4 TIPE DATA, NAMA, NILAI

4.1 TIPE

Pada umumnya, program komputer bekerja dengan memanipulasi objek(data) di dalam memori. Objek(data) yang akan diprogram bermacam-macam jenis atau tipenya misalnya nilai numerik(angka), karakter(huruf), kumpulan karakter, dll.

Suatu tipe menyatakan jenis data yang akan dimanipulasi dalam program, gunanya untuk mendefinisikan objek yang akan diprogram. Suatu tipe diacu dari namanya. Nilai-nilai yang dicakup oleh tipe tersebut dinyatakan dalam domain nilai. Tipe data dikelompokkan menjadi tipe dasar dan tipe bentukan

4.1.1 Tipe dasar

Tipe yang dapat langsung dipakai(angka-angka atau karakter) karena sudah didefinisikan sebelumnya oleh pemroses bahasa

1. **Bilangan Bulat**

   Bilangan yang tidak mengandung pecahan desimal.

   - **Nama Tipe : Integer**
   - **Domain Nilai :**

     Secara teoritis tak terbatas dari -∞ s.d +∞. Pada algoritma dapat dibatasi tergantung kebutuhan untuk objek, misalnya untuk jam [0..23]. Dalam implementasinya pada bahasa pemrograman, tipe integer punya rentang nilai terbatas untuk menghemat memory.
   - **Contoh nilai :** 300, 0, -1000, 113010038, -24
   - **Contoh objek :** Nim, Jam, Menit, Detik

2. **Bilangan Riil**

   Bilangan yang mengandung pecahan desimal

   - **Nama Tipe : *Real***
   - **Domain Nilai :**

     Secara teoritis tak terbatas dari -∞ s.d +∞. Ditulis dengan titik desimal
   - **Contoh nilai :** 2.8 , -0.39, 4.24 , 57.567, -102.00
   - **Contoh objek :** Nilai ujian

3. **Bilangan Logika**

   - **Nama Tipe : *Boolean***
   - **Domain Nilai :** Benar(*True*--1) atau Salah(*False*--0)
   - **Konstanta :** *True* dan *False*

4. **Karakter**

Karakter tunggal yang diapit oleh tanda petik satu.

- **Nama Tipe : char**
- **Domain Nilai :**
  - Huruf alfabet ('a'..'z' dan 'A'..'Z')
  - Tanda baca ('!', '?' , ',' , '.')
  - Angka '0' , '1' ,..., '9'
  - Karakter khas seperti '#' , '&' , '%' , '@' , '*' , dll
- **Contoh nilai :** 'l' , 'p' , '+' , '6', 'A'
- **Contoh objek :** Jenis Kelamin, Indeks nilai

4.1.2 Tipe bentukan

Tipe yang dibentuk(dan diberi nama) dari tipe dasar atau dari tipe lain yang sudah dikenal, bahkan dapatt didefinisikan sendiri oleh pemrogram.

Macam tipe bentukan :

1. **String(kumpulan karakter)**

Deretan karakter dengan panjang tertentu.

- **Nama Tipe : *String***
- **Domain Nilai :** Satu atau lebih karakter yang diapit oleh tanda petik tunggal
- **Contoh nilai :** 'Apa kabar','Teknik Informatika','A234', 'Ramayana', '123'
- **Contoh objek :** Nama, Alamat

2. **Tipe bentukan dari tipe data dasar atau tipe bentukan lain : Kata Kunci type**

Deklarasi kamus data : `type nama_tipe_bentukan : tipe_data`

Contoh : membuat sebuah tipe data baru bernama BilBulat yang memiliki domain nilai yang sama dengan tipe integer

```
type BilBulat : integer;
```

3. ***Record***

Record tersusun dari satu atau lebih field. Tiap field menyimpan data dengan tipe tertentu

field 1	*field 2*	*field 3*	…	*field n*

Deklarasi kamus data :

```
type Nama_Record : record < nama_field1  :tipe_field1,
                            nama_field2  :tipe_field2,
                            ....
                            nama_fieldn  : tipe_fieldn>
```

