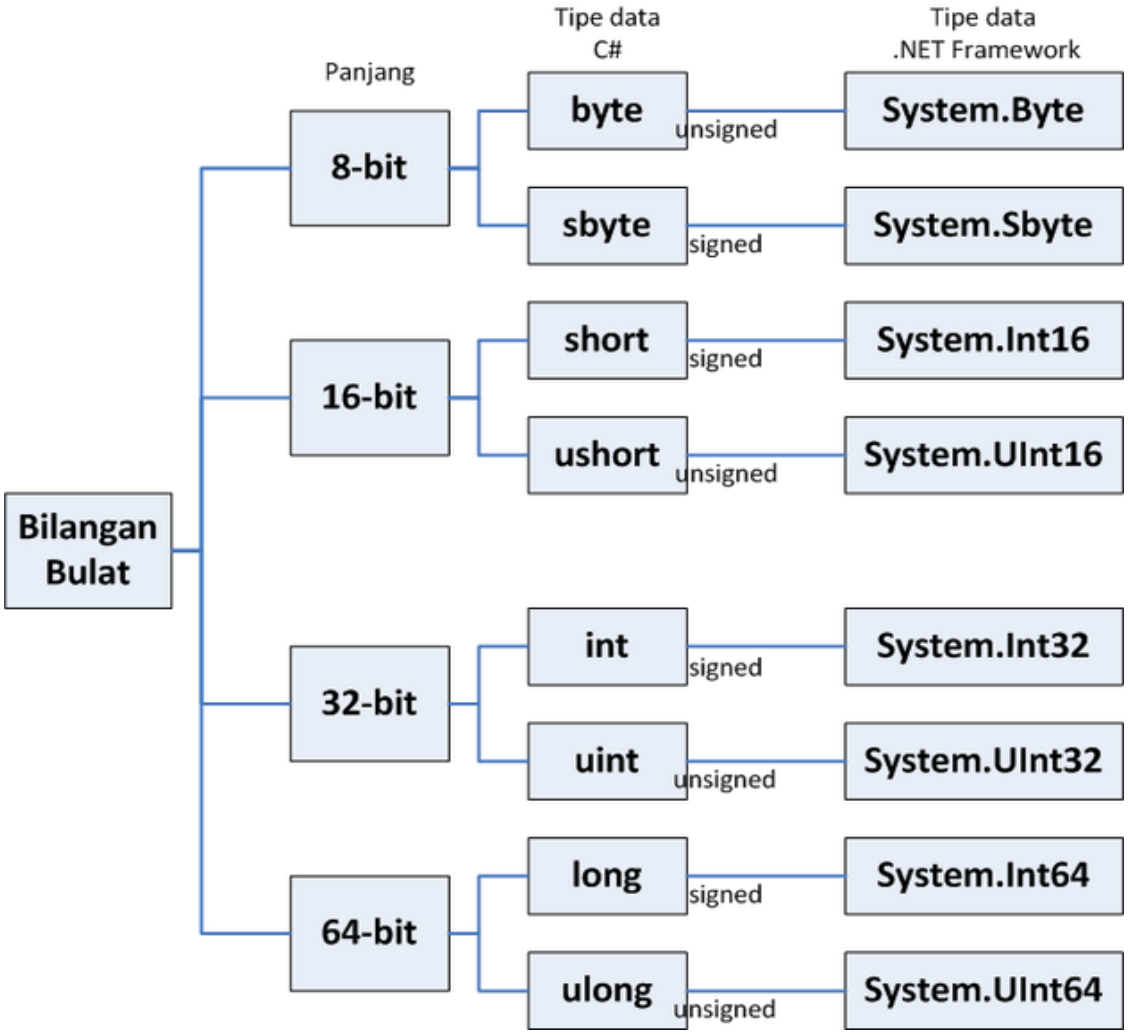
Pada kompilator Pascal yang lebih baru, juga dikenal tipe-tipe data yang lebih besar seperti:

- **longint**, ukuran 4 byte, jangkauan dari -2,147,483,648 sampai 2,147,483,647
- **cardinal**, ukuran 4 byte, jangkauan dari 0 sampai 4,294,967,295

Perbandingan nama tipe data bilangan bulat dalam bahasa pemrograman C# dan Microsoft .NET Framework, sebagai berikut :

