2 MANIPULASI DATA DALAM PEMBUATAN PROGRAM

Konsep umum dari Algoritma yang disusun dalam Bahasa Pemrograman tingkat tinggi, beserta rinciannya dalam bahasa Pascal akan dibahas di dalam Bab 2 ini, dan juga dalam Bab 3 berikutnya. Konsep tersebut mencakup Bahasa Pemrograman yang umum dan serba guna, seperti bahasa Pascal ataupun FORTRAN dan sebagainya. Di sini kita akan memperoleh dasar yang kuat, agar bisa lebih baik dan mudah dalam mempelajari Algoritma dan Bahasa Pemrograman tersebut lebih lanjut. Nantinya, kita akan mengkonsentrasikan diri pada pembahasan mendalam, khusus terhadap bahasa Pascal.

Di samping itu, bentuk Ekspresi yang didasarkan pada konsep tersebut, akan dipakai untuk membuat perancangan Algoritma, sebelum kita membuat kode yang sebenarnya. Komponen Program yang dipakai untuk memanipulasikan Data, juga dibahas di dalam Bab ini. Data akan memberikan informasi tentang obyek dari Algoritma secara umum. Umpamanya, suatu Program untuk menghitung pajak pendapatan, antara lain membutuhkan Data tentang besarnya pendapatan Wajib Pajak, dan Klasifikasi Pungutan Pajak yang berlaku. Data yang umum, jika disajikan dalam Struktur Data yang tepat, akan dapat menghasilkan sebuah Algoritma yang

lebih sederhana. Misalnya, jika Klasifikasi Pungutan Pajak disusun dalam sebuah Tabel, kemudian bersama Data penghasilan, dimanipulasi menggunakan Program perhitungan pajak, sehingga nilai pajak yang menjadi kewajiban seorang Wajib Pajak, dapat ditentukan dengan mudah. Juga, spesifikasi tersebut dapat dengan mudah dimodifikasi seperlunya.

Dengan demikian, berbagai Jenis dan Struktur Data dapat dipakai untuk menyajikan obyek nyata. Jelas bahwa terdapat saling ketergantungan di antara kedua faktor tersebut, Algoritma dan Struktur Data, untuk menghasilkan Program yang baik. Data dari jenis yang sederhana juga diperkenalkan di dalam Bab 2 ini.

2.1 PERNYATAAN (STATEMEN)

Program merupakan bentuk lain, atau penyajian lain dari sebuah Algoritma, guna keperluan pelaksanaan oleh komputer. Suatu Program umumnya tidak ditulis dalam bahasa mesin, tetapi dalam bahasa tingkat tinggi. Oleh karena itu, komputer harus meminta bantuan Penterjemah (berupa Kompilator atau Interpreter) untuk menterjemahkan bahasa tersebut terlebih dahulu, kemudian baru dilaksanakannya.

STATEMEN merupakan komponen dasar dari sebuah Program. Statemen pada Program tersebut dapat kita golongkan atas statemen yang dilaksanakan, dan Statemen yang tidak dilaksanakan. STATEMEN YANG DILAKSANAKAN, adalah merupakan perintah yang diberikan kepada Central Processing Unit (CPU) komputer. Statemen tersebut harus dijalankan; dan terlebih dahulu harus diinterpretasikan ke dalam bentuk instuksi mesin. STATEMEN YANG TIDAK DILAKSANAKAN, merupakan perintah yang diberikan kepada Penterjemah Bahasa atau Program sistem yang lain. Ia tidak diubah sebagai instruksi dalam bahasa mesin, dan tidak termasuk di dalam Program obyek yang bersangkutan.

Statemen mengandung berbagai uraian. Biasanya berfungsi untuk mengalokasikan ruangan di dalam memori bagi Data yang akan dimanfaatkan oleh suatu Program. Cara lain untuk membedakan apakah suatu Statemen merupakan Statemen Bahasa Pemrograman, ataukah Statemen yang lain, tergantung dari keadaan apakah ada pengaruh terhadap arus pengendalian pelaksanaan Program.

Statemen dilaksanakan secara berurutan sesuai dengan pemasukannya ke dalam komputer, kecuali jika terdapat Statemen kendali **alih jalur** yang mampu mengendalikan dan mengubah urutan pelaksanaan Statemen. Kendali alih jalur dapat bersifat tak bersyarat ataupun bersyarat. Pada alih bersyarat, keputusan atau decision didasarkan pada terlaksana tidaknya suatu Kondisi tertentu.

Seringkali pula kita memerlukan penempatan lebih dari satu Statemen di dalam alternatif cabang Statemen keputusan atau di dalam sebuah gelung. Hal tersebut dapat dilakukan dengan memakai Statemen Majemuk.

STATEMEN MAJEMUK merupakan serangkaian Statemen yang dilaksanakan berurutan. Untuk menuliskannya dalam Bahasa Pemrograman, Pascal misalnya, ia diawali dengan kata BEGIN dan diakhiri dengan kata END.

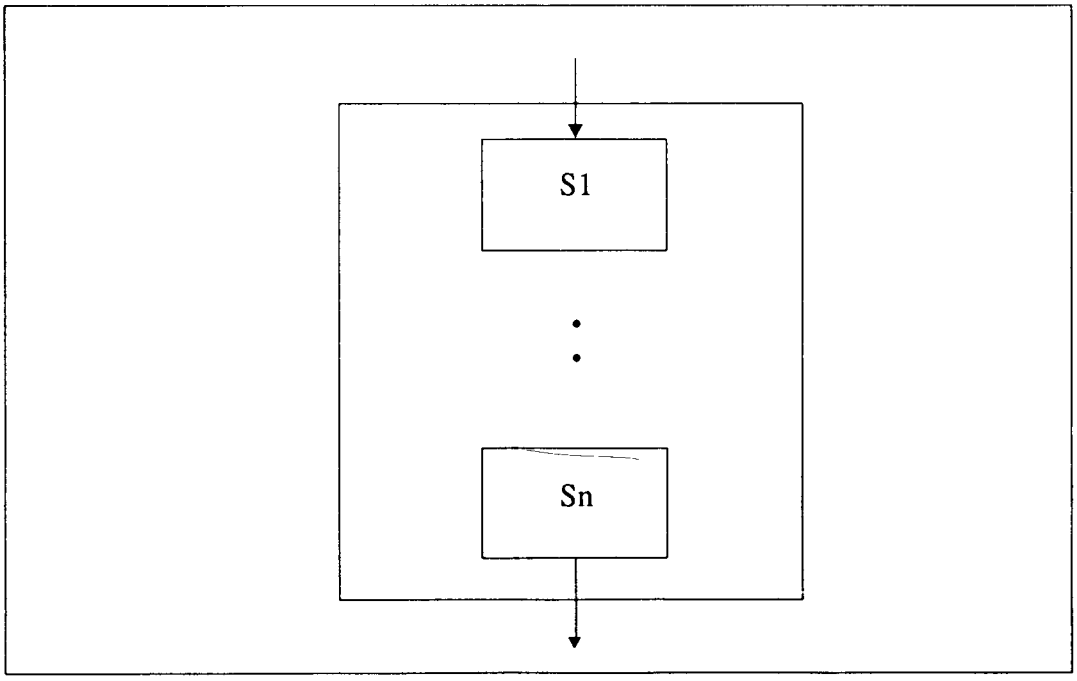
Bentuk dari Statemen Majemuk adalah :

BEGIN S1; S2; ...; Sn **END** •

di sini S1, S2, ...,Sn masing-masing adalah suatu Statemen.

