

Handout Pemrograman Komputer (Turbo Pascal)



Nur Hadi Waryanto, S.Si., M.Eng.

**Laboratorium Komputer Jurusan Pendidikan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Negeri Yogyakarta
2010**

Kata Pengantar

Puji syukur penulis panjatkan kepada Allah SWT atas nikmat kesehatan, kekuatan dan hidayah-Nya yang diberikan kepada penulis sehingga Handout Pemrograman Komputer ini dapat diselesaikan

Handout Pemrograman Komputer ini disusun untuk dapat dipergunakan sebagai panduan pelaksanaan kegiatan perkuliahan dan praktikum mata kuliah Pemrograman Komputer pada Program Studi Pendidikan Matematika dan Program Studi Matematika Jurusan Pendidikan Matematika FMIPA Universitas Negeri Yogyakarta.

Penulis menyadari masih terdapat kekurangan dalam penulisan Handout Pemrograman Komputer ini. Oleh karena itu saran dan masukan dari berbagai pihak baik dosen maupun mahasiswa penulis harapkan sebagai bahan untuk pengembangan dan perbaikan demi kesempurnaan Handout Pemrograman Komputer ini. Ucapan terima kasih penulis sampaikan kepada semua pihak yang telah membantu dalam penyusunan Handout Pemrograman Komputer ini.

Yogyakarta, Oktober 2010

Penulis

Daftar Isi

Kata Pengantar	ii
Daftar Isi	iii
Bab 1 Bahasa Pemrograman dan Algoritma	1
Bab 2 Struktur Program Pascal	7
Bab 3 Tipe Data	9
Bab 4 Operasi Input Output	11
Bab 5 Pernyataan If dan Case (Percabangan)	14
Bab 6 Operator	20
Bab 7 Pengulangan Proses (Looping)	23
Bab 8 ARRAY	27
Bab 9 Record	33
Bab 10 Prosedur	37
Bab 11 Fungsi dan Rekursi	47
Bab 12 Sorting	51
Daftar Pustaka	54

Bab 1

Bahasa Pemrograman dan Algoritma

Bahasa (*language*)

Bahasa adalah suatu sistem untuk berkomunikasi. Bahasa tertulis menggunakan simbol (yaitu huruf) untuk membentuk kata. Dalam ilmu komputer, bahasa manusia disebut bahasa alamiah (*natural languages*), dimana komputer tidak bisa memahaminya, sehingga diperlukan suatu bahasa komputer.

Bahasa pemrograman (*programming language*)

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

Generasi bahasa pemrograman:

- Generasi I: *machine language*
- Generasi II: *assembly language* : *Assembler*
- Generasi III: *high-level programming language*: C, PASCAL, dsb.
- Generasi IV: 4 GL (*fourth-generation language*): SQL

Bahasa Tingkat Rendah (*low-level language*)

Merupakan bahasa assembly atau bahasa mesin. Lebih dekat ke mesin (*hardware*), dimana high-level programming languages dekat pada bahasa manusia.

Bahasa Mesin (*machine language*)

Bahasa mesin merupakan representasi tertulis machine code (kode mesin), yaitu kode operasi suatu mesin tertentu. Bahasa ini bersifat khusus untuk mesin tertentu dan "dimengerti" langsung oleh mesin, sehingga pelaksanaan proses sangat cepat.

Bahasa Assembly (*assembly language*)

Bahasa rakitan (*assembly language*) merupakan notasi untuk menyajikan bahasa mesin yang lebih mudah dibaca dan dipahami oleh manusia. Bahasa ini sudah menggunakan simbol alpabet yang bermakna (*mnemonic*). Contoh "MOV AX 1111", pindahkan ke register AX nilai 1111.

Bahasa Tingkat Tinggi (*high-level language*)

Adalah bahasa pemrograman yang dekat dengan bahasa manusia, kelebihan utama dari bahasa ini adalah mudah untuk di baca, tulis, maupun diperbaharui, sebelum bisa dijalankan program harus terlebih dahulu di-compile. Contoh Ada, Algol, BASIC, COBOL, C, C++, FORTRAN, LISP, dan Pascal, dsb.

Pada generasi bahasa pemrograman terakhir sekarang ini, kedua cara interpretasi dan kompilasi digabungkan dalam satu lingkungan pengembangan terpadu (*IDE = integrated development environment*).

Fourth-Generation Language (4GL)

Lebih dekat ke bahasa manusia dibandingkan dengan high-level programming languages. Biasanya dipakai untuk mengakses database. Contoh perintah pada bahasa SQL: FIND ALL RECORDS WHERE NAME IS "JOHN"

Bahasa Pemrograman untuk tujuan tertentu

Tabel 1. Bahasa Pemrograman untuk tujuan tertentu

Jenis Program	Bahasa Terbaik	Bahasa Terburuk
Data terstruktur	ADA, C /C++, PASCAL	Assembler, BASIC
Proyek cepat	BASIC	PASCAL, ADA, Assembler
Eksekusi cepat	Assembler, C	BASIC, Interpreter Language
Kalkulasi matematika	FORTTRAN	PASCAL
Menggunakan memori dinamis	PASCAL, C	BASIC
Lingkungan bermemori terbatas	BASIC, Assembler, C	FORTTRAN
Program real-time	ADA, Assembler, C	BASIC, FORTTRAN
Manipulasi string	BASIC, PASCAL	C
Program mudah dikelola	PASCAL, ADA	C, FORTTRAN

Compiler dan Interpreter

Compiler adalah suatu program yang menterjemahkan bahasa program (*source code*) ke dalam bahasa objek (*object code*). **Compiler** menggabungkan keseluruhan bahasa program dikumpulkan kemudian disusun kembali.

Compiler memerlukan waktu untuk membuat suatu program yang dapat dieksekusi oleh komputer. Tetapi, program yang diproduksi oleh **Compiler** bisa berjalan lebih cepat dibandingkan dengan yang diproduksi oleh **Interpreter**, dan bersifat independen.

Interpreter berbeda dengan **Compiler**, **Interpreter** menganalisis dan mengeksekusi setiap baris dari program tanpa melihat program secara keseluruhan. Keuntungan dari **Interpreter** adalah dalam eksekusi yang bisa dilakukan dengan segera. Tanpa melalui tahap kompilasi, untuk alasan ini **Interpreter** digunakan pada saat pembuatan program berskala besar.

Tabel 3. Perbedaan Compiler dan Interpreter

No	Interpreter	Compiler
1	Menerjemahkan instruksi per Instruksi	Menerjemahkan secara keseluruhan
2	Tidak menghasilkan objek program	Menghasilkan objek program
3	Tidak menghasilkan <i>executable</i> program karena langsung dijalankan pada saat program Diinterpretasi	Menghasilkan <i>executable</i> program, sehingga dapat langsung dijalankan.
4	Proses interpretasi terasa cepat, karena tiap-tiap instruksi langsung dikerjakan dan dapat dilihat hasilnya	Proses kompilasi lama, karena sekaligus menterjemahkan seluruh instruksi program
5	<i>Source</i> program terus dipergunakan karena tidak dihasilkan <i>executable</i> program	<i>Source</i> program sudah tidak dipergunakan lagi untuk mengerjakan program

Tipe Pemrograman

1. Pemrograman terstruktur

Pemrograman terstruktur adalah cara pemrosesan data yang terstruktur. Terstruktur dalam: analisa, cara dan penulisan program.

2. Bahasa pemrograman prosedural – terstruktur

Bahasa pemrograman prosedural adalah bahasa pemrograman yang mendukung pembuatan program sebagai kumpulan prosedur. Prosedur-prosedur ini dapat saling memanggil dan dipanggil dari manapun dalam program dan dapat menggunakan parameter yang berbeda-beda untuk *setiap* pemanggilan. Prosedur adalah bagian dari program untuk melakukan operasi-operasi yang sudah ditentukan dengan menggunakan parameter tertentu. Bahasa pemrograman terstruktur adalah pemrograman yang mendukung abstraksi data, pengkodean terstruktur dan kontrol program terstruktur.

Kontrol program terstruktur:

1. Runtun - urut (*sequence*)
2. Pilihan (*selection*)
3. Pengulangan (*repetition - loop*)

ALGORITMA

Perencanaan dan perancangan program komputer juga disebut pembuatan algoritma. Beberapa definisi algoritma:

- a. kumpulan urutan perintah yang menentukan operasi-operasi tertentu yang diperlukan untuk menyelesaikan suatu masalah ataupun mengerjakan suatu tugas".
- b. logika, metode dan tahapan (urutan) sistematis yang digunakan untuk memecahkan suatu permasalahan.