- **byte**: bilangan bulat tak bertanda (unsigned integer) 8-bit. Ekuivalen dengan tipe data **System.Byte** dalam Microsoft .NET Framework.
- **sbyte**: bilangan bulat bertanda (signed integer) 8-bit. Ekuivalen dengan tipe data System.Sbyte dalam Microsoft .NET Framework.
- **short**: bilangan bulat bertanda 16-bit. Ekuivalen dengan tipe data System.Int16 dalam Microsoft .NET Framework.
- **ushort**: bilangan bulat tak bertanda (unsigned integer) 16-bit. Ekuivalen dengan tipe data System.UInt16 dalam Microsoft .NET Framework.
- **int**: bilangan bulat bertanda (signed integer) 32-bit. Ekuivalen dengan tipe data System.Int32 dalam Microsoft .NET Framework.
- **uint**: bilangan bulat tak bertanda (unsigned integer) 32-bit. Ekuivalen dengan tipe data System.UInt32 dalam Microsoft .NET Framework.

- **long**: bilangan bulat bertanda (signed integer) 64-bit. Ekuivalen dengan tipe data System.Int64 dalam Microsoft .NET Framework.
- **ulong**: bilangan bulat tak bertanda (unsigned integer) 64-bit. Ekuivalen dengan tipe data System.UInt64 dalam Microsoft .NET Framework.



Objek Data

Objek data adalah mengacu kumpulan elemen D (Domain)

Misalnya :

Objek data *integer* mengacu domain berikut :

D = { 0, ±1, ±2, ±3, … }

Objek data string karakter dengan panjang kurang dari 3 mengacu kepada domain berikut:

D = {‘ ‘, ‘A’, ‘B’, …, ‘Z’, ‘AA’, …, ‘ZZ’, ‘AAA’, ‘BBB’,… }

D dapat berjumlah berhingga atau tak berhingga. Jika D sangan besar maka diperlukan cara khusus untuk merepresentasikan elemen-elemen Domain D di komputer.

Struktur Data

Untuk membuat menjadi struktur data, kita harus melakukan aktifitas terhadap objek data, yaitu :

- Mendeskripsikan kumpulan operasi sah yang diterapkan terhadap elemen-elemen objek data
- Menunjukkan mekanisme kerja operasi-operasi

Secara terstruktur : struktur data adalah objek data ditambah himpunan operasi untuk memanipulasi objek.

Struktur data = objek data + {operasi manipulasi}

Operasi manipulasi harus dapat menjamin terpenuhinya spesifikasi relasi antar elemn-elemen data atau menjamin invarian.

Contoh :

1. Struktur data integer:

   Integer : D { 0, ±1, ±2, ±3, … }
   Operasi = {+, -, *, mod, div}

   Operasi manipulasi +, -, * menjamin terpenuhinya spesifikasi relasi antar elemen-elemen data integer. Sementara itu operasi bagi (/) tidak dapat menjamin spesifikasi relasi dan invarian, karena hasil operasi bagi (/) antara 2 integer tidak lagi menghasilkan type real, bukan integer. Operasi yang memenuhi sarat adalah DIV mengambil nilai bulat pembagian dan MOD mengambil sisa pembagian.

2. Struktur Data Waktu : t = (j : m : d)

   Waktu : Domain D { j,m,d | $0 \le j \le 23$, $0 \le m \le 59$, $0 \le d \le 59$, j,m,d $\in Z$ }
   Operasi { totaldetik = f(t), lama = f (t1, t2) }

III. DEFINISI FORMAL STRUKTUR DATA

Struktur data mendeskripsikan :
- Kumpulan Objek data
- Operasi dasar terhadap objek data
- Hubungan (relasi) antar objek data.

Secara formal :

Struktur data dapat dinyatakan sebagai *triple (D, F, A),* yaitu :

a. Domain D → d ∈ D

b. Himpunan fungsi F,

   yaitu kumpulan operasi-operasi yang legal terhadap anggota domain D

c. Himpunan aksiom A,

   yaitu kumpulan relasi atau invarian yang dipenuhi oleh anggota di domain D.

Triple (D,F,A) : disebut tipe data abstrak/TDA *(Abstract Data Type)*, karena imlementasinya tersembunyi dari pemakai, baik implementasi datanya maupun implementasi oparasinya.