Statemen Majemuk tersebut mengakibatkan pelaksanaan secara sekuensial, dimulai dari Statemen S1, berturut-turut sampai dengan Statemen Sn. Struktur **BEGIN-END**, biasa disebut pula struktur Barisan atau Sequence.

Diagram alur dari struktur tersebut diperlihatkan pada Gambar berikut ini.



GAMBAR Barisan (**BEGIN-END**)

Jika sebuah Statemen Majemuk tidak ditulis dalam satu baris, maka bentuknya disusun sebagai berikut :

```
BEGIN
    S1;
    •
    •
    •
    Sn
END.
```

Setiap program yang terdiri lebih dari satu Statemen, pada hakikatnya sudah merupakan Statemen Majemuk.

Setiap komponen dari Statemen Majemuk tersebut (kecuali yang terakhir) dibatasi dari Statemen berikutnya oleh sebuah titik-koma, sedangkan Statemen yang terakhir dibatasi oleh kata **END**. Tanda tersebut memungkinkan kita untuk menempatkan berbagai Statemen dalam satu baris.

Statemen Majemuk yang paling sederhana adalah Statemen Hampa, yakni

```
BEGIN


END .
```

Di sini, tak terdapat satupun Statemen antara **BEGIN** dan **END.**
Bentuk di atas adalah sah (juga dalam Pascal).

Sebagai contoh lain, misalnya sederetan Instruksi untuk mengirimkan surat kepada Teman (instruksi dalam bahasa Indonesia), ingin kita tulis sebagai Statemen Majemuk, misalnya demikian

```
BEGIN
Tulis surat pada kertas surat;
Ambil sampul surat;
Tulis alamat pada sampul surat;
Masukkan surat pada sampul surat;
Ambil perangko secukupnya;
Ambil perekat;
Lekatkan perangko pada sampul surat;
Tutup sampul surat menggunakan perekat;
Masukkan surat dalam bis surat
END .
```

2.2 TIPE DATA SEDERHANA

Algoritma dan Program yang didasarkan pada suatu Tipe Data akan beroperasi menggunakan Data tersebut. Kegiatan dari Statemen yang dapat dilaksanakan tercermin dari perubahan nilai suatu satuan Data.

Data input diubah oleh Program, setelah melalui tahapan proses menjadi Data Output. Di dalam proses pemecahan masalah, perancangan dari Struktur Data adalah sama pentingnya dengan perancangan dari Algoritma, dan perancangan dari Program yang didasarkan pada Struktur Data tersebut. Hanya DATA SEDERHANA saja (tidak berstruktur) yang akan dibahas dalam bab ini. DATA YANG BERSTRUKTUR (dibahas di dalam Bab yang akan datang) merupakan kumpulan satuan (field) Data sederhana dan mempunyai hubungan satu dengan yang lain dalam aturan yang tertentu.

Satuan Data, dan instruksi Program dimasukkan ke dalam memori komputer berupa satu, atau umumnya beberapa BIT. Suatu Tipe Data merupakan penafsiran yang diterapkan pada sejumlah bit yang mewakili satuan Data tertentu. Proses atau operasi yang diterapkan pada suatu satuan Data, adalah tergantung pada jenis satuan Data tersebut. Bahasa Pemrograman tingkat tinggi, termasuk Pascal, biasanya dilengkapi dengan Statemen Deklarasi, yang dapat dipakai oleh Pemrogram untuk menyatakan serta merinci jenis satuan Data di dalam Program.

Berikut ini adalah garis besar Tipe Data sederhana, dan nilai lingkupnya:

1. *Integer :*
   melingkupi himpunan bilangan bulat ..., -3,-2,-1,0,1,2,3,...
2. *Real :*
   melingkupi himpunan bilangan real, termasuk Integer, pecahan dan irrasional. Baik Tipe Integer maupun Real tersebut merupakan Data numerik. Selain kedua Tipe Data numerik di atas, ada pula beberapa Tipe Data numerik lain, yakni misalnya Tipe Data Kompleks di dalam FORTRAN.
   Untuk menuliskan data real kita dapat menggunakan penulisan titik desimal (dalam bahasa Indonesia menggunakan koma desimal), seperti misalnya : 37.125
   Selain cara tersebut, biasanya dalam Bahasa pemrograman digunakan penulisan eksponensial, misalnya : 37.125 ditulis sebagai 3.7125E+01 atau 371.25E-01
3. *Boolean* atau Logika:
   merupakan jenis Data yang hanya memiliki 2 nilai, yakni benar (True), dan salah (False), yang juga diwujudkan sebagai 1 dan 0. Jenis Data tersebut terutama sekali dipakai untuk menunjukkan Kondisi yang memerlukan keputusan untuk kelanjutan proses. Tipe Data tersebut dapat dimanfaatkan untuk mewakili suatu obyek yang memiliki nilai binar.
4. *Character :*
   meliputi huruf dalam Alfabet (bersifat Alfabetik) serta simbol Digit (disebut karakter Alphanumerik) atau simbol khusus seperti $, !, dan sebagainya.
   Karakter tersebut biasanya diorganisir dalam bentuk barisan karakter, yang disebut Untai atau String. Angka atau Digit sebagai satuan Data numerik adalah berbeda dengan Angka sebagai String.

Adanya jenis Data Character serta String tersebut akan memperluas kemampuan komputer dari hanya sekedar alat untuk mengerjakan kalkulasi numerik, menjadi alat pemroses Data simbolik, atau Text-processing.

Sebagian besar dari bahasa komputer serba-guna masa kini memiliki keempat jenis Data tersebut. Adanya berbagai jenis Data akan memudahkan kita untuk merancang Program di dalam suatu Bahasa Pemrograman tertentu. Beberapa bahasa memungkinkan Pemrogram untuk merumuskan jenis Data sendiri guna mendukung aplikasinya.

Selain itu, ada suatu jenis Data, yakni Pointer, yang terdapat di dalam beberapa Bahasa Pemrograman tingkat tinggi. POINTER menunjukkan "alamat" dari suatu satuan Data. Pointer sangat berguna untuk dipakai menyusun suatu Struktur Data.

Kita mengenal pula pengertian Tipe Data yang ORDINAL, yakni bahwa elemen data pada Tipe tersebut memungkinkan untuk kita urutkan secara diskrit. Artinya di sini jelas siapa "Pendahulu langsung" atau PREDECESSOR, serta "Penyusul langsung" atau SUCCESSOR (bila ada) dari setiap elemen Tipe Data.

Jadi yang termasuk Tipe Data Ordinal adalah INTEGER, dan CHARACTER. Tipe Data REAL ataupun BOOLEAN bukan Ordinal.

2.3 JENIS OPERASI PADA TIPE DATA

Pada Tipe Data Integer dapat dilakukan operasi sebagai berikut:

(1) Penambahan A + B

(2) Pengurangan A - B

(3) Perkalian A * B

(4) Pembagian Integer A **DIV** B
Operasi ini menghasilkan sebuah Integer, sebagai hasilbagi (biasa) dari A oleh B, tanpa memperhatikan sisa pembagian.
Sebagai Contoh :

34 DIV 7 = 4, karena 34 : 7 = 4 sisa 6
12 DIV 4 = 3, karena 12 : 4 = 3 sisa 0
5 DIV 8 = 0, karena 5 : 8 = 0 sisa 5
–9 DIV 2 = -4, karena -9 : 4 = -2 sisa -1
5 DIV 0 tidak terdefinisi

(5) Modulo A **MOD** B
Operasi ini menghasilkan sebuah Integer, yang merupakan sisa dari hasilbagi (biasa) dari A oleh B.
Sebagai Contoh:

34 MOD 7 = 6, karena 34 : 7 = 4 sisa 6
12 MOD 4 = 0, karena 12 : 4 = 3 sisa 0
5 MOD 8 = 5, karena 5 : 8 = 0 sisa 5
-9 MOD 2 = -1, karena -9 : 4 = -2 sisa -1
5 MOD 0 tidak terdefinisi

(6) Pemangkatan A ^ B

(7) Negatif –A
Misalnya A = 5, maka -A = -5, dan bila A = - 11, maka -A = 11

(8) Harga mutlak **ABS** (A)
Di sini didefinisikan bahwa ABS(A) adalah A bila A positif, dan adalah -A bila A negatif. Sebagai contoh, untuk A = 7, maka ABS(A) = ABS(7) = 7, dan untuk A = -7, maka ABS(A)= ABS(-7)= 7

(9) Predecessor **PRED**(A)
Operasi ini bermaksud menentukan Predecessor dari A. Jadi PRED(54) adalah 53, PRED(-13) adalah -14, dan sebagainya.