Algoritma - dan tentu program yang baik, bercirikan:

- a. Tepat sasaran, benar, sederhana, standar dan efektif : memenuhi spesifikasi pekerjaan dan bekerja sesuai tujuan
- b. *Flexible* dan *portable*: - *Flexible* untuk dikembangkan lebih lanjut - *Portable* untuk digunakan pada berbagai sistem dan mesin
- c. Bersih dari kesalahan sistem ataupun logic
- d. Murah: - Efisien dalam penggunaan piranti memori dan penyimpanan lainnya. - Cepat waktu pelaksanaannya.
- e. Didokumentasi dengan baik untuk pengoperasian, pemeliharaan dan pengembangan.
- f. Algoritma merupakan pemberian (*description*) pelaksanaan suatu proses. Sebuah proses dikerjakan oleh pemroses mengikuti algoritma yang sudah dibuat. Algoritma merupakan urutan langkah instruksi yang logis. *Setiap* langkahinstruksi mengerjakan suatu tindakan aksi.
- g. Logis, terstruktur dan sistematis
- h. Semua operasi terdefinisi
- i. Semua proses harus berakhir *setelah* sejumlah langkah dilakukan

- j. Ditulis dengan bahasa yang standar dengan format pemrograman agar mudah untuk diimplementasikan dan tidak menimbulkan arti ganda

Aturan Penulisan Teks Algoritma

Tidak ada notasi yang baku dalam penulisan teks algoritma. Algoritma bukanlah program yang harus mengikuti aturan-aturan tertentu. Meski demikian, algoritma dituliskan mendekati gaya bahasa pemrograman umumnya. Misal, tulis nilai X dan Y, dituliskan dalam algoritma sebagai `write(X,Y)`. Perhatikan dalam notasi `write(X,Y)` ini hanya memerintahkan penyajian nilai X ke piranti keluaran (output). Dalam notasi itu juga tidak memasalahkan format ataupun bentuk-bentuk tampilan lainnya, seperti dicetak dalam satu baris X dan Y, pemakaian pemisah antara X dan Y menggunakan koma atau spasi. Hal-hal yang bersifat teknis ini baru dipikirkan waktu penulisan program. Algoritma adalah bebas bahasa pemrograman.

Teks Algoritma

Mengikuti alur konsep pemrograman prosedural, suatu teks algoritma disusun dalam tiga bagian, yaitu:

- a. Bagian kepala algoritma,
- b. Bagian deklarasi,
- c. Bagian deskripsi algoritma.

Setiap bagian disertai dengan penjelasan atau dokumentasi tentang maksud pembuatan teks. Bagian penjelasan diawali dan diakhiri dengan simbol { dan }.



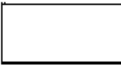





Algoritma NAMA_ALGORITMA { Penjelasan tentang algoritma yang menguraikan secara singkat hal-hal yang dilakukan oleh algoritma }

DEKLARASI { Semua nama yang digunakan, meliputi nama-nama: tipe, konstanta, variabel. Juga nama sub-program dinyatakan di sini }

DESKRIPSI { Semua langkah atau aksi algoritma dituliskan di sini }

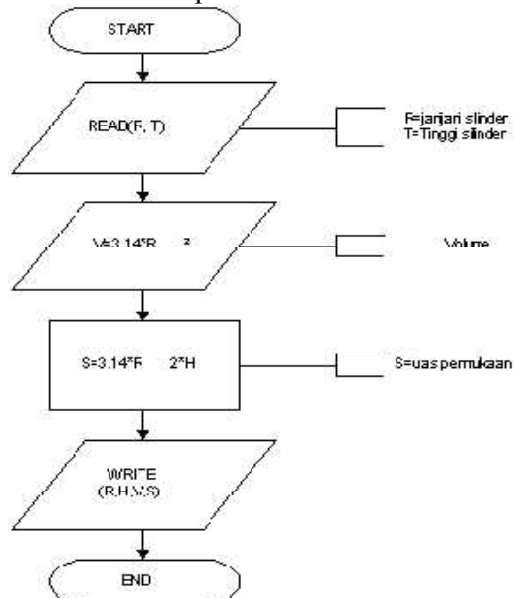
Diagram Alir (*Flow chart*)

Merupakan bentuk grafis/visual dari algoritma Bentuk umum dari simbol-simbol dalam diagram alir:

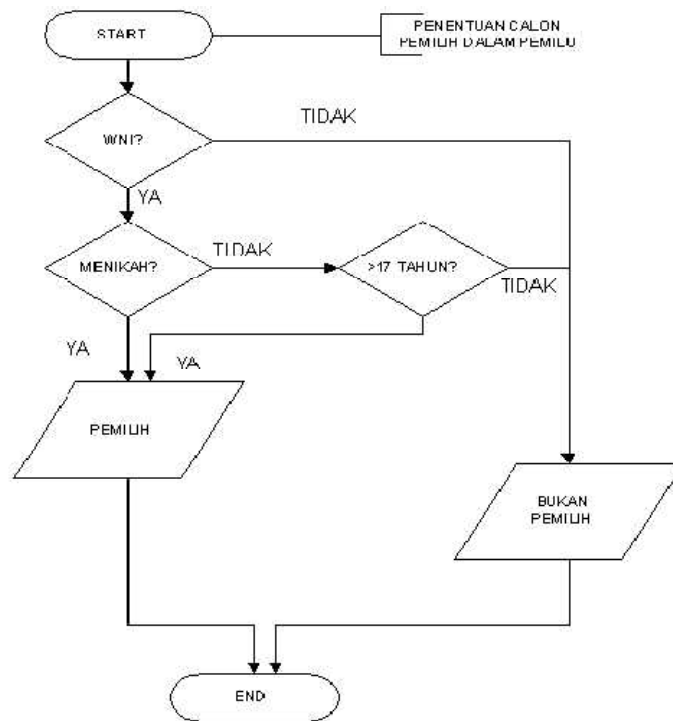
	Simbol untuk mulai (start) atau akhir (end) program
	Simbol untuk pembacaan (read) data atau penulisan hasil (write) pada layar
	Simbol untuk suatu proses terhadap data pada program
	Simbol untuk suatu pernyataan pilihan (optional) pada program.
	Simbol untuk penghubung antar aktifitas.
	Konektor, Simbol untuk memutus aktivitas karena keterbatasan media kertas.
	Sub program
	komentar

Contoh pemakaian flowchart:

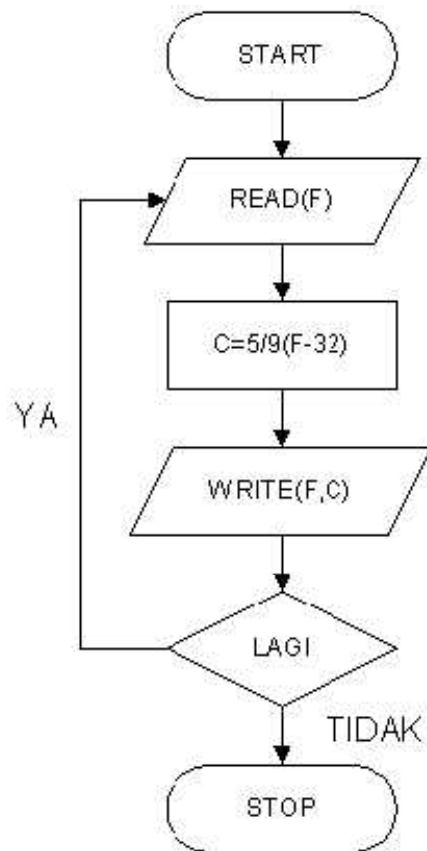
- **Sequential (berurutan)**
perhitungan volume dan luas permukaan silinder



- **Selection/Branching Structure (Struktur pemilihan)**



- **Repetition/Looping Structure (Struktur pengulangan)**



Bab 2 Struktur Program Pascal

Struktur Program Pascal

Struktur dari suatu program pascal terdiri dari sebuah judul program dan suatu blok program atau badan program. Blok program dibagi lagi menjadi dua bagian, yaitu : bagian deklarasi dan bagian pernyataan. Secara ringkas, struktur suatu program pascal dapat terdiri dari :

1. Judul program
2. Blok program
 - a. Bagian deklarasi
 - deklarasi label
 - deklarasi konstanta
 - deklarasi tipe
 - deklarasi variable
 - deklarasi prosedur
 - deklarasi fungsi
 - b. Bagian pernyataan

Judul Program

Judul program ini digunakan untuk memberi nama program dan sifatnya optional. Jika ditulis harus terletak pada awal dari program dan diakhiri dengan titik koma (;).

Contoh penulisan judul program :

```
PROGRAM coba ;  
PROGRAM gaji (input , output) ;  
PROGRAM latihan_1 ;
```

Bagian Deklarasi

Bagian ini menjelaskan secara rinci semua data yang akan digunakan pada suatu program. Dalam penulisannya tidak boleh sama dengan katakata cadangan (*reserved words*) dan selalu diakhiri dengan titik koma (;).

Deklarasi Label

Digunakan jika pada penulisan program akan menggunakan statemen GOTO (untuk meloncat ke suatu statemen tertentu).

Deklarasi Konstanta

Deklarasi ini digunakan untuk mengidentifikasi data yang nilainya sudah ditentukan dan pasti, tidak dapat dirubah dalam program.

Deklarasi Tipe

Deklarasi ini digunakan untuk menyebutkan tipe *setiap* data yang akan digunakan pada program Pascal. Tipe data menentukan jangkauan nilai yang mungkin dari data yang digunakan.

Pascal menyediakan beberapa macam tipe data, yaitu :

1. Tipe data sederhana, terdiri dari :
 - a. Tipe data standar : *integer, real, char, string, boolean.*
 - b. Tipe data didefinisikan pemakai : *enumerated* atau *scalar, subrange*
2. Tipe data terstruktur : *array, record, file, set.*
3. Tipe data penunjuk

Deklarasi variabel

Deklarasi ini berisi data-data yang bisa berubah-ubah nilainya di dalam program. Deklarasi variabel harus diletakkan *setelah* deklarasi tipe (jika ada).