TDA → pemodelan matematis terhadap objek data

TDA merupakan pemodelan matematis terhadap objek data dengan kumpulan operasi yang didefinisikan pada model tersebut. TDA merupakan generalisasi tipe data primitif.

Kegunaan spesifikasi struktur data yang digambarkan dengan TDA adalah :
- Untuk membuktikan kebenaran program yang menggunakan struktur data.
- Untuk membuktikan kebenaran implementasi struktur data.

Tujuan pembentukan struktur data adalah untuk mengkapsulkan informasi, yaitu :

- Perubahan implementasi struktur data tidak mengubah program yang menggunakan struktur data bila interface pada struktur data tersebut dirancang secara luwes sehingga independen terhadap program.
- Dengan pembakuan interface, pemakaian dan pembuatan struktur data dapat dilakukan secara terpisah.
- Struktur data adalah sarana pemrograman modular yang dapat dijadikan basis pembentukan team programmer.

Contoh :

1. TDA Type Integer
   Domain D = { *0, ±1, ±2, ±3, …}*
   Function F = {+, -, *, mod, div}
   Aksioma A = { }

2. TDA Type String
   Domain D = { s | s [x] = string ASCII, 1<x< 256 & x∈ Z }
   Function F = { penggalan = f(s[x],n,m) }
   Aksioma A = { }

3. TDA Type Jam
   Domain D = { j,m,d | $0 \le j \le 23$, $0 \le m \le 59$, $0 \le d \le 59$, j,m,d ∈ Z }
   Function F = { Lama = f1(t_0, t_1), TotalDetik = f2(t) }
   Aksioma A = { perhitungan lama dikoreksi jika tangga berbeda }

4. TDA Type Nilai Matakuliah UMB
   Domain D = { n, abs | $0 \le n \le 100$, $0 \le abs \le 14$, n ∈ R & abs ∈ Z }

   Function F = { n-uts = f1($mhs_{(i)}$, n),
   n-uas = f2($mhs_{(i)}$, n),
   n-tugas = f3($mhs_{(i)}$, n),
   n-prakt = f4($mhs_{(i)}$, n),
   n-abs = f5($mhs_{(i)}$, n)
   n-akhir = f6(n-uts, n-uas, n-tugas, n-prakt, n-abs)}

   Aksioma A = { n-abs $\le$ 75 $\Rightarrow$ n-akhir = “E” }

Soal Latihan :

1. Sajikan TDA Type Tanggal
2. Anda dihadapkan pada masalah pengolahan data transaksi di kasir sebuah super market. Barang dikodekan dengan barcode 12 digit
   - Definisikan objek-objek yang merepresentasikan persoalan.
   - identifikasi type data yang diperlukan, dan sajikan TDA nya

3. Anda dihadapkan pada persoalan pengolahan data parkir mobil di area Blok Z Plaza, untuk mengetahui berapa biaya yang harus dibayar.
   - Definisikan objek-objek yang merepresentasikan persoalan.
   - identifikasi type data yang diperlukan, dan sajikan TDA nya

IV. PEMBUATAN STRUKTUR DATA

Terdapat tiga tahap pembuatan struktur data :

Tahap pertama : spesifikasi

Tahap ke dua : implementasi

Tahap ke tiga : pemrograman

Tahap Pertama : Spesifikasi

Pendeskripsian atau spesifikasi struktur data yang merupakan batasan-batasan struktur data. Pendefinisian melibatkan logik sehingga sering digunakan ketetapan/teorema matematika untuk menyatakan sifat-sifat struktur data yang dikehendaki.

Spesifikasi dapat disajikan dangan dua cara :

- Spesifikasi secara formal
  Diwujudkan dalam bentuk Type Data Abstrak
- Spesifikasi secara informal
  Diwujudkan dengan membuat deskripsi dalam bahasa alami.

Tahap ke dua : Implementasi

Implementasi menyatakan cara penerapan struktur data dengan struktur data yang telah ada. Implementasi struktur data merupakan proses pendefinisian type data abstrak sehingga semua operasi dapat diekspresikan dengan fungsi-fungsi yang dapat dieksekusi oleh komputer.

Implementasi berisi deklarasi struktur penyimpanan item-item data serta algoritma-algoritma untuk implmentasi operasi-operasi yang menjadi karakteristik strukltur data.