(10) Successor **SUCC**(A)
Operasi ini bermaksud menentukan Successor dari A. Jadi SUCC(54) adalah 55, SUCC(-13) adalah -12, dan sebagainya.

Dapat dicatat bahwa Operasi (1) sampai dengan (5) merupakan Operasi **BINAR**, yakni Operasi yang selalu membutuhkan 2 Operand (boleh sama) dalam setiap pengoperasinya. Operasi (6) sampai dengan (10) merupakan Operasi **UNAR**, yang hanya menggunakan satu Operand dalam setiap pengoperasinya. Selain kesepuluh Operasi tersebut, tentunya masih banyak lagi Operasi lain pada Integer yang dapat didefinisikan.

Pada Tipe Data Real dapat dilakukan Operasi sebagai berikut:

(1) Penambahan A + B

(2) Pengurangan A – B

(3) Perkalian A * B

(4) Pembagian A : B atau A / B, dengan B tidak sama dengan 0

(6) Pemangkatan A ^ N, dengan N merupakan Integer
Khusus untuk pangkat dua atau kuadrat, disediakan pula bentuk **SQR**(A).
Jadi SQR(2.50) = 6.25
SQR(-3.00) = 9.00
Selain itu, khusus untuk Real Positif, dapat dihitung akarnya, yakni **SQRT**(A).
Jadi SQRT(2.56) adalah 1.60
SQRT(16.0) adalah 4.0
SQRT(-16.0) tidak terdefinisi
Perhatikan, bahwa SQR dalam bahasa BASIC justru dimaksudkan untuk menghitung akar. Jadi dalam BASIC, SQR(4.0) adalah 2.0, bukan 16.0.

(7) Negatif –A

(8) Harga Mutlak **ABS**(A)

(9) Pembulatan **TRUNC**(A)
menentukan bagian bulat dari bilangan Real dengan lebih dahulu membulatkannya. Sebagai Contoh :

TRUNC(13.34) adalah 13
TRUNC(45.68) adalah 46
TRUNC(26.50) adalah 27
TRUNC(-32.12) adalah -32

(10) Pengambilan bagian Bulat **INT(A)**
menentukan bagian bulat dari bilangan Real dengan langsung membuang bagian pecahan. Sebagai Contoh :

INT(13.34) adalah 13
INT(45.68) adalah 45
INT(26.50) adalah 26
INT(-32.12) adalah -32

(11) Pengambilan bagian Pecahan **FRAC(A)**
menentukan bagian pecahan dari bilangan Real dengan langsung membuang bagian Bulatnya. Sebagai Contoh :

FRAC(13.34) adalah 0.34
FRAC(45.68) adalah 0.68
FRAC(26.50) adalah 0.50
FRAC(-32.12) adalah -0.12

Dalam pelaksanaan pada Bahasa Pemrograman, Turbo Pascal misalnya, Tipe Data Integer maupun Real tersebut di atas masih diperinci lagi. Tipe Data Integer diperinci sebagai Tipe Data **BYTE**, Tipe Data **SHORTINT** (Short Integer), Tipe Data Integer, serta Tipe Data **LONGINT** (Long Integer). Perincian ini didasarkan Daerah Jangkauan nilainya. Hal yang hampir serupa juga terjadi pada Tipe Data Real. Untuk lebih jelasnya hal ini, anda dapat mengikutinya pada pembicaraan mengenai Bahasa Pascal (Turbo Pascal khususnya) pada Bagian akhir dari bab ini.

2.4 KONSTANTA, VARIABEL DAN HIMPUNAN

Nilai dari suatu satuan Data sederhana di dalam suatu Algoritma dapat bersifat tetap (konstan), ataupun bersifat dapat berubah-ubah (Variabel). Oleh karena itu, satuan Data tersebut dibagi lagi menjadi **Konstanta** dan **Variabel.**

2.4.1 KONSTANTA DAN VARIABEL

Suatu **Konstanta** merupakan Satuan Data, atau Besaran yang tidak pernah berubah nilainya selama komputasi atau pelaksanaan Algoritma atau Program.

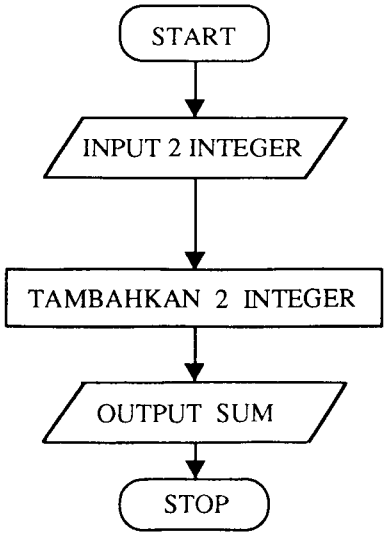
Suatu **Variabel** merupakan Satuan Data, atau Besaran yang nilainya dapat berubah-ubah ketika Program dilaksanakan.

Suatu Konstanta ataupun Variabel, dapat diberi nama tertentu sebagai **Pengenal** atau **Identifier**. Setiap Bahasa Pemrograman memiliki Aturan tersendiri mengenai pemberian Nama sebagai Identifier tersebut. Konstanta dan Variabel dapat mempunyai lingkup data dari berbagai Tipe Data, sehingga ada Konstanta Logika, Variabel Real, Variabel Integer, dan sebagainya.

CONTOH 2-1.

Tambahkan dua Integer, dan keluarkan hasilnya.

Algoritma:
Diagram alur dari Algoritma tersebut diperlihatkan di dalam gambar 2-1



GAMBAR 2-1.

Di Jalam contoh di atas terlihat bahwa Integer merupakan sebuah Konstanta (2); integernya sendiri merupakan Variabel. Pemakaian Variabel hanya dapat dilakukan melalui lokasi memori tempat Variabel tersebut disimpan. Nilai Variabel tersebut merupakan nilai dari suatu lokasi.

Komputer mampu memakai Variabel sedangkan kalkulator biasanya tidak. Nilai dapat disimpan sementara waktu hingga saat diperlukan di dalam pelaksanaan Program. Variabel dialokasikan ke dalam ruang memori. Ini biasanya sebagai hasil dari Statemen Deklarasi. Untuk dapat menggunakan kembali Variabel tersebut, kita harus memakai "alamat" lokasinya. Jika memakai Bahasa Pemrograman tingkat tinggi, maka pemrogram tidak perlu memikirkan alamat Numerik yang sesungguh-

nya. Lokasi tertera secara simbolik. Alamat simbolik mengandung arti bahwa Program yang bersangkutan mengetahui lokasi cukup dengan menyebutkan Nama Variabel yang terdapat di dalamnya.