Contoh :

a. Titik dalam koordinat kartesian dinyatakan sebagai (x,y) dengan x adalah nilai absis dan y adalah nilai ordinat. Kita dapat menyatakan titik sebagai record dengan (x,y) sebagai field

x	*y*

```
type Titik : record < x : real,  y : real >    atau
type Titik : record< x , y : real >
```

b. Definisi tipe terstruktur yang mewakili tanggal dalam kalender Masehi. *dd* sebagai tanggal, *mm* sebagai bulan, *yy* sebagai tahun

dd	*mm*	*yy*

```
type Tanggal : record <dd : integer,    {1..31}
                       mm : integer,  {1..12}
                       yy : integer   {>0} >
```

c. NilMhs adalah tipe terstruktur yang menyatakan nilai ujian seorang mahasiswa untuk mata kuliah yang diambil(MK)

Nim	*NamaMhs*	*KodeMK*	*Nilai*

```
type NilMhs : record < Nim : integer,
                       NamaMhs : string,
                       KodeMK : string,
                       Nilai : char >
```

d. Tipe terstruktur untuk jadwal kereta api yang terdiri dari nomor kereta(NoKA), kota asal(KotaAsal), kota tujuan(KotaTujuan), jam berangkat(JamBerangkat), jam tiba(JamTiba)

NoKA	*KotaAsal*	*KotaTujuan*	*JamBerangkat*	*JamTiba*

```
type JadwalKA : record <NoKA : string,
                        KotaAsal : string,
                        KotaTujuan : string,
                        JamBerangkat : Jam,
                        JamTiba : Jam >
```

4.2 NAMA

Nama digunakan mengidentifikasi objek dan mengacu objek tersebut.. Dalam sebuah teks algoritmik, objek yang diberi nama bisa berupa :

- Modul program, Algoritma
- variabel
- konstanta
- type
- fungsi
- prosedur

Karena adanya bermacam-macam nama tersebut, maka dalam suatu teks algoritma dikenal nama program, nama variabel, nama konstanta, nama fungsi, nama prosedur, nama type.

Setiap bahasa pemrograman memiliki aturan masing-masing untuk mendefinisikan nama(panjang maks nama, perbedaan huruf besar dan kecil) tetapi dalam algoritma batasan pendefinisian nama tidak seketat pada bahasa pemrograman. Syarat-syarat penggunaan sebuah nama pada algoritma:

- Pemilihan nama harus interpretatif(disesuaikan dengan objek yang diidentifikasi) dan tidak menimbulkan kerancuan
- Nama harus unik dalam sebuah algoritma/program
- Nama tidak boleh dipisahkan oleh spasi
- Tidak *case sensitive* (huruf besar dan kecil tidak dibedakan)
- Tidak boleh mengandung symbol khusus
- Nama harus dideklarasikan pada bagian tertentu supaya dapat dipakai.

1. Nama Algoritma

Digunakan untuk mengidentifikasi sebuah program atau algoritma, dideklarasikan pada bagian **Judul algoritma**

Contoh : `Algoritma Luas_Lingkaran,  Algoritma Kelulusan_Mhs`

2. Nama Peubah(*variabel*)

Tempat penyimpan data/informasi/nilai yang isinya dapat diubah selama eksekusi program berlangsung. Setiap variabel mempunyai tiga atribut, yaitu **nama**, **tipe**, dan **nilai**. Nama variabel dan tipe datanya dideklarasikan pada bagian **Kamus.** Sedangkan nilai yang disimpan dalam variabel didefinisikan pada bagian deskripsi algoritma.

Bentuk umum deklarasi variabel adalah :

```
nama_variabel : tipe data
```

Contoh :

```
Kamus
  nama : string              {variabel nama bertipe string}
  nim: integer               {variabel nim bertipe integer/bilangan bulat}
  jns_kelamin : char         {variabel jns_kelamin bertipe karakter}
  rata, nilai_uts, nilai_uas, nilai_tugas : real; {variabel dengan nama rata, nilai_uts,
                                                   nilai_uas, nilai_tugas bertipe sama yaitu real}
```

3. Nama Tetapan(*konstanta*)

Tempat penyimpan data/informasi/nilai yang isinya tidak dapat diubah selama pelaksanaan program. Nama, tipe, dan nilai Konstanta dideklarasikan pada bagian Kamus. Untuk mendefinisikan konstanta harus memakai kata kunci **const** dan konstanta harus langsung diisi dengan sebuah nilai tertentu.