Implementasi struktur data adalah pemetaan fungsi **d** ke kumpulan struktur data lain, **e**, yaitu :

Pemetaan domain : merepresentasikan objek **d** dengan objek **e**
Pemetaan Fungsi **n** : menggunakan fungsi-fungsi pada struktur data **e**

Tahap ke tiga : Pemrograman

Pemrograman struktur data adalah penterjemaahan menjadi pernyataan dalam bahasa pemrograman tertentu.

Terdiri dari proses :

- Deklarasi yang mendefinisikan objek-objek data dan hubungannya

- Pembuatan prosedur atau rutin untuk operasi-operasi dasar
- Perancang harus memilih tipe-tipe data yang telah ada untuk merepresentasikan struktur data.
- Struktur data dibangun menggunakan fasilitas pembetukan struktur data yang disediakan bahasa pemrograman.

Spesifikasi Non Formal

Untuk kebutuhan praktis, strukutur data dapat didefinisikan dengan bahasa alami (bukan matematis), yang melibatkan pendefinisian komponen-komponen berikut:

Deklarasi struktur data dengan type yang telah ada

Operasi-operasi terhadap tipe data sehingga dapat mengakomodir fungsi atau aksi yang dikehendaki.

Pembentukan struktur data adalah pembentukan tipe data lengkap, dengan properti sebagai berikut :

Nama	: sebagai identifier type data
Domain	: himpunan semesta nilai bertipe data
Penyebutan anggota	: cara penyebutan anggota-anggota type data
Operasi terhadap tipe	: daftar operasi terhadap anggota tipe data sesuai spesifikasi

DEKLARASI STRUKTUR DATA

I. PENGERTIAN DEKLARASI STRUKTUR DATA

Deklarasi struktur data adalah suatu pernyataan formal tentang representasi data dalam pemrograman. Terdapat dua tahapan deklarasi :

- Deklarasi Type Data
- Deklarasi Struktur Data

a. Deklarasi Type Data

Deklarasi type data dimaksudkan untuk mengkapsulkan informasi dasar tentang suatu type dalam suatu batasan objek (domain) dan operasi yang legal untuk type tersebut. Tujuan deklarasi type data, agar pemrogram tidak dirumitkan dengan penulisan secara berulang detil fungsi-fungsi untuk suatu type data.

Pada awalnya hanya tersedia type-type data yang didefinisikan secara built-in oleh bahasa pemrograman. Type built-in berorientasi pada cara data disimpan dalam sistem memory komputer, pada keadaan tertentu hanya memiliki hubungan yang kecil dengan permasalahan yang akan diselesaikan. Hal ini disebabkan karena type tersebut tidak didefinisikan dengan orientasi domain persoalan yang dihadapi pemrograman.

Untuk mengadaptasi berbagai keunikan persoalan dan kebutuhan pemrogram, diperlukan fasilitas untuk dapat mengekspresikan dan mendefinisikan type data sesuai dengan domain persoalan. Bahasa pemrograman yang ada, saat ini telah menyediakan fasilitas yang memungkinkan pemrogram mendefinisikan sendiri type-type data yang mendekati abstraksi type pada domain persoalan.

b. Deklarasi Struktur Data

Deklarasi struktur data mengacu pada objek permasalahan yang akan diselesaikan dengan suatu program/aplikasi. Dalam hal ini terfokus pada tata cara mewadahi data-data yang merupakan representasi permasalahan. Wadah data diimplementasikan dalam bentuk pendeklarasian variabel-variabel.

Deklarasi struktur data dilakukan setelah menganalisa himpunan data/objek dan himpunan kuasanya *(power set)*. Analisa terhadap data/objek akan menghasilkan pengklusteran/pengelompokan data/objek.

Untuk objek data yang direpresentasikan oleh item data tunggal, akan dideklarasikan dengan menggunakan veriabel bertype data tunggal. Sedangkan objek data yang membutuhkan sekumpulan item data (bundel data) diimplementasikan dalam deklarasi type record.

II. DEKLARASI TYPE DAN FUNCTION

Untuk type-type dasar sudah dideklarasikan oleh bahasa pemrograman, misalnya :

Type integer

Type Integer berlaku bagi objek data bilangan bulat → 0, ±1, ±2, dst

*Terhadap objek data ini hanya diizikan operasi +, -, *, DIV, dan MOD*

Operasi matematis lainnya akan ditolak. Terdapat pengecualian jika type tersebut hanya sebagai operand (mengambil nilai / get value / read only), misalnya :

A & B adalah integer,

Operasi bagi (/) diizinkan selama hasilnya disimpan ke variabel lain yang bertype real → C ⇐ A / B dimana C adalah Real

Contoh lain adalah type String.