Perangkat Penterjemah bahasa, Sistem Operasi, dan perangkat sistem lainnya, akan menghubungkan nama Variabel yang dipakai di dalam Program dengan lokasi numerik di dalam memori. Nama Variabel ditulis oleh Pemrogram sesuai dengan aturan pemberian Nama yang terdapat di dalam Bahasa Pemrograman tertentu tersebut. Misalnya saja, pada bahasa Turbo Pascal, nama Variabel serta Identifier lainnya, harus terdiri dari barisan huruf dan angka saja, namun boleh memuat simbol Strip-Bawah, harus diawali dengan suatu huruf, dan tidak diselingi oleh BLANK. Selain itu panjang Nama hanya sebanyak 63 karakter yang dianggap signifikan, selebihnya diabaikan.

Adapun mengenai apakah harus digunakan huruf besar ataukah huruf kecil, tidaklah menjadi masalah. Turbo Pascal tidak membedakannya.

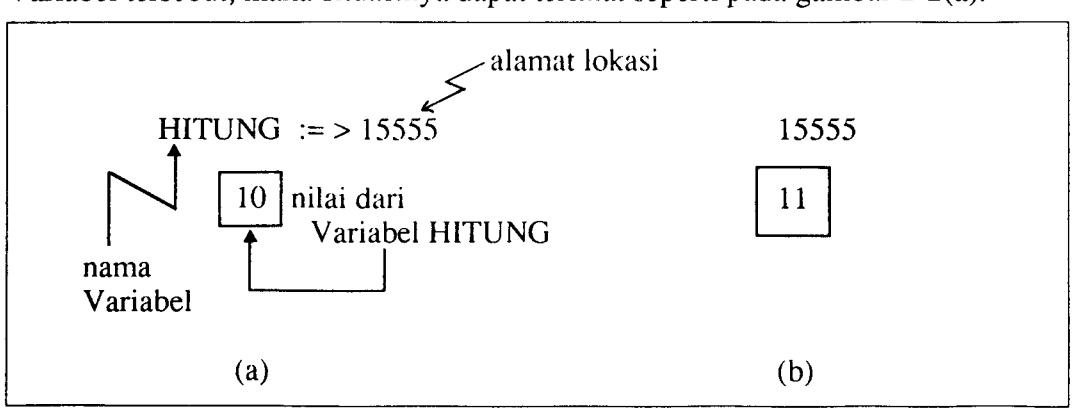
Contoh :

Perhatikan pemberian Nama sebagai Identifier pada Program Turbo Pascal berikut ini :

Tanggal – lahir	(benar)
Anak&Isteri	(salah, karena memuat tanda &)
NILAI2	(benar)
9orang	(salah, karena tidak diawali dengan huruf)
Upah Mingguan	(salah, karena diselingi BLANK)

CONTOH 2-2

Kita asumsikan bahwa sebuah Statemen Deklarasi menetapkan HITUNG sebagai Variabel Integer. Jika lokasinya beralamat 15555 dan dikirimkan kepada Variabel tersebut, maka situasinya dapat terlihat seperti pada gambar 2-2(a).



GAMBAR 2.2

Lokasi dengan alamat 15555 mengandung nilai Variabel yang disebut HITUNG. Nilai tersebut semula adalah 10. Jika dilaksanakan Statemen :"Menambah HITUNG dengan 1," maka yang berubah hanyalah nilainya saja, seperti terlihat pada gambar 2-2(b).

Suatu nilai dapat diberikan kepada suatu Variabel dengan 2 cara, yakni dibaca dari alat input, atau dimasukkan pada waktu pelaksanaan Program. Tipe Nilai yang diberikan kepada suatu Variabel adalah tergantung dari Tipe Variabel tersebut.

2.4. 2 DEKLARASI KONSTANTA DAN VARIABEL

Sebuah Deklarasi berisi informasi, yang terutama diperuntukkan bagi perangkat Penterjemah bahasa, tentang jenis Variabel yang bersangkutan. Jika dipakai Struktur Data, maka diperlukan pula informasi tentang jenis Struktur Data tersebut. Dengan memakai informasi tersebut, perangkat Penterjemah dan Sistem Operasi dapat mengalokasikan ruang memori, untuk diisi Data, menghubungkan "alamat" lokasi yang tersedia untuk Variabel dengan Namanya, dan menyiapkan bentuk operasi yang dapat diaplikasikan pada Data tersebut. Jika suatu Variabel sudah dinyatakan melalui Statemen seperti itu, maka Nama Variabel tersebut dapat digunakan berulang kali di dalam Program.

Di dalam CONTOH 2-1 kedua Variabel input, dan Variabel JUMLAH merupakan Variabel Integer. Kita dapat mendeklarasikan mereka dengan :

```
Integer NUM1, NUM2, JUMLAH
```

atau di dalam Pascal, sebagai:

```
VAR  NUM1,NUM2,JUMLAH : Integer;
```

yang boleh pula ditulis satu-persatu, sebagai:

```
VAR
    NUM1     :  Integer;
    NUM2     :  Integer;
    JUMLAH   :  Integer;
```

Variabel tersebut selanjutnya dapat "dipanggil" dengan Nama tersebut. Dalam Bahasa Pascal, keberadaan suatu Identifier Konstanta ataupun Variabel dalam Program, harus dideklarasikan menggunakan Statemen Deklarasi. Dalam Pascal, Statemen Deklarasi Konstanta diawali Reserved-word **CONST**. Bentuk umumnya adalah :

```
CONST
    K1  =N1;
    K2  = N2
          •
          •
          •
    Kn  =Nn;
```

Di sini K1,K2, ..., Kn adalah Nama Konstanta dengan nilainya berturut-turut N1, N2, ..., Nn, sesuai Tipe Data masing-masing.

Sedangkan Statemen Deklarasi diawali dengan Reserved-word **VAR.** Bentuk umumnya adalah:

```
VAR
    V1  :  T1;
    V2  :  T2;
           •
           •
           •
    Vk  :  Tk;
```

Di sini V1, V2, ..., Vk adalah Nama Variabel, berturut-turut dengan Tipe Data T1, T2, ..., Tk.

CONTOH

Pandang misalnya sekelompok Konstanta dan Variabel : Konstanta Integer Maksimum_Anak yang bernilai 2, Konstanta Real bernama Tingkat_Bunga yang bernilai 1.24, Konstanta String bernama Lokasi dengan nilai JAKARTA, dan Konstanta Logika bernama PTS dengan nilai True, serta Variabel Integer bernama Jumlah_Anak, Variabel Real bernama Pinjaman, Variabel String dengan panjang maksimal 25 karakter bernama Kota_Tinggal, serta Variabel Logika atau Boolean bernama Lulus, merupakan Identifier yang digunakan dalam sebuah Program Pascal. Mereka dapat kita deklarasikan sebagai berikut :

```
CONST
    Maksimum _ Anak =  2;
    Tingkat _ bunga   =  1.24;
    Lokasi            =  'JAKARTA';
    PTS               =  True;
```

```
VAR
        Jumlah _ Anak     :  Integer;
        Pinjaman          :  Real;
        Tempat _ Tinggal  :  String[25];
        Lulus             :  Boolean;
```

2.4.3 HIMPUNAN

Suatu **himpunan** dapat diartikan adalah *kumpulan dari sejumlah satuan Data yang sejenis,* dan disebut secara kolektif dengan sebuah nama saja. Masing-masing satuan Data disebut **elemen** himpunan. Mereka diurutkan berdasarkan nomor indeksnya; oleh karena itu, elemen himpunan kadang kala juga disebut **subvariabel.** Banyaknya indeks dari himpunan disebut **tingkat** atau **dimensi** himpunan. Suatu elemen dapat diketahui dari nama himpunan tersebut serta subvariabelnya yang ditulis dalam tanda kurung. Himpunan yang umum dipakai adalah *Vektor*,yang merupakan himpunan satu dimensi; Matriks, yang merupakan himpunan dua dimensi. Setiap elemen himpunan , dari suatu jenis tertentu, memiliki dua karakteristik , yakni **posisi** di dalam himpunan dan **nilainya** sendiri.