Unit

Suatu unit adalah kumpulan dari konstanta, tipe-tipe data, variable, prosedur dan fungsi-fungsi. Tiap-tiap unit tampak seperti suatu program Pascal yang terpisah. Unit standar sudah merupakan kode mesin (sudah dikompilasi), bukan kode sumber Pascal lagi dan sudah diletakkan di memori pada waktu menggunakan pascal. Untuk menggunakan suatu unit, harus diletakkan suatu anak kalimat Uses diawal blok program, diikuti oleh daftar nama unit yang digunakan.

Unit CRT

Digunakan untuk memanipulasi layar teks (*windowing*, peletakkan cursor dilayar, color untuk teks, kode extended keyboard dan lainnya). Unit standar crt hanya dapat digunakan oleh program yang digunakan dikomputer IBM PC, IBM AT, IBM PS/2 atau yang kompatibel dengannya.

Bagian Pernyataan / Terproses

Bagian yang akan diproses dan terdapat dalam suatu blok yang diawali dengan BEGIN dan diakhiri dengan END, setiap statemen yang merupakan instruksi program diakhiri dengan tanda titik koma (;).

Bentuk umumnya adalah sebagai berikut :

```
BEGIN
.....
statemen;
.....
END.
```

Contoh program

```
Program Cobal;
uses wincrt;
Label satu;
CONST a = 15;      { selalu menggunakan tanda = }
TYPE nyata = real; { selalu menggunakan = }
VAR  b : integer;
     c : nyata;    { selalu menggunakan : }

Begin
  b:=1;
  c:=a+b;
  writeln ('Hasil = ',c:3:2);
  writeln(c);
  WRITELN('Universitas');
  GOTO SATU;
  WRITELN('Negeri');
  satu:
  WRITELN('Yogyakarta');
End.
```

Output Program

```
Hasil : 16.00
Universitas
Yogyakarta
```

BAB 3

Tipe Data

Tipe Data Sederhana

Tipe data yang sering dipakai oleh program, meliputi: *integer* (bilangan bulat), *real* (bilangan pecahan), *char* (alphanumerik dan tanda baca), dan *boolean* (logika).

Tipe Data Integer

Tipe Data	Ukuran Tempat	Rentang Nilai
Byte	1 byte	0 s/d +255
Shortint	1 byte	-28 s/d +127
integer	2 bytes	-32768 s/d 32767
Word	2 bytes	0 s/d 65535
Longint	4 bytes	2147483648 s/d 2147483647

Tipe Data Real

Tipe Data	Ukuran Tempat	Rentang Nilai
real	6 bytes	2.9×10^{-39} s/d 1.7×10^{38}
single	4 bytes	1.5×10^{45} s/d 3.4×10^{38}
double	8 bytes	5.0×10^{-324} s/d 1.7×10^{308}
extended	10 bytes	3.4×10^{-4932} s/d 1.1×10^{4932}
comp	8 bytes	-9.2×10^{18} s/d 9.2×10^{18}

Tipe Data Char

Tipe data ini menyimpan karakter yang diketikkan dari keyboard, memiliki 266 macam yang terdapat dalam tabel *ASCII* (*American Standard Code for Information Interchange*). Contoh: 'a' 'B' '+', dsb. Yang perlu diingat bahwa dalam menuliskannya harus dengan memakai tanda kutip tunggal. Jenis data ini memerlukan alokasi memori sebesar 1(satu) byte untuk masing-masing data.

Tipe Data Boolean

Merupakan tipe data logika, yang berisi dua kemungkinan nilai: *TRUE* (benar) atau *FALSE* (salah). Turbo Pascal for Windows memiliki tiga macam jenis ini yaitu: Boolean, WordBool, dan LongBool. Tipe boolean memakai memori paling kecil, sedangkan WordBool dan LongBool dipakai untuk menulis program yang sesuai dengan lingkungan Windows.

Tipe Data	Ukuran Tempat
Boolean	1 byte
WordBool	2 byte
Longbool	3 byte

Sebagai bilangan ordinal boolean *TRUE* mempunyai nilai 1(satu), sedangkan *FALSE* nilainya adalah 0(nol).

Tipe data Tersruktur

Tipe ini terdiri atas : array, record, *set*, dan file. String adalah tipe data jenis array, tetapi karena string memiliki kekhasan tersendiri sebagai array dari karakter maka penulis perlu memberikan penjelasan tersendiri. Sedangkan untuk array, record, dan file perlu dijelaskan dalam bab yang lain karena agak banyak hal-hal yang perlu dibahas.

Tipe Data String

Merupakan suatu data yang menyimpan array (larik), sebagai contoh 'ABCDEF' merupakan sebuah konstanta string yang berisikan 6 byte karakter. Ukuran Tempat untuk tipe data ini adalah 2 s/d 256 byte, dengan jumlah elemen 1 s/d 255. String dideklarasikan dengan string [konstanta] atau string. Bila ukuran string tidak didefinisikan maka akan banyak memakan ruang, karena ukuran string menyesuaikan dengan defaultnya. Misalkan var kata: string [20]; atau var kata: string; karena string merupakan array dari karakter. Maka kata[1] merupakan karakter pertama dari string, kemudian kata[2], merupakan elemen kedua, dst.

Tipe Data Set

Sebuah *set* merupakan suatu himpunan yang berisi nilai (anggota). *Set* merupakan Tipe data yang khusus untuk Pascal. *Set* dalam pemrograman sangat mirip dengan himpunan dalam matematika.

contoh:

```
A = { 1, 2, 3, 4, 5 }
Syntax: set of contoh:
type Digits = set of 0..9;
Letters = set of 'A'..'Z';
type Day = (Sun, Mon, Tue, Wed, Thu, Fri, Sat);
CharSet = set of Char;
Digits = set of 0..9;
Days = set of Day;
```

Kita tidak bisa menulis atau membaca isi dari *set*, tetapi kita bisa melakukan operasi yang lain dengan data yang ada pada *set* (mis. relasional).

```
Program contoh_set;
Uses wincrt;
type hari = (ahad, sen, sel, rab, kam, jum, Sab);
var semua_hari : set of hari;
    hari_kerja : set of sen .. jum;
    hari_ini : hari;

begin
hari_ini:=sen;
if hari_ini in hari_kerja then writeln('HARI INI HARI
KERJA')
else
writeln('HARI LIBUR');
end.
```

Tipe Data Pointer

Tipe data pointer merupakan variabel khusus yang berisi suatu address (alamat) di lokasi lain didalam memory. Suatu variabel yang points(menunjuk) ke sesuatu sehingga disebut pointer. Ada dua macam pointer:

- *typed* (tertentu): merupakan pointer yang menunjuk pada tipe data tertentu pada variable.
- *Generic* (umum): merupakan pointer yang tidak menunjuk pada tipe data tertentu pada variable.

Bab 4

Operasi Input Output

Statemen adalah perintah untuk pengerjaan program pascal. Statemen terletak di bagian deklarasi statemen dengan diawali oleh kata cadangan BEGIN dan diakhiri dengan kata cadangan END. Akhir dari *setiap* statemen diakhiri dengan titik koma [;]. Statemen statemen dalam bahasa Pascal terdiri dari pernyataan yang berupa fungsi dan prosedur yang telah disediakan sebagai perintah standar Turbo Pascal.

Perintah Output

Perintah write dan writeln digunakan untuk menampilkan output di layar. Perintah write digunakan untuk mencetak pada baris yang sama dari beberapa argument. Perintah writeln digunakan untuk mencetak pada satu baris tersendiri dari beberapa argument. Perintah writeln yang tidak diikuti argument hanya mencetak baris kosong.

Contoh :

<pre>Write ('Bahasa '); Write ('Pascal');</pre> <p>Hasilnya setelah di cetak. Bahasa Pascal.</p>	<pre>Writeln ('Bahasa '); Writeln ('Pascal');</pre> <p>Hasilnya setelah di cetak. Bahasa Pascal</p>
--	---

Perintah Input

Perintah input **Read/Readln** digunakan untuk memasukkan [*input*] data lewat keyboard ke dalam suatu variabel. Perbedaan perintah read dan readln sama dengan perbedaan perintah write dan writeln. Perintah read akan membaca masukan dari keyboard tanpa memindahkan posisi kursor *setelah* pembacaan, sedangkan perintah readln akan membaca masukan sekaligus memindahkan posisi kursor.

Contoh :

```
Program input;
Uses Crt;
Var nama, NIM : String;
Begin
    Clrscr;
    Writeln ('masukkan nama dan NIM ');
    Writeln ('-----');
    Write ('nama anda : ');
    Readln (nama);
    Writeln ('NIM anda : '); Readln (NIM);
End.
```

Bila dijalankan hasilnya adalah:

```
masukkan nama dan NPM
-----
nama anda : ( di input )
NIM anda : ( di input )
```

Dengan menggunakan komputer, praktekan beberapa contoh program dibawah ini :

1. Program luas_PersegiPanjang

```
uses wincrt;
var panjang, lebar, luas : integer;
```

```

BEGIN
    write('Panjang = '); readln(panjang);
    write('Lebar = '); readln(lebar);
    luas:= panjang *lebar;
    writeln('Luas = ', luasPSP);
End.