Bentuk umum deklarasi konstanta adalah :

const nama_konstanta : tipe = nilai

Contoh :

```
Kamus
  const phi : real = 3.14
  const Nmaks : integer = 200
  const password : string = 'abcd'
```

4. Nama Tipe bentukan

Nama tipe bentukan disini berarti nama tipe bentukan yang dibuat/didefinisikan oleh perancang program. Nama tipe bentukan dideklarasikan pada bagian Kamus

Nama tipe bentukan tidak dapat langsung digunakan di dalam bagian deskripsi algoritma, tetapi sebelumnya harus mendeklaraikan variabel yang bertipe bentukan tersebut

Contoh :

```
Kamus
    type Karakter : char
    type Titik : record < x : real; y : real >
    type Jam : record <hh : integer,  {0..23}
                       mm : integer,  {0..59}
                       ss : integer   {0..59} >
    Indeks : karakter
    T : Titik
    J : Jam
```

Dari contoh di atas telah didefinisikan tipe data baru bernama karakter dan dua buah tipe record yang bernama Titik dan Jam. Nama karakter, Titik, dan Jam tidak bisa langsung digunakan pada bagian Deskripsi algoritma tetapi harus mendeklarasikan variabel baru. Pada contoh di atas dideklarasikan variabel Indeks yang bertipe karakter, variabel T bertipe Titik dan variabel J yang bertipe Jam. Indeks, T, dan J inilah yang bisa digunakan pada bagian deklarasi program.

5. Nama Fungsi

Bagian Judul Fungsi(nama fungsi, parameter) dideklarasikan pada bagian Kamus

Contoh :

```
Kamus
    Function Penjumlahan(a , b : integer) → integer
    {mengembalikan hasil penjumlahan antara dua bilangan}
```

6. Prosedur

Bagian Judul Prosedur(nama prosedur , parameter) dideklarasikan pada bagian Kamus

Contoh :

```
Kamus
    Procedure TUKAR(input/output A , B : integer)
    {mempertukarkan nilai A dan B}
```

Contoh-Contoh Pendefinisian/Delarasi Nama Di Dalam Bagian Kamus :

```
Kamus
   {nama konstanta}
   const phi = 3.14
   const Nmaks = 200
   const password = 'abcd'
  {nama tipe}
   type karakter : char
   type Titik : record < x : real; y : real >
   type Jam : record <hh : integer,  {0..23}
                      mm : integer,  {0..59}
                      ss : integer   {0..59} >
   type NilMhs : record <Nim : integer,
                         NamaMhs : string,
                         KodeMK : string,
                         Nilai : char >

   {nama variabel}
   luasL : real
   nama : string
   indeks : karakter
   ketemu : boolean
   J : Jam
   T : Titik
   Nilai : NilMhs

   {nama fungsi}
   Function Penjumlahan(a , b : integer) → integer
   {mengembalikan hasil penjumlahan antara dua bilangan}
   function CARI <input x : integer> → Boolean
   {mencari nilai x,bila ketemu maka true,bila tidak maka
   false}
   {nama prosedur}
   procedure HITUNG_TITIK_TENGAH(input P1 :Titik, input P1
   :Titik, output Pt :Titik)
   {menghitung nilai titik tengah dari sebuah garis dengan
   ujung-ujung Px dan Py}
```

4.3 NILAI

Nilai/Harga adalah besaran dari tipe data yang sudah dikenal. Nilai dapat berupa konstanta yang dipakai langsung, isi yang disimpan oleh variabel atau konstanta, hasil perhitungan suatu ekspresi, atau hasil yang dikirim suatu fungsi.