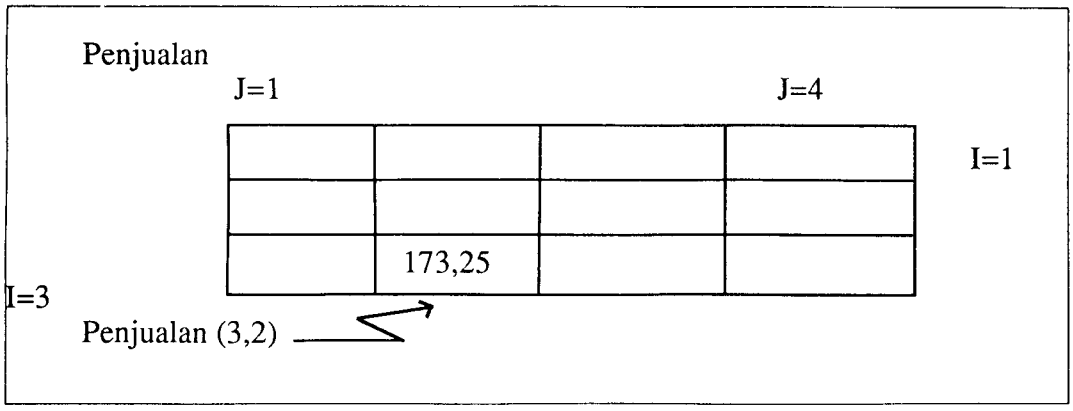
CONTOH 2-3.

Pandang NILAI adalah himpunan Integer satu dimensi. Untuk "memanggil" elemen ke K, kita harus memakai indeks NILAI(K). Misalkan himpunan tersebut terdiri dari 5 elemen, masing-masing adalah 17 , -3 , 2 , 4 , -5 seperti terlihat pada gambar 2-3. Diasumsikan bahwa elemen diberi indeks mulai dari 1 (bukan 0). Akan terlihat dengan jelas adanya suatu analogi antara sebuah himpunan dan sebuah tabel.

```
NILAI (1) ------> 17
NILAI (2) ------> -3
NILAI (3) ------>  2
NILAI (4) ------>  4
NILAI (5) ------> -5
```

GAMBAR 2.3

Untuk memperlihatkan Data penjualan sebuah perusahaan yang memiliki tiga wilayah pemasaran, dengan empat daerah per wilayah, dapat digunakan himpunan berdimensi dua atau Matriks berikut ini.



GAMBAR 2.4

Gambar grafikal dari himpunan diperlihatkan pada Gambar 2-4. Sub-Variabel dari himpunan yang pertama adalah sesuai dengan deretan angka (yang menunjukkan wilayah), sedangkan sub-Variabel yang kedua sesuai dengan jumlah kolom (menunjukkan daerah). Nampak bahwa sub-Variabel selalu harus merupakan nilai Integer, karena menjunjukkan posisi dari elemen himpunan. Di dalam berbagai Bahasa Pemrograman, mengharuskan himpunan dinyatakan secara jelas apabila dipakai dalam Statemen yang tidak dapat dilaksanakan.

Di dalam Bahasa Pemrograman yang menggunakan alokasi memori yang statis seperti FORTRAN atau COBOL, ruang memori dialokasikan kepada Data oleh perangkat Penterjemah . Di dalam bahasa yang menggunakan alokasi memori yang dinamis, seperti ALGOL atau PL/I, memori dapat dimintakan Data, atau didiamkan saja bila tidak dibutuhkan dalam pelaksanaan Program. Jika Storage harus dialokasikan sewaktu Program sedang diterjemahkan, yakni sebelum pelaksanaan Program, maka ruangan yang dibutuhkan untuk mengisi Data himpunan harus dituliskan di dalam deklarasinya. Dengan demikian, untuk menuliskan himpunan, karena tidak ada batas-batas yang ditetapkan oleh alokasi memori yang statis, maka kita cukup menyatakan jenis dan dimensinya saja; misal

```
Integer SALES (1:M,1:N)
```

dengan M dan N merupakan Variabel Integer,serta dengan menganggap awal subvariabel adalah 1, dapat kita tulis:

```
Integer SALES (M,N)
```

Di dalam bahasa yang menggunakan alokasi memori yang statis, maka barisan sub-bariabel harus ditulis secara konstan, misalnya SALES(1:3,1:5) atau SALES(3,5). Jika uraian dilakukan dalam sebuah Algoritma (bukan Program), tidaklah perlu untuk menuliskan Variabel secara konstan.

2.5. INPUT DAN OUTPUT

Operasi terhadap input yang mengalihkan suatu nilai secara tunggal menjadi Output, merupakan kegiatan utama Pemrograman tingkat tinggi. Hal yang rinci adalah dilaksanakan oleh perangkat keras sistem komputer dan sistem operasi. Untuk input, nilainya dimasukkan ke dalam komputer melalui alat input, seperti penggerak 'disket atau keyboard. Output dihasilkan oleh komputer melalui alat Output, misalnya printer atau layar terminal.

Bentuk Statemen yang umum, terdiri dari :

Daftar Variabel input
Daftar Variabel ouput.

Beberapa bahasa memakai istilah "Read" atau "Get" sebagai kata lain dari Input dan "Write" atau "Put" untuk Output.

Jenis media sumber Data harus diketahui secara jelas, tetapi tidak perlu disebut di dalam Statemen. Dengan Kondisi seperti itu, Data input dapat dianggap berasal dari Berkas input yang umum. Demikian pula Output adalah untuk Berkas Output yang umum pula. Dari alat input, Data disalurkan ke lokasi memori komputer. Lokasi tersebut berhubungan dengan nama Variabel sebagaimana tertulis di dalam Deklarasi. Selanjutnya, Data tersebut dapat digunakan secara simbolis dengan "memanggil" namanya. Konstanta sudah tertulis di dalam Program, sehingga tidak memerlukan input lagi. Jika suatu Statemen Output sudah terlaksana, maka diperoleh hasil berupa salinan nilai di dalam lokasi yang diminta oleh Statemen Output tersebut.
Untuk CONTOH 2-1, kita membutuhkan Statemen berikut ini :

Input NUM1,NUM2
Output JUMLAH.

Untuk melaksanakan input atau Output, seringkali kita ingin memperinci Data layoutnya. Beberapa Bahasa Pemrograman memiliki bentuk Statemen yang berfungsi untuk maksud tersebut di atas.

2.6. PENUGASAN (ASSIGNMENT)

Nilai dari suatu Variabel bisa didapat melalui pembacaan dari alat input, ataupun memakai suatu Statemen Penugasan, untuk dapat melaksanakan manipulasi Data.