```

2. Program Simpan_Nilai;

```

uses wincrt;
var a,b,c,d:integer;
Begin
write('Nilai a = ');readln(a);
write('Nilai b = ');readln(b);
write('Nilai c = ');readln(c);
d:=a+b+c;
writeln('Nilai d = ',d);
a:=b+d;
b:=d;
d:=a-d;
writeln;
writeln('Nilai a = ',a);
writeln('Nilai b = ',b);
writeln('Nilai d = ',d);
End.

```

3. Program Menghitung_Jarak;

```

Uses WinCrt;
var
x1,x2,y1,y2:integer;
d:real;
begin
    Writeln('Program Menghitung Jarak Titik A dan B');
    Writeln('=====');
    Writeln;
    Write('Masukan Nilai A (X1): ');readln(x1);
    Write('Masukan Nilai B (X2): ');readln(x2);
    Write('Masukan Nilai A (Y1): ');readln(y1);
    Write('Masukan Nilai B (Y2): ');readln(y2);
    d:=sqrt(sqr(x2-x1)+sqr(y2-y1));
    Writeln;
    Writeln('Jadi Jarak Titik A ke B Adalah: ',d:4:2);
end.

```

4. Program Konversi_Suhu;

```

Uses WinCrt;
var f,c:real;
begin
    Writeln('Program Konversi Fareinheit Ke Celcius');
    Writeln('=====');
    Writeln;
    Write('Masukan Suhu dalam Farenheit: ');readln(f);
    c:=5/9*(f-32);
    Writeln;
    Writeln('Jadi Suhu Dalam Celcius Adalah: ',c:4:2);
end.

```

5. Program Menukar_Nilai;

```

Uses WinCrt;
var A,B:integer;
Begin
    Writeln('Program Menukar Nilai A Menjadi B');
    Writeln('=====');
    Writeln;
    Write('Masukkan Nilai A: ');readln(A);
    Write('Masukkan Nilai B: ');readln(B);

```

```
Writeln;  
A:=A-B;  
B:=B+A;  
A:=B-A;  
Writeln;  
Writeln('Hasil A=',A,' B=',B);  
End.
```

Soal Latihan

1. Buatlah program untuk membuat data pribadi/biodata (input dan output bebas, misal nama, alamat, tanggal lahir dll)
2. Buatlah program untuk Mengubah derajat temperatur, dari derajat Celcius ke derajat Fahrenheit, Reamur dan kelvin (input : derajat celciusderajat Celcius diinput)
3. Buatlah program untuk mengkonversi waktu.
Input : jam, menit, detik.
Output : detik
4. Buatlah program untuk menukar nilai. (*buat program yang berbeda dengan contoh program no. 5*)

Bab 5

Pernyataan If dan Case

Statemen IFTHEN....

Bentuk struktur If...Then..... adalah sebagai berikut :

```
If Kondisi Then Statemen
```

Statemen ini digunakan untuk mengendalikan jalannya suatu program berdasarkan suatu *kondisi* atau syarat yang diberikan. Ungkapan adalah *kondisi* yang diseleksi oleh statemen If. Bila *kondisi* yang diseleksi terpenuhi, maka *statemen* yang mengikuti Then akan diproses, sebaliknya bila *kondisi* tidak terpenuhi, maka yang akan diproses *statemen* berikutnya. Jika *kondisi* bernilai benar (*TRUE*) maka *statemen* akan dikerjakan. Jika *kondisi* bernilai salah maka (*FALSE*) maka *statemen* tidak akan dikerjakan.

Contoh Program

```
Program Lulus;  
Uses wincrt;  
Var Nilai : Real;  
  
Begin  
Write ('Jumlah Nilai :');  
Readln (nilai);                               {Pemasukan  
data}  
If nilai > 60 Then                             {seleksi kondisi variabel  
nilai}  
Writeln('Lulus');                             {Dilaksanakan jika nilai  
lebih End.                                   besar dari 60}
```

Statemen IF.....THEN.....ELSE.....

Bentuk statemen IF...THEN...ELSE.... merupakan pengembangan dari struktur IF.....THEN..... Bentuk statemen tersebut adalah sebagai berikut :

1. Bentuk Pertama

```
IF kondisi THEN  
    statemen1  
ELSE  
    statemen2;
```
2. Bentuk Kedua

```
If kondisi Then  
    Begin  
    .....  
    .....  
    End  
Else  
    Begin  
    .....  
    .....  
    End;
```

Perintah ini berguna untuk memilih statemen mana yang akan dikerjakan oleh komputer berdasarkan kondisi/syarat yang diberikan. Jika *kondisi* bernilai benar (*TRUE*)

maka *statemen1* akan dikerjakan. Jika *kondisi* bernilai salah maka (*FALSE*) maka *statemen2* yang akan dikerjakan.

Statemen1 dan statemen2 dapat berupa suatu blok statemen tersendiri yang diapit oleh BEGIN...END seperti dalam bentuk kedua. di atas. Apabila kata ELSE digunakan, maka statemen antara THEN dan ELSE tidak diakhiri dengan tanda titik koma(;). Jika diakhiri dengan tanda titik koma (;), maka statemen *setelah* kata ELSE tidak akan dikerjakan atau penyeleksian kondisi akan berhenti pada statemen sebelum kata ELSE. Jika tidak menggunakan kata ELSE (bentuk statemen IF....THEN....) maka statemen *setelah* then harus diakhiri dengan tanda titik koma (;).

Contoh Program

1. Program Lulus1;
Uses wincrt;
Var Nilai : Real;

Begin
Write ('Jumlah Nilai :');
Readln (nilai);
If nilai > 60 Then
 Writeln('Lulus')
Else
Writeln('Tidak lulus')
End.
2. Program Lulus2;
Uses wincrt;
Var Nilai : Real;

Begin
Write ('Jumlah Nilai :');
Readln (nilai);
If nilai > 60 Then
 Writeln('Lulus');
Else
Writeln('Tidak lulus')
End.
3. Program IF_ELSE_DEMO;
uses wincrt;
var angka,tebakan : integer;

begin
angka := 2;
writeln('Tebak angka antara 1 dan 10'); readln(tebakan);
if angka = tebak then
 writeln('Tebakan anda benar, Selamat!')
 else writeln('Maaf, Tebakan anda salah.')
end.

Statemen CASE

Statemen Case mempunyai bentuk sebagai berikut

1. **Bentuk Case....Of**
Case *kondisi* of

```

        nilai1 : statemen1;
        nilai2 : statemen2;
        nilai3 : statemen3;
        .....
End;

```

2. Bentuk Case.....Of...Else...

```

Case kondisi of
    nilai1 : statemen1;
    .....
    nilaiN : statemenN;
Else
    statemenLain;
End;

```

Struktur Case - Of mempunyai suatu ungkapan logika yang disebut dengan selector dan sejumlah statemen yang diawali dengan suatu label permasalahan (case label) yang mempunyai tipe sama dengan selector. Statement yang mempunyai case label yang bernilai sama dengan case label yang bernilai sama dengan nilai selector akan diproses sedang statemen yang lainnya tidak.

Nilai kondisi harus suatu kondisi yang nilainya berjenis ordinal (dapat diurutkan). Nilai1, nilai2, dst dapat berupa sebuah nilai berjenis ordinal (sesuai dengan jenis kondisi) atau beberapa nilai yang dapat dipisahkan dengan tanda titik koma.

Contoh Program

```

1. Program nilai;
uses wincrt;
Var nilai : Char ;
Begin
Write ('Nilai Numerik yang didapat A,B,C,D atau E :');
Readln (nilai);
Case nilai Of
    'A' : write('Sangat baik');
    'B' : write('Baik');
    'C' : write('Cukup');
    'D' : write('Kurang');
    'E' : write('Sangat kurang');
End;
End.

```

```

2. Program pilihan;
uses wincrt;
var pil : integer;

begin
writeln(' Silakan pilih');
writeln('(1) Bentuk IF...Then...');
writeln('(2) Bentuk IF...Then...Else....');
writeln('(3) entuk Case....OF');
write('Pilihan = ');readln(pil);
clrscr;
case pil of
    1 : writeln('Anda memilih bentuk IF...Then...');

```

```

        2 : writeln('Anda memilih bentuk
        IF...Then...Else...');
        3 : writeln ('Anda memilih bentuk Case..Of');
    end;
End.

```

3. Program pilihan;

```

uses wincrt;
var pil : integer;

begin
writeln(' Silakan pilih');
writeln('(1) Bentuk IF...Then...');
writeln('(2) Bentuk IF...Then...Else....');
writeln('(3) entuk Case...OF');
write('Pilihan = ');readln(pil);
clrscr;
case pil of
    1 : writeln('Anda memilih bentuk IF...Then...');
    2 : writeln('Anda memilih bentuk
    IF...Then...Else...');
    3 : writeln ('Anda memilih bentuk Case..Of');
else
    writeln('Pilihan antara 1-3');
end;
End.