Algoritma pada dasarnya adalah proses memanipulasi nilai. Nilai dapat dimanipulasi dengan cara : mengisi nilai ke dalam variabel, menuliskan nilai ke piranti keluaran, diacu dari suatu nama untuk perhitungan/ekspresi

1. Pengisian Nilai

Suatu nama konstanta secara otomatis akan mempunyai harga tetap yang terdefinisi(sudah ditentukan) pada saat nama konstanta dideklarasikan dalam kamus sehingga nama konstanta dapat langsung digunakan dalam program. Tetapi tidak demikian halnya dengan nama variabel. Suatu nama variabel dapat digunakan dalam ekspresi program jika harganya telah terdefinisi. Ada dua cara untuk mengisi nama variabel dengan harga/nilai :

- ***Assignment*** **: ←**

Assignment adalah instruksi untuk menyimpan harga pada suatu nama variabel. Dengan pemberian harga ini, harga lama yang disimpan tidak lagi berlaku, yang berlaku adalah harga paling akhir yang diberikan.

Nilai/Harga yang dimasukkan ke dalam nama variabel bisa berupa nilai tetap, nilai dari variabel lain, atau ekspresi :

```
Deskripsi Algoritma
  nama_var1 ← nama_var2  {harga dari nama variabel2 disalin ke nama variabel1}
  nama_var ← konstanta   {harga dari nama konstanta disalin ke nama variabel}
  nama_var ← ekspresi    {hasil perhitungan ekspresi diisikan ke nama variabel}
  nama_var ← nama_fungsi {nilai yang dikembalikan fungsi diisikan ke nama variabel}
```

dengan syarat :

- Bagian kiri dan bagian kanan tanda assignment (←) bertipe sama
- nama_var1 dan nama_var (bagian kiri tanda ←) harus merupakan nama variabel, tidak boleh nama konstanta, type, fungsi, atau prosedur
- nama yang tertulis di bagian kanan tanda assigment (←)boleh berupa nama variabel, nama fungsi, nama konstanta
- semua nama yang dipakai dalam assignment tidak bleh berupa nama type atau prosedur

Contoh :

```
Kamus
   k, suhu1, suhu2, Total : integer
   ketemu : boolean          J : Jam
   Jarak : real              NamaKota : string
Deskripsi Algoritma
   k ← 10
   ketemu ← false
   Jarak ← 34.8
   NamaKota ← 'Tasikmalaya'
   Suhu1 ← 40
   Suhu2 ← 30
   Total ← Suhu1 + Suhu2
   Suhu1 ← Suhu2
   Total ← k*20+14
```

- **Pembacaan Nilai dari Piranti Masukan**

Selain dengan assignment, suatu nilai dapat diisikan ke suatu nama variabel melalui pembacaan nilai tersebut dari piranti masukan(keyboard, mouse, scanner, dsb). Disebut "dibaca" karena arah dari pengisian harga yaitu seakan-akan komputer "membaca" nilai yang diberikan pengguna.

Bentuk Umum :

```
Deskripsi Algoritma
  read(nama_variabel)          {membaca sebuah nilai}
  read(list nama_variabel)     {membaca lebih dari satu nilai}
```

Contoh :

```
Kamus
    Nim : integer
    Nama : string
    Indeks : char
    Nilai : real
Deskripsi Algoritma
   read(Nim)
   read(Nama)
   read(Nilai, Indeks)
```

2. Penulisan nilai ke piranti keluaran

Suatu nilai/harga yang disimpan dalam memori komputer harus dapat dikomunikasikan ke dunia luar untuk diinterpretasikan oleh pemakai program. Dalam hal ini, nilai harus dapat dituliskan ke suatu piranti keluaran, misalnya layar, printer.

Bentuk Umum :

```
Deskripsi Algoritma
  write(nama_variabel) {menuliskan isi nama_variabel ke piranti keluaran}
  write(konstanta)     {menuliskan konstanta / isi nama_konstanta ke piranti keluaran}
  write(ekspresi)      {menuliskan harga hasil perhitungan ekspresi ke piranti keluaran }
  write(list-nama)     {menuliskan semua harga sesuai urutan penulisannya }
```

dengan syarat :

- list-nama adalah satu atau lebih nama : boleh nama variabel, nama konstanta, atau hasil pemanggilan fungsi
- nama-nama dalam list-nama tidak boleh berupa nama type atau nama prosedur
- nama yang dituliskan sudah terdefinisi harganya. Jika nama_variabel sudah didefinisikan dengan *assignment* atau instruksi *read*