Secara formal type integer dan type string disajikan dengan TDA sebagai berikut :

- TDA Type Integer

  Domain D = { *0, ±1, ±2, ±3, …}*

  Function F = {+, -, *, mod, div}

  Aksioma A = { }

- TDA Type String

  Domain D = { s | s [x] = string ASCII, 1<x< 256 & x∈ Z }

  Function F = { penggalan = f(s[x],n,m) }

  Aksioma A = { }

Begitu juga untuk beberapa type rakitan, sudah banyak dideklarasikan oleh bahasa pemrograman. Misalnya type *date*/tanggal dan type *time*/waktu saat ini sudah built-in dalam bahasa pemrograman.

Yang harus diingat ketika sebuah type data didefinisikan, maka operasi-operasi yang melekat pada type tersebut juga harus didefinisikan. Berikut ini akan diberikan beberapa contoh Deklarasi Type dan Function.

5. Type Jam

Data jam merupakan kombinasi hirarki 3 komponen yang basisnya adalah bilangan bulat dengan *range* tertentu, yaitu Jam, Menit, Detik. Domain untuk masing-masing komponen data memiliki range berbeda. Terhadap Jam, operasi yang umum dilakukan adalah : menghitung selisih 2 waktu (P/lama), menghitung nilai detik (TS) dari data jam (data skalar), mengambil komponen jam (Hr) dari data jam, mengambil komponen menit (Mn) dan mengambil komponen detik (Sc). Operasi selisih 2 waktu hanya untuk waktu pada hari/tanggal yang sama.

Agar lebih jelas, berikut ini didefinisikan dengan TDA Type Jam, yaitu:

Domain D = { j,m,d | $0 \le j \le 23$,
$0 \le m \le 59$,
$0 \le d \le 59$,
$j,m,d \in Z$ }

Function F = { Periode = f1(t_0, t_1),
TotalDetik = f2(t),
DJam = f3 (t),
DMenit = f4 (t),
DDtk = f5 (t) }

Aksioma A = { perhitungan lama dikoreksi jika tangga berbeda }

Deklarasi Type Jam

a. Type

```
jm   = [0…23]
ms   = [0…59]
wkt  = record of
       | j   : jm
       | m   : ms
       | d   : ms
  end-record
```

b. Function

Function TotalDetik (t : wkt) : Integer

```
Begin
   |   TotalDetik ⇐ (t.j * 3600) + (t.m * 60) + t.d
End TotalDetik
```

```
Function Periode(w0, w1 : wkt) : wkt
Var  dt0, dt1, dt : integer
Begin
    dt0 ⇐ w0.d + w0.m*60 + w0.j*3600
    dt1 ⇐ w1.d + w1.m*60 + w1.j*3600
    If  w1.j > w0.j
        Then  dt  ⇐  dt1 – dt0
        Else    dt  ⇐  dt0 – dt1
    End-if
    Periode.j ⇐  dt div 3600
    Periode.m ⇐ (dt mod 3600) div 60
    Periode.d ⇐ (dt mod 3600) mod 60
End-waktu
```

Untuk Function lainnya : DJam = f3 (t), DMenit = f4 (t), dan DDtk = f5 (t), silahkan deklarasikan fuction sebagai latihan.

6. Type Tanggal

Sama seperti data jam, data tanggal juga merupakan kombinasi hirarki 3 komponen yang basisnya adalah bilangan bulat dengan *range* tertentu, yaitu tanggal, bulan, dan tahun. Domain untuk masing-masing komponen data memiliki range berbeda. Terhadap Tanggal, operasi yang umum dilakukan adalah : menghitung selisih 2 tanggal, menghitung jumlah hari(Total_hari) dari data jam (data skalar), mengambil komponen jam (Hr) dari data jam, mengambil komponen menit (Mn) dan mengambil komponen detik (Sc). Operasi selisih 2 waktu hanya untuk waktu pada hari/tanggal yang sama.