```

Dengan menggunakan komputer, praktekan beberapa contoh program dibawah ini :

1. Program Maksimum;

```

uses wincrt;
var A, B, C : integer;

begin
writeln('Masukkan tiga angka dengan spasi'); readln( A,
B, C );
if A >= B then
    begin
        if A >= C then writeln( A,' adalah terbesar')
        else
            writeln( C,' adalah terbesar')
        end
    else if B >= C then writeln( B,' adalah terbesar') else
        writeln( C,' adalah terbesar')
    end.

```

2. Program Konversi_Waktu;

```

Uses WinCrt;
var j,m,d,dm, sisa, sisa1:integer;

begin
Writeln('Program Konversi Waktu 1');
Writeln('=====');
Writeln;

```

```

Write('Masukkan Jumlah Detik : ');readln(dm);
if (dm/3600)>0 then
  begin
    j:=dm div 3600;
    sisa:=dm-(j*3600);
  end
else
  begin
    j:=0;
    sisa:=dm;
  end;
  if (sisa/60)>0 then
    begin
      m:=sisa div 60;
      sisa:=sisa-(m*60);
    end
  else
    begin
      m:=0;
      sisa:=sisa;
    end;
d:=sisa;
Writeln;
Writeln('Hasil => ',j,' jam ',m,' menit ',d,' detik');
end.

```

3. Program Menghitung_Selisih_Waktu;

```

Uses WinCrt;
Var j,m,d,h,j1,m1,d1,h1,hj,hm,sl,sisa,sisal:longint;

Begin
Writeln('Program Menghitung Selisih Waktu');
Writeln('=====');
Writeln;
Write('Waktu ke-1 jam : ');readln(j);
Write('Waktu ke-1 Menit : ');readln(m);
Write('Waktu ke-1 Detik : ');readln(d);
Writeln('=====');
Write('Waktu ke-2 jam : ');readln(j1);
Write('Waktu ke-2 Menit : ');readln(m1);
Write('Waktu ke-2 Detik : ');readln(d1);
h:=(j*3600)+(m*60)+d;
h1:=(j1*3600)+(m1*60)+d1;
sl:=h1-h;
if (sl/3600)>0 then
  begin
    begin
      hj:=sl div 3600;
      sisa:=sl-(hj*3600);
    end
  else
    begin
      hj:=0;
      sisa:=sl;
    end;
if (sisa/60)>0 then

```

```

begin
  hm:=sisa div 60;
  sisal:=sisa-(hm*60);
end
else
begin
  hm:=0;
  sisal:=sisa;
end;
Writeln;
Writeln('Selisih Waktu: ',hj,' jam ',hm,' Menit ',sisal,'
Detik');
End.

```

Soal Latihan

1. Buatlah program untuk menentukan apakah suatu bilangan bulat merupakan bilangan bulat positif, bilangan bulat negatif atau bilangan nol.
2. Buatlah program untuk menentukan akar-akar persamaan kuadrat dengan menggunakan rumus ABC.
3. Berdasarkan data berikut ini:

A = TVRI	D = ANTV
B = RCTI	E = INDOSIAR
C = SCTV	

Buatlah program yang meminta masukan huruf saluran TV, kemudian program menampilkan nama stasiun penyiarannya. Bila yang huruf yang dimasukkan tidak diantara A sampai dengan E, berikan komentar 'Nomor saluran salah'.

4. Buatlah program untuk mengkonversi nilai akhir mahasiswa, dengan ketentuan Nilai Akhir (NA) sebagai berikut :

$$\begin{aligned}
 81 - \dots &= A \\
 71 - 80 &= B \\
 61 - 70 &= C \\
 51 - 60 &= D \\
 \dots - 50 &= E
 \end{aligned}$$

Input => Nilai Presensi (NP), Nilai Keaktifan (NK), Nilai Tugas (NT), Nilai USIP1 (NUS1), Nilai USIP 2 (NUS2), Nilai UAS (NUAS).

Output => NP, NK, NT, Rata-rata NUS, NUAS, NA, nilai huruf

$$NA = \frac{10NP + 5NK + 20NT + 30\left(\frac{NUS1 + NUS2}{2}\right) + 35NUAS}{100}$$

BAB 6 Operator

Beberapa operator yang disediakan oleh PASCAL: *Aritmatika, Boolean, Relasional, Set*.
Operator Aritmatika

Operator	Operasi	Tipe Operand	Tipe Hasil Operasi
+	Penjumlahan	Integer, real	Integer, real
-	Pengurangan	Integer, real	Integer, real
*	Perkalian	Integer, real	Integer, real
/	Pembagian	Integer, real	Integer, real
div	Pembagian	integer, integer	integer
mod	Sisa pembagian	integer, integer	integer

Operator Boolean (Logika)

Operator	Operasi	Tipe Operand	Tipe Hasil Operasi
not	negasi	boolean	boolean
and	logika 'and'	boolean	boolean
or	logika 'or'	boolean	boolean
xor	logika 'xor'	boolean	boolean

Operator Relasional

Operator	Operasi	Tipe Operand	Tipe Hasil Operasi
=	Sama dengan	tipe sederhana, string, pointer dan <i>set</i>	boolean
<>	Tidak sama dengan	tipe sederhana, string, pointer dan <i>set</i>	boolean
<	Lebih kecil dari	tipe sederhana, string	boolean
>	Lebih besar dari	tipe sederhana, string	boolean
<=	Lebih kecil atau =	tipe sederhana, string	boolean
>=	Lebih besar atau =	tipe sederhana, string	boolean

Operasi pada *Set*

Operasi Relasional pada *Set*

Ada empat perbandingan relasional yang diperkenankan pada *set*.

Operator	Operasi	Tipe Operand	Tipe Hasil Operasi
=	Sama dengan	<i>Set, set</i>	boolean
<>	Tidak sama dengan	<i>Set, set</i>	boolean
<=	Lebih kecil atau =	<i>Set, set</i>	boolean
>=	Lebih besar atau =	<i>Set, set</i>	boolean

Operasi Logika pada *Set*

Ada tiga operasi logika pada *set*.

Operator	Operasi	Tipe Operand	Tipe Hasil Operasi
+	Union	<i>Set</i>	<i>Set</i>
-	Difference	<i>Set</i>	<i>Set</i>
*	Intersection	<i>Set</i>	<i>Set</i>

Fungsi Matematik Standar dalam PASCAL

Nama Fungsi	Deskripsi	Tipe Argumen	Tipe Hasil Operasi
abs	absolute value	real/integer	real/integer
arctan	arctan (radian)	Real/integer	real
cos	cosine (radian)	real/integer	real

sin	sin (radian)	real/integer	real
exp	fungsi Perpangkatan e	real/integer	real
ln	ln	real/integer	real
Round	Pembulatan terdekat	real	integer
sqr	kuadrat	real/integer	real/integer
sqrt	Akar kuadrat	real/integer	real
Trunc	Pembulatan ke bawah	real/integer	integer

Dengan menggunakan komputer, praktekan beberapa contoh program dibawah ini :

1. Program boolean1;

```
uses wincrt;

begin
writeln('A>a = ', 'A'>'a');
writeln('a>A = ', 'a'>'A');
writeln('6>3 = ', 6>3);
writeln('-3>4 = ', -3>4);
end.
```

2. program boolean1;

```
uses wincrt;

begin
writeln('A>a = ', 'A'>'a');
writeln('a>A = ', 'a'>'A');
writeln('6>3 = ', 6>3);
writeln('-3>4 = ', -3>4);
end.
```

3. Program Urut_Bil;

```
Uses Wincrt;
Var A,B,C:integer;

Begin
Writeln('Program Mengurut Bilangan');
Writeln('=====');
Writeln;
Write('Masukkan Nilai A: ');readln(A);
Write('Masukkan Nilai B: ');readln(B);
Write('Masukkan Nilai C: ');readln(C);
Writeln;
if (A<=B) and (A<=C) then
  if (B<=C) then
    Writeln(A,' ',B,' ',C)
  else
    Writeln(A,' ',C,' ',B)
  else if (B<=A) and (B<=C) then
    if (A<=C) then
      Writeln(B,' ',A,' ',C)
    else
      Writeln(B,' ',C,' ',A)
    else if (C<=A) and (C<=B) then
      if (A<=B) then
        Writeln(C,' ',A,' ',B)
```



```
else
    Writeln(C, ' ', B, ' ', A)
End.
```

Soal Latihan

1. Buatlah program untuk menentukan apakah suatu bilangan bulat itu habis dibagi 3 atau tidak!

Contoh tampilan:

Masukkan sembarang bilangan bulat = 9

Bilangan 9 habis dibagi 3.

2. Buatlah program untuk menentukan apakah sebuah bilangan merupakan bilangan ganjil atau bilangan genap!
3. Diberikan dua buah bilangan yang diinputkan dari keyboard. Sebutkan nama variabelnya adalah A dan B. Buatlah program untuk menampilkan nilai terbesar di antara kedua bilangan tersebut !
4. Buatlah program untuk menentukan besarnya pajak pendapatan dari seorang pegawai berdasarkan golongannya, dengan ketentuan sebagai berikut :

Gol A = 0

Gol B = 10% dari gaji

Gol C = 15% dari gaji

Gol D = 20% dari gaji.