2.6.1. BENTUK UMUM DARI STATEMEN PENUGASAN

Suatu **Penugasan** merupakan pemberian nilai kepada suatu Variabel. Dengan perkataan lain, jika suatu Penugasan sudah dilaksanakan, maka nilainya dimasukkan ke dalam lokasi memori, dengan nama seperti yang diberikan kepada Variabel tersebut.
Bentuk umum dari suatu Statemen Penugasan adalah :

Nama - Variabel ←——— nilai

Nilai, pada ruas kanan, dapat berupa Konstanta atau nilai suatu variabel lain, ataupun nilai dari suatu rumus. Panah merupakan operator pemberi Penugasan. Beberapa Bahasa Pemrograman memakai tanda = (dalam BASIC), atau := (dalam Pascal), sebagai pengganti ←———; misalnya :

```
x   <———   y + 2  atau
x     =    y + 2  atau
x     :=   y + 2
```

Jenis dari suku (nilai) di ruas kanan haruslah sama dengan jenis Variabel di ruas kiri. Jenis Variabel yang tertulis itu harus konsisten dengan jenis rumus atau operasinya. Jika terdapat suatu Variabel di ruas kanan, maka nilai Variabel tersebut harus ditetapkan terlebih dahulu, sebelum Statemen Penugasan dilaksanaan.

Kita dapat pula membentuk Statemen Majemuk yang terdiri dari beberapa Statemen Penugasan, sebagai contoh misalnya :

```
BEGIN

    Nama    := 'Astie Darmayantie';
    Tinggi  := 48;
    Berat   := 3.72;
    Normal  := True

END .
```

Pada contoh ini, Statemen memerintahkan agar berturut-turut diberikan nilai Astie Darmayantie kepada Variabel String, Nama; nilai 48 kepada Variabel Integer, Tinggi; nilai 3.72 kepada Variabel Real, Berat; dan nilai True kepada Variabel Logika, Normal.

2.6.2 EKSPRESI

Ekspresi dapat diartikan sebagai Rumus untuk mendapatkan suatu nilai. Ekspresi terdiri dari Konstanta dan / atau Variabel serta operator. Konstanta serta Variabel tersebut dinamakan *operand.* Jadi, **Ekspresi** adalah barisan simbol operand dan operator yang membentuk ketentuan untuk mengubah nilai Konstanta dan / atau Variabel. Ketentuan dalam Aritmetika akan mengatur pemakaian operator terhadap Konstanta atau Variabel untuk menghasilkan suatu jenis Ekspresi tertentu. Suatu Ekspresi dievaluasi dari kiri ke kanan, kecuali ditentukan lain.

Di bawah ini terdapat beberapa jenis Ekspresi sebagai berikut :

(a) Ekspresi Campuran.
Jika implementasi suatu bahasa tertentu memperbolehkan adanya Ekspresi campuran, maka setiap Ekspresi harus terdiri dari Variabel Integer atau real dan Konstanta saja. Ekspresi tersebut dibentuk sesuai dengan ketentuan umum dari Aritmetika.
Operator Aritmetika Unar digunakan untuk mencari nilai negatif. Operator Aritmetika binar terdiri dari penambahan (+), pengurangan (-), perkalian (*), pembagian (/), dan perpangkatan (**), biasanya dapat dikerjakan dengan tanda kurung.
Untuk mengerjakan prioritas hitungan berlaku ketentuan umum dalam Aritmetika. Jika memakai Integer untuk Ekspresi, maka pembagiannya diberi tanda DIV. Proses tersebut mengakibatkan pecahan dari nilai yang dibagi itu terbuang.

(b) Ekspresi Logika, menghasilkan salah satu dari dua nilai TRUE atau FALSE. Operator Logika di dalam Ekspresi tersebut (selanjutnya dibahas di dalam Bab 3) dimaksudkan untuk menyatakan suatu Kondisi guna menentukan pengendalian alur Program .

(c) Ekspresi Manipulasi String, yang beroperasi atas lingkup Konstanta dan Variabel String. Operator untuk memanipulasi String tersebut di antaranya penyambungan dua String, ataupun untuk salahsatu substring.

2.6.3. PELAKSANAAN STATEMEN PENUGASAN

Statemen Penugasan menghasilkan rangkaian kegiatan sebagai berikut :

a. Ekspresi, pada ruas kanan Statemen Penugasan, dievaluasi sesuai dengan ketentuan Aritmetika, dan akan menghasilkan suatu nilai tunggal yang jenisnya sesuai dengan jenis Ekspresi tersebut
b. Nilai tersebut ditempatkan pada lokasi yang sesuai dengan nama Variabel di ruas kiri Statemen Penugasan yang bersangkutan.

Di dalam CONTOH 2-1, Penugasan tersebut menjadi

```
BEGIN
JUMLAH := NUM1 + NUM2
END.
```

yang akan menyelesaikan penambahannya.

Jika kita ingin menggunakan suatu lokasi (yakni membaca isinya), maka nilai yang berada di dalamnya tidak akan berubah. Jika suatu lokasi diisi oleh suatu nilai baru (yakni menulis ke dalam lokasi) maka isi yang terdahulu otomatis terhapus. Dalam hubungan dengan Statemen Penugasan, hal tersebut berarti bahwa nilai Variabel yang sesungguhnya yang namanya berada di sebelah kiri sudah digantikan; sedangkan nilai dari variabel yang terdapat di sebelah kanan akan tetap.

Perhatikan :

(a) Jika Statemen dengan rumus

```
VAR 1 := VAR 2
```

dilaksanakan, maka nilai VAR2 disalin ke dalam lokasi yang sesuai dengan VAR1. Nilai dari VAR2 tidak berubah, ataupun menjadi terhapus.

(b) Untuk menukar nilai dari dua Variabel VAR1 dan VAR2, kita perlu memakai lokasi perantara ANTARA. Caranya adalah sebagai berikut :

```
BEGIN
ANTARA := VAR1;
VAR1   := VAR2;
VAR2   := ANTARA
END.
```

Berikut ini sebuah CONTOH Statemen Majemuk yang mengandung beberapa Statemen Penugasan, dan Statemen Input/OUTPUT. Apa hasilnya, jika nilai Variabel A diberikan sebesar 3 ?

```
BEGIN
    Read (A);
    B := 7;
    C := (A + B) * 2;
    D := ( C - A ) MOD B;
    C := C DIV D;
    WRITELN (A,B,C,D)
END .
```

JAWABNYA adalah
A=3 B=7 C=6 D=3

Keterangan:

Read (A)	:	berarti dimasukkan nilai A, yakni 3
B := 7	:	berarti nilai B adalah 7
C := (A + B) * 2	:	berarti nilai C adalah (3+7)*2 = 20
D := (C - A) MOD B	:	berarti nilai D adalah (20–3) MOD 7=3
C := C DIV D	:	berarti nilai C yang baru 20 DIV 3 = 6

2.7. ULASAN

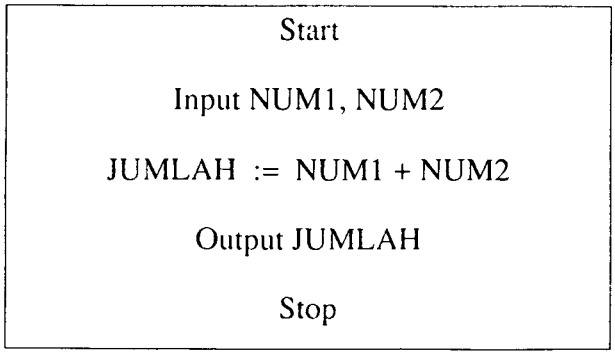
Seluruh Bahasa Pemrograman memberi peluang untuk menyisipkan **Ulasan** atau **Remark**, guna menjelaskan tentang Data dan Logika Program, serta Statemennya. Kita akan berasumsi bahwa suatu baris yang dimulai dengan tanda asteris (*), ataupun Tulisan di dalam tanda kurung kurawal { }, ataupun di antara tanda (* dan *) (dalam bahasa Pascal), merupakan Ulasan. Di dalam menuliskan sebuah Algoritma kita juga dapat memberi Ulasan. Ulasan memang sebaiknya diberikan, agar program dapat lebih mudah kita ikuti, dan kita mengerti.