Mendefinisikan Type Tanggal

Contoh data → 7 / 3 /2007

Identifikasi objek dan masalah-masalahnya

1. Objek

Tanggal → 1, 2, 3, 4.. 31

Bulan → 1, 2, .. 12

tahun → integer positif

2. Operasi yang legal :

menghitung hari → umur

nama bulan

3. aksiom

tahun kabisat, max hari perbulan berbeda-beda

TDA Tanggal

D : t = $\{d|\ 1 \le d \le 31,\ d \in Z\} \cup$

$\{m|\ 1 \le m \le 12,\ m \in Z\} \cup$

$\{y|\ y \ge 0,\ y \in Z\}$

F : fungsi = { hari = f(t1, t2), nabul = g(t.m) }

A : batas = **{** Max(t.m =1,3,5,7,8,10,12) = 31;

Max(t.m =4,6,9,11) = 30;

Max(t.m = 2) = 28;

t.y mod 4 = 0 $\Rightarrow$ max(t.m = 2) = 29 **}**

Deklarasi Type

```
Type    date = [1…31]
        month = [1…12]
        tgl  = record of
                        d : date
                        m: month
                        y : integer
               end-record
```

Deklarasi Fungsi

Function Hari(t1, t2 : tgl) : integer

Var h1, h2, h3, x : integer

Begin

```
Case t1.m of
        1    :  h1 ⇐  365 – t1.d
        2    :  h1 ⇐  334 – t1.d
         :
        12:  h1 ⇐  31 – t1.d
end-case

IF   (t1.y mod 4 = 0) and ( t1.m ≤ 2)
  then
        h1 ⇐ h1+1
end-IF

x   = (t2.y – t1.y – 1)
h2 = x * 365 + x div 4

IF  [(x mod 4 = 1)  and  (t1.y+1 mod 4 = 0)]  or
    [(x mod 4 = 2) and ((t1.y+1 mod 4 = 0) or (t1.y+2 mod 4 = 0))]  or
    [(x mod 4 = 3) and ((t1.y+1 mod 4 = 0) or (t1.y+2 mod 4 =0) or (t1.y+3 mod 4 =0]
  then
        h2 ⇐ h2+1
end-IF

Case t2.m of
        1   : h3 ⇐  t2.d
        2   : h3 ⇐  t2.d + 31
        3   : h3 ⇐  t2.d + 59
         :
        12: h3 ⇐  t2.d +334
end-case

IF  (t2.y mod 4 = 0) and (t2.m ≥ 3)
   then
        h3 ⇐  h3+1
end-IF

Hari ⇐  h1 + h2 + h3
```

End-function

Function Hari-koreksi(t1, t2 : tgl) : integer
Var h1, h2, h3, x : integer

{memperhentikan kemungkinan menghitung hari pada tahun yang sama }

Begin

IF t2.y > t1.y
Then

Case t1.m of
1 : h1 ⇐ 365 – t1.d
2 : h1 ⇐ 334 – t1.d
:
12: h1 ⇐ 31 – t1.d
end-case

IF (t1.y mod 4 = 0) and (t1.m ≤ 2) then h1 ⇐ h1+1 end-IF

x = (t2.y – t1.y – 1)
h2 = x * 365 + x div 4

IF [(x mod 4 = 1) and (t1.y+1 mod 4 = 0)] or
[(x mod 4 = 2) and ((t1.y+1 mod 4 = 0) or (t1.y+2 mod 4 = 0))] or
[(x mod 4 = 3) and ((t1.y+1 mod 4 = 0) or (t1.y+2 mod 4 =0) or (t1.y+3 mod 4=0]
then h2 ⇐ h2+1
end-IF

Case t2.m of
1 : h3 ⇐ t2.d
2 : h3 ⇐ t2.d + 31
:
12: h3 ⇐ t2.d +334
end-case

IF (t2.y mod 4 = 0) and (t2.m ≥ 3) then h3 ⇐ h3+1 end-IF

Hari ⇐ h1 + h2 + h3

else

Case t1.m of
1 : h1 ⇐ 365 – t1.d
2 : h1 ⇐ 334 – t1.d
:
12: h1 ⇐ 31 – t1.d
end-case