Contoh Tampilan:

NIP : 135904373

Nama : Budi Darmawan

Golongan : B

Gaji : Rp.1500000

Pajak : Rp.150000

Gaji Bersih : Rp.1350000

BAB 7 Pengulangan Proses (Looping)

Terdapat tiga macam bentuk pengulangan dalam Pascal, yaitu dengan menggunakan statemen `For`, `While...do`, `Repeat....Until`.

Statemen For

Bentuk pengulangan dengan statemen `For` dapat berbentuk pengulangan positif (`For...to...do`) dan pengulangan negatif (`For...Downto...do`).

Pengulangan `For...to...do` adalah pengulangan dengan penghitung (*counter*) dari kecil ke besar atau disebut juga pertambahannya positif. Sintaksnya adalah sebagai berikut :

For variabel:=nilai1 **to** nilai2 **do** statemen

Pengulangan `For...downto...do` adalah pengulangan dengan penghitung (*counter*) dari besar ke kecil atau disebut juga pertambahannya negatif. Sintaksnya adalah sebagai berikut :

For variabel:=nilai2 **downto** nilai1 **do** statemen

Statemen `For` dengan syarat $nilai2 > nilai1$ mengakibatkan statemen *setelah* kata `do` dikerjakan sebanyak ($nilai2 - nilai1 + 1$) kali, dari nilai *variabel=nilai1* sampai *variabel=nilai2*. Dalam pengulangan `For` variabel, nilai1, nilai2 harus bertipe sama dan termasuk jenis ordinal(nilainya dapat dihitung secara berurutan, misal `char` dan `integer`).

Contoh

```
Program Loop1;  
uses wincrt;  
var i:integer;
```

```
Begin  
For i:=1 to 3 do  
writeln('Matematika');  
end.
```

Output :

```
Matematika  
Matematika  
Matematika
```

```
Program Loop2;  
uses wincrt;  
var i:integer;
```

```
Begin  
For i:=3 downto 3 do  
writeln('Matematika');  
end.
```

Output :

```
Matematika  
Matematika  
Matematika
```

Program di atas, statemen *setelah* kata `do` (`writeln('Matematika')`) diulang sebanyak 3 kali.

```
Program Loop3;  
uses wincrt;  
var i:integer;  
  
Begin  
For i:=1 to 3 do  
write(i);  
writeln(' Matematika');  
end.
```

Output :

```
123 Matematika
```

```
Program Loop4;  
uses wincrt;
```

```
var i:integer;  
  
Begin  
For i:=1 to 3 do  
Begin  
write(i);  
writeln(' Matematika');  
End;  
end.
```

Output :

```
1 Matematika  
2 matematika  
3 Matematika
```

Jika Program Loop3 dijalankan maka statemen yang diulang hanya statemen *setelah* kata *do* saja (statemen pertama) karena statemen kedua (`writeln(' Matematika')`) tidak masuk dalam bentuk blok statemen seperti dalam Program Loop4. Jika Program Loop4 dijalankan maka statemen *setelah* kata *do* diantara `Begin...end` (blok statemen) akan diulang sebanyak 3 kali.

Statemen While....Do

Statemen `while...do` digunakan untuk melakukan proses pengulangan suatu statemen atau blok statemen terus menerus selama kondisi bernilai benar. Statemen `while...do` biasa dipakai untuk melakukan pengulangan yang jumlahnya tidak diketahui di depan atau selang pencacahannya tidak sebesar 1 atau -1. Bentuk statemen `while...do` adalah sebagai berikut

```
While kondisi Do
    Statemen
```

Jadi statemen *setelah* kata *Do* akan terus dikerjakan selama kondisi bernilai benar. Jika kondisi bernilai `FALSE` di awal (sebelum `while`) maka statemen tidak akan pernah dikerjakan.

Contoh :

```
Program Loop5;
uses wincrt;
var i:integer;
begin
i:=0;
while i<4 do
begin
writeln(i);
i:=i+1;
end;
end.
```

Output :

```
0
1
2
3
```

Statemen Repeat....Until

Statemen `Repeat...until` digunakan untuk mengulang statemen atau blok statemen sampai kondisi bernilai `TRUE`. Jadi pengulangan justru dilakukan selama kondisi bernilai salah. Pemeriksaan kondisi pada pengulangan dengan `Repeat...until` dilakukan belakangan(diakhir), kebalikan dengan `While...do`. Hal ini mengakibatkan statemen-statemen di dalam pengulangan `Repeat...until` paling sedikit akan diprose satu kali.

Bentuk statemen `Repeat...until` adalah sebagai berikut :

```
Repeat
Statemen1;
Statemen2;
.....
Statemen;
Until kondisi;
```

Contoh

```
Program Loop6;
uses wincrt;
var i:integer;
begin
i:=0;
Repeat
writeln(i);
until i=4;
end.
```

Output :

```
1
2
3
```

```

4                                     i:=i+1;
                                     until i=4;
Program Loop7;                       end.
uses wincrt;
var i:integer;                         Output :
                                     0
Begin                                  1
i:=0;                                  2
Repeat                                 3
writeln(i);

```

Dengan menggunakan komputer, praktekkan beberapa contoh program dibawah ini :

```

1. Program Faktorial;
Uses Wincrt;
Var i,n,x:integer;

Begin
Writeln('Program Faktorial');
Writeln('=====');
Writeln;
Write('Masukkan Nilai Faktorial: ');Readln(n);
Writeln;
if (n<=0) then
  Writeln('Hasil Faktorial: ',1)
else
  Begin
  x:=1;
  For i := 1 to n do
  x:=x*i;
  Writeln('Hasil Faktorial: ',x);
  End;
End.

```

```

2. Program nested;
uses wincrt;
var i,j:integer;

Begin
For i:=1 to 3 do
  begin
  For j:=1 to 4 do
  writeln(i,j);
  writeln;
  end;
end.

```

```

3. Program nested2;
uses wincrt;
var i,j:integer;

Begin
For i:=1 to 3 do

```

```

begin
  For j:=1 to 4 do
    write(i,j,' ');
    writeln;
  end;
end.

```

4. Program Menampilkan_Bintang;

```

Uses Wincrt;
Var i,j,n:integer;

Begin
  Writeln('Program Menampilkan Bintang');
  Writeln('=====');
  Writeln;
  Write('Masukkan Jumlah Baris: ');readln(n);
  For i:= 1 to n do
    Begin
      For j:= 1 to i do
        Write('*');
        Writeln;
      End;
    End;
  End.

```

Soal Latihan

1. Buatlah program untuk mengkonversi bilangan desimal ke bilangan biner.
2. Buatlah program rata-rata sejumlah data (bilangan), misal output program sebagai berikut :

```

Program Rata-rata
Banyaknya data : 4
Masukkan Data :
Data ke-1 = 4
Data ke-2 = 2
Data ke-3 = 4
Data ke-4 = 6
Jumlah Total = 16.00
Rata-rata = 4.00

```

3. Buatlah progra untuk mencetak tanda bintang (*) sehingga didapatkan output sebagai berikut :

```

Jumlah Baris : 5
  *
 ***
*****
*****
*****

```

4. Buatlah program untuk menghitung perpangkatan bilangan bulat. Misal output program seperti di bawah ini :

```

Program Menghitung Pangkat Bilangan
=====
Masukkan Bilangan : 4
Masukkan Pangkat : 3
Hasil : 4^3 = 64.00

```

BAB 8 ARRAY

Array (larik) adalah suatu tipe terstruktur yang terdiri dari sejumlah komponen-komponen yang mempunyai tipe yang sama. Komponen-komponen ini disebut sebagai tipe komponen. Banyaknya komponen dalam suatu larik jumlahnya selalu tetap. Suatu indeks yang disebut tipe indeks (bertipe ordinal) menunjukkan banyaknya komponen dalam suatu larik. Tiap komponen dalam suatu larik dapat diakses dengan menunjukkan nilai indeksnya. Larik dapat bertipe data sederhana *byte*, *word*, *integer*, *real*, *boolean*, *char* atau *string* dan dapat juga bertipe *skalar* atau *subrange*.

Array Berdimensi Satu

Bentuk array berdimensi satu salah satu contohnya adalah sebagai berikut :

```
nama_larik : array[tipe indeks] of tipe larik
```

Contoh :

```
X : array [1..100] of integer;
```

dengan

```
X           : nama larik  
[1..100]    : tipe indeks  
Integer     : tipe dari larik
```

Larik X telah dideklarasikan sebagai larik bertipe *integer* dengan jumlah elemennya maksimum sebanyak 100 elemen. Nilai elemen dalam larik X harus bertipe atau berisi nilai-nilai *integer*. Misal elemen-elemen dari Larik X adalah :

```
X[1] := 10;  
X[2] := 5;  
X[3] := 8;
```

`X[1] := 10` menunjukkan bahwa X adalah nama larik, 1 adalah nilai indeks, 10 menunjukkan nilai *integer*.

Jika nilai ke-2 dari larik X akan ditampilkan, maka dalam pascal penulisannya menggunakan statemen

```
Writeln(X[2]);
```

Array dalam program pascal, banyak dipergunakan dalam statemen-statement pengulangan misalnya dalam statemen `For...to...do` dengan menggunakan variabel indeks.