Contoh :

```
Kamus
  const pass : string = 'abce'
  A, B : integer
  Nilai : real
Deskripsi Algoritma
  Nilai ← 92.7
  read(A , B)
  write(pass, Nilai)
  write('Teknik Informatika')
  write(100)
  write(A + B)
  write((A + B)/2*10)
```

4.4 OPERATOR DAN EKSPRESI

Operator adalah lambang-lambang yang biasa dilibatkan dalam program untuk melakukan suatu operasi atau manipulasi. Misalnya untuk perkalian, penjumlahan, perbandingan, dll. Sedangkan ekspresi adalah suatu "rumus perhitungan" yang terdiri dari operan dan operator. Operan harus mempunyai harga, karena itu dapat berupa **konstanta**, **nama variabel**(yang dipakai dalam perhitungan adalah harga yang dikandung nama variabel), hasil pengiriman suatu **fungsi**, atau merupakan suatu **ekspresi**

Contoh Ekspresi :

$a \leftarrow b + c - 2$

Pada ekspresi ini, a, b, dan c merupakan nama variabel yang berperan sebagai operand sedangkan simbol ←, + dan – merupakan operator. Dalam hal ini variabel a diisi dengan hasil penjumlahan b dan c dikurangi 2.

Jenis –jenis operator :

1. Operator Perbandingan

Operator perbandingan digunakan untuk membandingkan dua operand. Operand yang dibandingkan bisa bertipe bilangan bulat, karakter, real, boolean, atau string. Ekspresi yang menggunakan operator perbandingan akan menghasilkan **nilai boolean(true atau false).**

Operator	Operasi	Contoh Ekpresi	Hasil
=	Sama dengan	a := 6 = 9	a = false
≠	Tidak sama dengan	a := 7 ≠ 5	a = true
<	Lebih kecil dari	a := 4 < 6	a = true
>	Lebih besar dari	a := 10 > 1	a = true
≤	Lebih kecil atau sama dengan	a := 8 ≤ 4	a = false
≥	Lebih besar atau sama dengan	a := 3 ≥ 1	a = true

2. Operator aritmatika

Operator aritmatika hanya dapat dikenakan pada operand bertipe bilangan bulat atau bilangan real. Ekspresi yang menggunakan operator ini pun hanya akan menghasilkan nilai bilangan bulat atau real

Operator	Operasi	Hasil	Contoh Ekpresi	Hasil
+	Jumlah	Integer/Real	x ← 8 + 13 x ← 4.3 + 2	x = 21 x = 6.3
-	Kurang	Integer/Real	x ← 15 – 2 x ← 2.1 – 1.1	x = 13 x = 1.0
*	Kali	Integer/Real	x ← 5 * 6 x ← 2.0 * 1.1	x = 30 x = 2.2
/	Bagi	Real	x ← 6 / 4	x = 1.5
div	Pembagian bilangan bulat	integer	z ← 7 div 2	z = 3
mod	Sisa pembagian bilangan bulat	integer	z ← 7 mod 2	z = 1
^	Pangkat	integer/real	z ← 2 ^ 3	z = 8

3. Operator logika

Operator ini dikenakan pada operand bertipe boolean dan ekspresinya akan menghasilkan nilai boolean(true atau false)

Operator	Arti	Hasil
not	negasi	boolean
and	dan	boolean
or	atau	boolean
xor	exclusive OR	boolean

Hasil operator not, and, or, dan xor untuk berbagai kombinasi kondisi

A	B	not A	not B	A and B	A or B	A xor B
true	true	false	false	true	true	false
true	false	false	true	false	true	true
false	true	true	false	false	true	true
false	false	true	true	false	false	false

Contoh penggunaan operator pada ekspresi :

```
Kamus
     Gaji_Total, Gaji_Pokok, Potongan : real
     HBagi,HSisa : integer
     k, l, m, n: boolean;
Deskripsi
     k := true;
     l := false;
     read(Gaji_Pokok,Potongan)
     Gaji_Total ← Gaji_Pokok – Potongan
     HBagi ← (5 * 7) div 3
     HSisa ← (5 * 7) mod 3
     m := (k or l) and l;
     n := ((6 >= 8) and (9 <> 1)) or (3 < 7);
     write(HBagi, HSisa, Gaji_Total, m , n);
```