IF (t1.y mod 4 = 0) and (t1.m ≤ 2) then h1 ⇐ h1+1 end-IF

Case t2.m of
1 : h3 ⇐ t2.d
2 : h3 ⇐ t2.d + 31
:
12: h3 ⇐ t2.d +334
end-case

IF (t2.y mod 4 = 0) and (t2.m ≥ 3) then h3 ⇐ h3+1 end-IF

IF t1.y mod 4 = 0
then Hari ⇐ h1 + h3 -366
else Hari ⇐ h1 + h3 -365
end-IF

end-IF
End-function

III. Deklarasi Struktur Data (REVIEW)

Kegunaan spesifikasi struktur data yang digambarkan dengan TDA adalah :

1. Untuk menjamin dan membuktikan kebenaran program yang menggunakan struktur data.

2. Untuk menjamin dan membuktikan kebenaran implementasi struktur data.

→ dengan mengetahui objek data, fungsi, aksioma

Pemrograman dapat diarahkan dalam batasan abstraksi data tersebut saja

Tujuan pembentukan struktur data adalah untuk mengkapsulkan informasi, yaitu :

1. Perubahan implementasi struktur data tidak mengubah program yang menggunakan struktur data bila interface pada struktur data tersebut dirancang secara luwes sehingga data independen terhadap program (prinsip independensi).
2. Dengan pembakuan interface, pemakaian dan pembuatan struktur data dapat dilakukan secara terpisah.
3. Struktur data adalah basis pemrograman modular yang dapat dijadikan dasar pembentukan team programmer atau pembagian kerja.

PEMBUATAN STRUKTUR DATA

Terdapat tiga tahap pembuatan struktur data :

1. Tahap pertama : spesifikasi
2. Tahap ke dua : implementasi
3. Tahap ke tiga : pemrograman

} **SPESIFIKASI, DEKLARASI STRUKTUR DATA DAN DEKLARASI FUNGSI**

Tahap Pertama : Spesifikasi

Pendeskripsian atau spesifikasi struktur data yang merupakan batasan-batasan struktur data.
Pendefinisian melibatkan logik sehingga sering digunakan ketetapan/ teorema matematika untuk menyatakan sifat-sifat struktur data yang dikehendaki.

Spesifikasi dapat disajikan dangan dua cara :

1. Spesifikasi secara formal
   Diwujudkan dalam bentuk Type Data Abstrak
2. Spesifikasi secara non formal

   Diwujudkan dengan membuat deskripsi dalam bahasa alami.

Tahap ke dua : Implementasi

Implementasi berisi deklarasi struktur penyimpanan item-item data (nama variabel dan typenya), serta algoritma-algoritma (algoritma fungsi) untuk implmentasi operasi-operasi yang menjadi karakteristik struktur data.

Contoh : objek jam pada kasus parkir kendaaraan

```
Objek jam :   hh:mm:ss
   Type       h = [0 ... 23]
              m = [0…59]
              s = [0…59]
              TJam = record
                       jam  :  h
                       mnt  : m
                       dtk   :  s
                 end-Tjam
   var
        Jmasuk, Jkeluar : Tjam
```

Function lamaparkir (jMasuk, jkeluar : tJam) : Tjam

```
{asumsi bukan parkir menginap}
Var  dtkmasuk, dtkkeluar, dtklama : integer
Begin
              Dtkmasuk <=  jmasuk.jam*3600 + jmasuk.mnt*60 + jmasuk.dtk
              Dtkkeluar <=  jkeluar.jam*3600 + jkeluar.mnt*60 + jkeluar.dtk
              Dtklama <= dtkkeluar – dtkmasuk

              Lamaparkir.jam <= dtklama div 3600
              Lamaparkir.mnt <= (dtklama mod 3600) div 60
              Lamaparkir.dtk <= (dtklama mod 3600) mod 60
End-function
```

Implementasi objek nopol kendaraan adalah :

```
Var   nopol : string[9]
```

Mengacu pada masalah yang harus diselesaikan → tidak memerlukan deklarasi fungsi

Tahap ke tiga : Pemrograman

Pemrograman struktur data adalah penterjemaahan menjadi pernyataan dalam bahasa pemrograman tertentu.

Terdiri dari proses :

1. Deklarasi yang mendefinisikan objek-objek data dan hubungannya
2. Pembuatan prosedur atau rutin untuk operasi-operasi dasar

Perancang harus memilih tipe-tipe data yang telah ada untuk merepresentasikan struktur data.

Struktur data dibangun menggunakan fasilitas pembetukan struktur data yang disediakan bahasa pemrograman.