Contoh :

```
Program Larik1;  
uses wincrt;  
var data:array[1..20] of integer;  
    m,n,i : integer;  
  
Begin  
Write('Banyaknya data : '); readln(n);  
for i:=1 to n do  
    begin  
        write ('Data ke-',i,' = ');readln(data[i]);  
    end;  
Write('Ingin melihat data ke : ');readln(m);
```

```
write('Data Ke-',m,' = ',data[m]);
end.
```

```
Output :
Banyaknya data : 4
Data ke-1 = 4
Data ke-2 = 5
Data ke-3 = 6
Data ke-4 = 9
Ingin melihat data ke : 3
Data Ke-3 = 6
```

Array Berdimensi Dua

Array (larik) berdimensi dua mewakili suatu bentuk tabel atau matrik, yaitu indeks yang pertama dapat menunjukkan baris dan indeks yang kedua menunjukkan kolom dari tabel atau matriks. Bentuk penulisan array berdimensi dua adalah sebagai berikut :

```
Nama_larik : Array[tipe-indeks1] of array [tipe-indeks2]
              of tipe larik
```

Atau

```
Nama_larik : Array [tipe-indeks1,tipe-indeks2] of tipe-
larik
```

Contoh

```
X : array [1..3,1..4] of integer
```

X menunjukkan suatu larik yang mempunyai jumlah baris maksimum 3 baris dan jumlah kolom maksimum 4 kolom.

Contoh :

```
Program Matriks;
uses wincrt;
var A:array[1..3,1..3] of integer;
    i,j,m,n,k,l : integer;

Begin
Writeln('Program input matriks A');
writeln('-----');
write('Banyaknya baris : ');readln(m);
write('Banyaknya kolom : ');readln(n);
for i:=1 to m do
  begin
  for j:=1 to n do
    begin
    write('A [' ,i ,',',j ,'] : ');readln(A[i,j]);
    end;
  writeln;
  end;
writeln('Melihat Elemen Matrik A');
write('Baris ke-');readln(k);
write('Kolom ke-');readln(l);
Write('Elemen Matrik A baris ke-',k,' kolom ke-',l,' =
',A[k,l]);
end.
```

Output :

```
Program input matriks A
-----
Banyaknya baris : 2
Banyaknya kolom : 2
A [1,1] : 3
A [1,2] : 4

A [2,1] : 5
A [2,2] : -6

Melihat Elemen Matrik A
Baris ke-2
Kolom ke-2
Elemen Matrik A baris ke-2 kolom ke-2 = -6
```

Dengan menggunakan komputer, praktekkan beberapa contoh program dibawah ini :

1. Program Larik2;
Uses Wincrt;
Var x : array [1..100] of integer;
 n,i :integer;
Begin
Write('Masukkan Jumlah Data: ');readln(n);
Writeln;
For i:= 1 to n do
 begin
 write('Data ke-',i,' : ');Readln(x[i]);
 end;
Writeln;
Write('Data Yang Telah Dimasukkan: ');
For i:= 1 to n do
Write(x[i],' ');
End.
2. Program MaxMin;
Uses Wincrt;
Var x : array [1..100] of integer;
n,i,max,min : integer;
Begin
Writeln('Program Array');
Writeln('=====');
Writeln;
Write('Masukkan Jumlah Data: ');readln(n);
Writeln;
For i:= 1 to n do
 Begin
 write('Data ke-',i,' : ');Readln(x[i]);
 end;
Writeln;
Write('Data Yang Telah Dimasukkan: ');
max:=x[1];
min:=x[1];
For i:= 1 to n do
 Begin


```

    Write(x[i], ' ');
    if (max<x[i]) then
        max:=x[i]
    else
        min:=x[i];
    End;
Writeln;
Writeln('Nilai Maximal: ',max);
Writeln('Nilai Minimal: ',min);
End.

```

3. Program MaxMin2;

```

Uses Wincrt;
Var x: array [1..100] of integer;
    n,i,max,min,tot,pos:integer;
    rt,sdt,sd,md:real;

Begin
Writeln('Program Array');
Writeln('=====');
Writeln;
Write('Masukkan Jumlah Data : ');readln(n);
Writeln;
For i:= 1 to n do
    Begin
        write('Data ke-',i,' : ');Readln(x[i]);
        end;
Writeln;
Write('Data Yang Telah Dimasukkan: ');
max:=x[1];
min:=x[1];
tot:=0;
sdt:=0;
For i:= 1 to n do
    Begin
        Write(x[i], ' ');
        if (max<x[i]) then
            max:=x[i]
        else
            min:=x[i];
            tot:=tot+x[i];
        End;
rt:=tot/n;
For i:= 1 to n do
    Begin
        sdt:=sdt+sqr(x[i]-rt);
        End;
sd:=sqrt(sdt/(n-1));
if (n mod 2 = 1) then
    begin
        pos:=(n div 2)+1;
        md:=x[pos];
    end
else
    begin
        pos:=(n div 2);

```

```

        md:=(x[pos]+x[pos+1])/2;
        end;
Writeln;
Writeln('Nilai Maximal : ',max);
Writeln('Nilai Minimal : ',min);
Writeln('Nilai Rata-Rata : ',rt:4:2);
Writeln('Standar Deviasi : ',sd:4:2);
Writeln('Median : ',md:4:2);
End.

```

4. Program Polindrom;

```

Uses Wincrt;
Var kt,hkt,hkt1:string;
    i,j:integer;

Begin
Writeln('Program Polindrom');
Writeln('=====');
Writeln;
Write('Masukkan Kata: ');Readln(kt);
Writeln;
j:=length(kt);
hkt:='';
For i:= 1 to j do
    hkt:=hkt+kt[i];
For i:= j downto 1 do
    hkt1:=hkt1+kt[i];
    Writeln('Asal: ',hkt,' Dibalik: ',hkt1);
    Writeln;
if (hkt=hkt1) then
    Writeln('Kata Tersebut Termasuk Polindrom!')
else
    Writeln('Kata Tersebut Tidak Termasuk Polindrom!');
End.

```

5. Program Pecahan;

```

Uses Wincrt;
Var pmb, pny : array [1..10] of integer;
    i,j,n,t1,t2 : integer;

Begin
Writeln('Program Pecahan');
Writeln('=====');
Writeln;
Write('Banyaknya pecahan: ');Readln(n);
Writeln;
For i := 1 to n do
    Begin
    Write('Pembilang ke-',i,' : ');Readln(pmb[i]);
    Write('Penyebut ke-',i,' : ');Readln(pny[i]);
    writeln;
    End;
Writeln;
Writeln('Pecahan Yang Di Masukkan:');
For i := 1 to n do
    Writeln(pmb[i], '/', pny[i]);
For i := 1 to n-1 do
For j := i+1 to n do
    Begin
    if ((pmb[i]/pny[i])>(pmb[j]/pny[j])) then
    Begin
    t1:=pmb[i];

```

```

        t2:=pny[i];
        pmb[i]:=pmb[j];
        pny[i]:=pny[j];
        pmb[j]:=t1;
        pny[j]:=t2;
    End;
End;
Writeln;
Writeln('Hasilnya: ');
For i := 1 to n do
    Writeln(pmb[i], '/' ,pny[i]);
End.

```

Soal Latihan

1. Buatlah program untuk menentukan jumlah dua buah matriks dan mencari selisih dua buah matriks. Dengan ketentuan sebagai berikut :
 - a. Proses input matriks langsung di inputkan di elemennya. (gubakan perintah GOTOXY)
 - b. Perhatikan syarat penjumlahan dan pengurangan dua buah matriks.

Contoh Output :

```

Program Penjumlahan Matriks
Jumlah baris : 2
Jumlah kolom : 2
Matriks A :
    2  2
    2  2

Matriks B :
    2  2
    2  2

Jumlah Matriks A + B =
    4  4
    4  4

```

2. Buatlah program perkalian antara dua buah matriks.(perhatikan syarat-syarat perkalian dua matriks)
3. Buatlah program untuk mencari transpose matriks berordo $m \times m$.

BAB 9 Record

Record adalah jenis tipe data terstruktur yang berisi beberapa data, yang masing-masing dapat berlainan tipe termasuk bertipe array. Masing-masing data tersebut disebut sebagai field. Tipe data record dideklarasikan dengan bentuk sebagai berikut :

```
Record
    Data_field_1 : tipe_1;
    Data_field_2 : tipe_2;
    .....
    Data_field_n : tipe_n;
End;
```

Masing-masing data field dapat berupa satu atau beberapa nama pengenalan. Jika data field berisi lebih dari satu maka antar data field dipisahkan dengan tanda koma.

Contoh :

```
Type
Data_Barang = Record
    Nama      : string;
    Kualitas  : char;
    Harga     : longint;
End;

Var Barang : Data_Barang;
```

Deklarasi record seperti di atas menunjukkan bahwa variabel barang mengandung tiga buah field, yaitu : Nama, Kualitas, Harga.

Field dari suatu record dapat diakses dengan bentuk :

```
Variabel.field
```

Contoh : Barang>Nama

Hal ini berarti field Nama dari variabel record bernama Barang.