2.8. RESENSI

Statemen dalam bahasa tingkat tinggi berikut ini :
(1) Statemen input/Output Statemen yang dilaksanakan

(2) Statemen Penugasan
(3) Deklarasi Statemen yang tidak dilaksanakan.

tidak berpengaruh pada arus pengendalian operasional di dalam Program, yakni urutan proses pelaksanaan Program. Statemen Deklarasi menguraikan isi satuan Data, sedangkan Statemen input dan Statemen Penugasan memberikan nilainya; Statemen Output mengeluarkan nilai untuk diketahui oleh pihakyang berkepentingan. Ulasan atau REMARK dapat disisipkan di mana-mana di dalam Program tanpa mempengaruhi pelaksanaan Program tersebut. Jika Statemen tersebut merupakan satu-satunya unsur pokok dari Bahasa Pemrograman, maka setiap Statemen di dalam suatu Program hanya dapat dilaksanakan sekali saja. Mengingat bahwa metode Algoritma dibuat dengan instruksi-instruksi yang bersifat ulang, diperlukan Statemen-Statemen tambahan supaya pengulangan dapat dilaksanakan secara selektif, dengan beberapa alternatif pelaksanaannya. Statemen-Statemen yang mempengaruhi arus pengendalian Program dibahas di dalam bab berikut ini. Setiap Bahasa Pemrograman juga memiliki unsur pembatas untuk menunjukkan akhir suatu Statemen dan kadang kala juga awal dari Statemen. Sebagai kesimpulan dari materi yang dibahas di atas, sampailah kita pada suatu diagram alur akhir untuk CONTOH 2-1 yang diperlihatkan pada gambar 2-5.



GAMBAR 2-5.

2.9 STRUKTUR PROGRAM PASCAL SECARA UTUH

Setelah pada pembicaraan yang lalu, kita bicarakan komponen pembentuk Program, secara terpisah-pisah, baiklah kali ini kita bahas Program dalam bahasa Pascal secara utuh, meskipun kita memulainya dari Program yang sangat sederhana.

Bahasa Pascal sendiri adalah suatu Bahasa Pemrograman Tingkat Tinggi, yang disusun oleh NIKLAUS WIRTH, seorang Guru Besar Matematika & Komputasi dari Technical University Zurich, Swiss. Nama Pascal diambil dari nama seorang Matematikawan Perancis, yang sangat besar jasanya dalam pengembangan Komputasi pada abad ke 17, yakni BLAISE PASCAL.

Kompilator Pascal yang pertama, dibuat pada tahun 1971 untuk keperluan belajar Pemrograman bagi para Mahasiswa, dan dijalankan pada komputer merek CDC (Control Data Corporation).

Bahasa Pascal ini dengan cepat berkembang dan menjadi sangat populer untuk segala pemakaian. Berbagai merek komputer berusaha menyediakan kompilator Pascal bagi komputernya, sehingga berbagai versi Pascal diciptakan orang. Semua itu bertolak dari Pascal Standar ciptaan Niklaus Wirth tersebut. Untuk kelompok mikrokomputer, dikenal adanya UCSD Pascal, Turbo Pascal, MS Pascal, dan beberapa jenis lagi. Turbo Pascal, yang merupakan ciptaan Borland, merupakan kompilator Pascal yang paling populer saat ini. Hal ini mungkin disebabkan karena sifatnya yang interaktif.

Program Pascal terdiri **Judul-Program** atau **Nama-Program**, dan **Blok-Program** atau **Badan-Program**. Blok-Program terdiri atas **Bagian-Deklarasi,** dan **Bagian-Statemen** atau **Bagian-Utama.** Dalam **Bagian-Deklarasi** dapat kita lakukan **Deklarasi Label, Deklarasi Konstanta, Deklarasi Variabel, Deklarasi Tipe, Deklarasi Prosedur** dan **Deklarasi Fungsi.**

Algoritma pelaksanaan Program, tertuang pada Bagian-Utama dalam bentuk barisan Statemen. Struktur dari suatu Program Pascal digambarkan pada gambar berikut ini :

1. JUDUL-PROGRAM

   PROGRAM nama _program;

2. BAGIAN DEKLARASI

– deklarasi Konstanta	CONST ...;
– deklarasi Tipe	TYPE ...;
– deklarasi variabel	VAR ...;
– deklarasi prosedur	PROCEDURE nama _ prosedur; ...;
– deklarasi fungsi	FUNCTION nama_fungsi; ...;

3. BAGIAN UTAMA PROGRAM

	begin
– pernyataan	...;
	...;
	...;
	...;
	...;
	end.

KETERANGAN

1. Judul-Program atau Nama-Program

   Pada Turbo Pascal, judul program sifatnya adalah **optional** (boleh ada atau boleh tidak). Jika ditulis dapat digunakan untuk memberi nama program dan daftar dari parameter tentang komunikasi program dengan lingkungannya yang sifatnya sebagai dokumentasi saja. Judul program ditulis pada awal program dimulai dengan reserved word **PROGRAM** dan diakhiri dengan **titik koma.**

2. Blok-Program atau Badan-Program :

   Bagian Deklarasi

   Seperti yang telah dijelaskan sebelum ini pada bagian **2.4.1.2** mengenai pendeklarasian, Bahasa Pascal mengharuskan kita untuk mendeklarasikan identifier yang kita gunakan dalam program. Identifier meliputi Konstanta, Variabel, Tipe, Label, Prosedur dan Fungsi.

   Deklarasi Konstanta

   Deklarasi Konstanta dan Variabel telah kita bahas pada bagian **2.4.1.2.** yang lalu.

   Deklarasi Tipe

   Berdasarkan cara pembentukannya, Bahasa Pascal memperbolehkan adanya *Tipe Data Standar (standard data types) dan Tipe Data Yang Didefinisikan oleh Pemakai (User-defined data Types).* Yang termasuk Tipe Data Standar adalah data dengan tipe Integer, real, char, string dan boolena (logika). Yang termasuk Tipe Data Yang Didefinisikan oleh Pemakai adalah scalar (sejumlah data disebutkan satu per satu) dan subrange. Selain itu, dikenal pula data dalam bentuk struktur larik (array), rekord (record), berkas (file) dan himpunan (set), penunjuk (pointer).
   Untuk mendeklarasikan Tipe Data Yang Didefinisikan oleh Pemakai bentuk umumnya adalah sebagai berikut :

TYPE

definisi tipe1;
definisi tipe2;
...
definisi tipen;

Sebagai contoh adalah sebagai berikut :

```
TYPE
        pecahan = real;
        logika = boolean;
        bulat = integer;
        huruf = string(25);
BEGIN
END.
```

Selanjutnya tipe data yang telah digunakan di atas dapat kita gunakan sebagai suatu tipe data tersendiri.

Deklarasi-Label

Deklarasi label biasanya diperlukan apabila kita menggunakan statement alih jalur tanpa syarat, yakni statemen GOTO. Hal ini akan dijelaskan pada Bab 3.

Deklarasi-Prosedur dan Fungsi akan dibahas secara lebih jelas pada Bab 4.

Sebagai contoh, pandang Program Pascal sederhana ini yang dimaksudkan untuk membaca dan mencetak sebuah Data.

```
PROGRAM CONTOH 1;
VAR
    a  : Integer;
BEGIN
    READLN (a);
    WRITELN ('Nilai Data adalah : ');
    WRITELN (a)
END.
```

Di sini, Nama-Program adalah CONTOH1. Program kita ini hanya memanfaatkan sebuah Variabel, yakni variabel a, yang telah dideklarasikan bertipe Integer. Di sini tidak ada Konstanta yang dideklarasikan. Demikian pula halnya, tidak ada tipe data yang kita bentuk secara khusus (tidak ada user-defined types data) dan tak ada Prosedur ataupun fungsi yang harus dipanggil. Juga, kita tidak menggunakan statemen GOTO, maka tidak diperlukan adanya LABEL.

Bagian-Statemen (Bagian-Utama) Program adalah bagian yang dibatasi oleh BEGIN dan END. Pada Program kita di atas, Bagian-Utama terdiri dari barisan 3 buah Statemen (yang di antara mereka dipisahkan dengan tanda titik koma).

Statemen **READLN** merupakan Statemen Input, yang dilaksanakan dengan cara : kita memasukkan Data melalui keyboard, begitu diminta oleh komputer. Statemen **READLN** ini serupa seperti Statemen INPUT pada bahasa BASIC.
Jadi, Statemen

```
READLN(a);
```

dalam Pascal adalah serupa dengan Statemen BASIC

```
INPUT A
```

Statemen **WRITELN** merupakan Statemen Output, yakni berfungsi untuk menampilkan nilai Variabel, ataupun menampilkan suatu Konstanta, baik numerik ataupun string. Statemen ini serupa dengan Statemen PRINT dalam BASIC.
Jadi, kedua Statemen

```
WRITELN ('Nilai Data adalah : '),
WRITELN (a)
```

serupa dengan

```
PRINT " Nilai Data adalah : "
PRINT A
```

dalam BASIC.

Untuk lebih jelasnya, berikut ini diberikan sebuah program BASIC yang setara dengan Program CONTOH1 kita, dalam Pascal tersebut.

```
10 INPUT A
20 PRINT " Nilai Data adalah : "
30 PRINT A
40 END
```

Perlu diperhatikan bahwa tulisan, atau kata Program merupakan Reserved-word, di samping VAR, INTEGER, BEGIN, **READLN, WRITELN**, ataupun END. Seperti telah kita ketahui, reserved-word adalah kata yang sudah didefinisikan oleh Pascal untuk tujuan tertentu yang spesifik. Reserved-word ini tidak boleh didefinisikan ulang oleh pemakai, sehingga tidak dapat dipergunakan sebagai pengenal (identifier). Berikut ini diberikan tabel reserved word pada Bahasa Pascal Standard.

Tabel Reserved Word

AND	**ARRAY**	**BEGIN**	**CASE**
CONST	**DIV**	**DO**	**DOWNTO**
ELSE	**END**	**FILE**	**FOR**
FORWARD	**FUNCTION**	**GOTO**	**IF**
IN	**LABEL**	**MOD**	**NIL**
NOT	**OF**	**OR**	**PACKED**
PROCEDURE	**PROGRAM**	**RECORD**	**REPEAT**
SET	**THEN**	**TO**	**TYPE**
UNTIL	**VAR**	**WHILE**	**WITH**

Selain dari reserved word di atas, Turbo Pascal masih memiliki tambahan reserved word berikut :

ABSOLUTE	**EXTERNAL**	**IMPLEMENTATION**	**INLINE**
INTERFACE	**INTERRUPT**	**SHL**	**SHR**
STRING	**UNIT**	**USES**	**XOR**

Berikut ini contoh program yang memanfaatkan Konstanta.

```
PROGRAM CONTOH2;
CONST
    sekolah   : 'SMA';
VAR
    a   : Integer;
BEGIN
    READ (a);
    WRITE (sekolah, a);
END.
```

Di sini kita deklarasikan sebuah Konstanta, yakni Konstanta String bernama sekolah. Kalau program CONTOH2 di atas dilaksanakan, dengan memberi nilai 8 kepada Variabel a, diperoleh hasil:

```
SMA 8
```

Statemen **READ** pada hakikatnya sama dengan **READLN**, hanya saja pada **READ,** kursor seusai Statemen akan berada di samping kedudukan terakhir. Pada **READLN**, begitu menyelesaikan Statemen, kursor akan meloncat ke baris di bawahnya. Hal serupa dapat diterangkan untuk **WRITE** dan **WRITELN**. Bandingkan dengan Statemen BASIC, PRINT A; dan PRINT A saja.

CONTOH3 merupakan sebuah Program untuk menghitung luas dan keliling lingkaran, yang diberikan jejarinya. Program dilengkapi pula dengan Remark atau Ulasan seperlunya (serupa dengan Statemen REM atau ', pada BASIC). Pada Program Pascal, isi Ulasan ditulis di antara kurung kurawal { }, atau di antara tanda (* dan *).

```
PROGRAM CONTOH3;

{ Program untuk menghitung luas dan keliling lingkaran }
CONST
    pi   =   3.14;{ Konstanta pi }
VAR
    a    :   Integer; { a adalah jejari lingkaran }
    l    :   real; { l adalah luas lingkaran }
    k    :   real; { k adalah keliling lingkaran }
```

```
BEGIN
    READLN (a);
    l   :=  pi * a * a;
    k   :=  pi * 2 * a;
    WRITELN ('Luas =',l);
    WRITELN ('Keliling =',k)
END.
```

Pada program di atas, kita menghitung luas lingkaran menggunakan rumus luas = (phi)a^2 dan keliling lingkaran dengan rumus keliling = 2 (phi)a. Secara Pascal rumus tersebut ditulis sebagai Statemen penugasan l:= pi*a*a dan k:=pi*2*a. Dalam Pascal konstanta Phi merupakan konstanta yang **'built in'** (sudah disediakan oleh Pascal) nilainya adalah 3.14 .....
Selain dari itu, terdapat beberapa lagi konstanta 'built in', yakni :

Maxint bertipe integer dengan nilai 32767

Maxlongint bertipe integer dengan nilai 2147483647

True bertipe boolean dengan nilai true

False bertipe boolean dengan false.

Konstanta 'built-in' ini tidak perlu dideklarasikan.

Seperti telah disebutkan dahulu mengenai tipe data, dalam Turbo Pascal Tipe Integer maupun Real masih diperinci lagi atas beberapa Tipe berdasarkan jangkauan nilainya.

Tipe Data Numerik Integer terdiri atas :	
Tipe	Jangkauan nilai
Byte	0 ... 255
Shortint	-128 ... 127
Integer	-32768 ... 32767
Word	0 ... 65535
Longint	-2147483648 ... 2147483647

Tipe Data Numerik Real berkisar dari 1E-38 sampai dengan 1E+38 dengan **mantissa** (banyak angka di belakang titik desimal) signifikan sampai dengan 11 digit. Nilai konstan numerik real menempati memori sebesar 6 byte. Penulisan .137 dalam Turbo Pascal, haruslah ditulis 0.137, demikian pula halnya 12. adalah salah, seharusnya adalah 12.0

IEEE (Institute of Electrical and Electronic Engineers) Standar 754 untuk artihmatika floating-point, yang terdiri dari tipe data single, double, extended dan comp sesuai adalah berikut :

Tipe	Jangkauan nilai	Digit signifikan
Single	1.5 x 10E-45 ... 3.4 x 10E38	7 - 8
Double	5.0 x 10E-324 ... 1.7 x 10E308	15 - 16
Extended	1.9 x 10E-4951 ... 1.1 x 10E4932	19 - 20
Comp	-2E+63+1 ... 2E+63-1	19 - 20