Isi dari suatu field dapat ditampilkan dengan perintah write atau writeln dengan bentuk

```
Writeln(variabel.field);
```

Contoh : writeln(Barang>Nama);

Hal ini berarti perintah untuk menampilkan isi field Nama dari variabel record bernama Barang. Isi dari suatu record tidak dapat ditampilkan dengan write atau writeln secara langsung misal writeln(Barang).

Contoh

```
Program Rekam_1;
Uses Wincrt;
Type mhs = record
    NIM : String[12];
    Nama : String[20];
End;
Var data : mhs;

Begin
Write('NIM : ');Readln(data.NIM);
Write('Nama : ');Readln(data>Nama);
Writeln;
Writeln;
Writeln('NIM : ',data.NIM);
Writeln('Nama : ',data>Nama);
```

end.

Output :
NIM : 1234
Nama : Lala

NIM : 1234
Nama : Lala

Statemen Pernyataan WITH.....DO

Program Pascal menyediakan pernyataan With.....Do untuk mempermudah pengetikan dan mengurangi kesalahan dalam penggunaan tipe data record. Bentuk pernyataan with...do adalah sebagai berikut :

```
WITH nama_recrod DO  
    Statemen;
```

Penggunaan statemen with...do mengakibatkan field-field yang terletak pada bagian statemen dapat dituliskan tanpa perlu menyertakan lagi nama record dan tanda titik.

Contoh

```
Program Rekam2;  
Uses Wincrt;  
Type mhs = record  
    NIM : String[4];  
    Nama : String[20];  
End;  
  
Var data : mhs;  
  
Begin  
With data do  
    Begin  
        Write('NIM : ');Readln(NIM);  
        Write('Nama : ');Readln(Nama);  
    End;  
Writeln;  
Writeln;  
Writeln('NIM : ',data.NIM);  
Writeln('Nama : ',data.Nama);  
end.
```

Output :
NIM : 1234
Nama : Lala

NIM : 1234
Nama : Lala

Dengan menggunakan komputer, praktekan beberapa contoh program dibawah ini :

1. Program DataPegawai;
Uses Wincrt;
Type Pegawai = record
 NIP : String[9];
 Nama : String[30];
 Golongan : Char;
 Jamkerja : Real;
End;

```

Var Data : Pegawai;
    Gapok : Real;
    Insentif,Gaber : Real;
    Ul : Char;

Begin
Repeat
Clrscr;
Writeln('Entry Data Pegawai PT. XYZ');
Writeln('=====');
Writeln;
Write('NIP : ');Readln(Data.NIP);
Write('Nama : ');Readln(Data>Nama);
Write('Golongan : ');Readln(Data.Golongan);
Write('Jam Kerja : ');Readln(Data.Jamkerja);
Writeln;
Writeln;
Case Data.Golongan of
    '1' : Gapok:=1000000;
    '2' : Gapok:=1500000;
    '3' : Gapok:=2000000;
    Else
        Gapok:=0;
End;
if Data.Jamkerja>200 then
    Insentif:=(Data.Jamkerja-200)*10000
else
    Insentif:=0;
    Gaber:=Gapok+Insentif;
Clrscr;
Writeln('Laporan Gaji Pegawai');
Writeln('PT. XYZ');
Writeln;
Writeln('=====
');
Writeln('|NIP          | Nama          |
Golongan | Jam Kerja | Gaji          |');
Writeln('=====
');
Writeln('|',Data.NIP:10,'|',Data>Nama:25,'|',Data.Golonga
n:10,'|',Data.Jamkerja:8:0,'|',Gaber:14:2,'|');
Writeln('=====
=');
Writeln;
Write('Mau Ulang Lagi? [Y/T]: ');Readln(Ul);
Until Ucase(Ul)<>'Y';
End.

```

2. Program rekam3;

```

uses wincrt;
Type mahasiswa = record
    nama    :string[20];
    nim     :string[10];

```

```

                prodi   :string[20];
                alamat  :string[30];
            end;
var data_mhs : array[1..50] of mahasiswa;
    n,i      :integer;

Begin
clrscr;
Writeln('Program Data Mahasiswa');
writeln('=====');
writeln;
write('Jumlah mahasiswa : ');readln(n);
for i:=1 to n do
    begin
        writeln('Data ke-',i);
        write('Nama Mahasiswa : ');readln(data_mhs[i].nama);
        write('NIM           : ');readln(data_mhs[i].nim);
        write('Prodi         : ');readln(data_mhs[i].prodi);
        write('Alamat           :
');readln(data_mhs[i].alamat);
        writeln;
    end;
writeln('Tekan ENTER....');readln;
clrscr;
writeln('Tabel Data');
for i:=1 to n do
    begin
        with data_mhs[i] do {#32 = spasi}
            begin
                write(i,#32, nama,#32:20-length(nama));
                write(nim,#32:8-length(nim));
                write(prodi,#32:20-length(prodi));
                writeln(alamat);
            end;
        end;
    end;
end.

```

Soal Latihan

1. Buatlah program untuk menyatakan data buku yang berisi judul buku, nama pengarang, penerbit, tahun terbit dan jumlah buku.
2. Buatlah program untuk menyatakan data nama nomor handphone yang berisi nomor handphone, operator, nama pemilik.
3. Buatlah modifikasi program soal no 2 diatas untuk menampilkan data ke-i berdasarkan input yang diberikan.

BAB 10 Prosedur

Prosedur adalah suatu program terpisah dalam blok tersendiri yang berfungsi sebagai subprogram. Prosedur banyak digunakan pada program yang terstruktur, karena :

1. Merupakan penerapan konsep modular (memecah program yang rumit menjadi beberapa subprogram yang sederhana).
2. Untuk suatu proses yang dilakukan berulang-ulang, cukup dituliskan sekali saja dalam prosedur dan dapat dipanggil atau digunakan sewaktu-waktu diperlukan dan juga dapat digunakan berulang-ulang.

PROCEDURE dibagi menjadi dua :

1. Procedure Sederhana tidak menerima argumen (nilai atau data) ketika dieksekusi.
2. Procedure Kompleks menerima nilai yang diproses ketika dieksekusi.

Bentuk penggunaan prosedur dalam pascal adalah sebagai berikut :

```
Program Judul_Program;  
    Procedure Judul_Prosedur;  
    Begin  
        .....  
        .....  
    End;  
Begin  
    .....  
    .....  
End.
```

Procedure Sederhana

Procedure sederhana dipakai untuk menampilkan pilihan menu, dsb. procedure (module) tersebut terdiri atas beberapa pernyataan (*statements*), yang dikelompokkan dengan kata kunci begin dan end . Setiap procedure mempunyai nama.

Contoh :

```
Program Procedure1 ;  
uses wincrt;  
  
    PROCEDURE MENU;  
    begin  
        writeln('Pilihan Menu');  
        writeln(' 1: Statemen Percabangan');  
        writeln(' 2: Statemen Perulangan');  
        writeln(' 3: Statemen Array');  
    end;  
  
Begin  
    writeln('memanggil prosedur');  
    MENU;  
    writeln('kembali dari prosedur');  
end.
```

Output :


```
memanggil prosedur
Pilihan Menu
1: Statemen Percabangan
2: Statemen Perulangan
3: Statemen Array
kembali dari prosedur
```

Jangkauan Variabel dalam Prosedur

Variabel Global

Variabel global adalah variabel yang didefinisikan/terletak pada program utama, dimana semua subprogram (prosedur) bisa mengakses, mempergunakan dan memodifikasinya. Dari gambar berikut, variabel A, B, dan C dapat diakses oleh procedure D maupun E

```
Program Global
Var A, B, C
```

```
Procedure D
```

```
Procedure E
```

Variabel Lokal

Suatu procedure dapat mendeklarasikan variabelnya sendiri. Variabel-variabel itu hanya bekerja pada procedure dimana mereka dideklarasikan. Variabel-variabel tersebut dinamakan variabel local (local variable).

```
Program Lokal
Var A, F, G

Procedure Alfa
Var A,B,C

Procedure Beta
Var x,y

Procedure beta_1
Var m

Procedure beta_2
Var n
```

Keterangan gambar:

- Semua bisa mengakses variabel global A, F, G.

- Pada procedure alfa definisi global variabel A diganti dengan variabel lokal.
- beta_1 dan beta_2 dapat mengakses x dan y.
- beta_1 tidak dapat mengakses variabel n dan beta_2 tidak dapat mengakses m.
- Tdak ada subprogram, kecuali alfa dapat mengakses B dan C.
- Procedure beta dapat mengakses alfa dan beta.

Supaya nilai-nilai variabel dapat digunakan di modul lainnya yang membutuhkannya, maka dapat dilakukan dengan cara :

1. Dibuat bersifat global
Supaya suatu variabel dapat bersifat global, maka harus dideklarasikan di atas modul yang menggunakannya
2. Dikirimkan sebagai parameter ke odul yang membutuhkannya.
3. Parameter yang dikirimkan dari program utama ke modul prosedur disebut parameter nyata (actual parameter) dan parameter yang ada dan dituliskan pada judul prosedur disebut parameter formal (formal parameter). Prose pengiriman data lewat parameter nyata ke parameter formal disebut dengan parameter passing. Parameter nyata dan parameter formal harus bertipe sama. Parameter dapat dikirimkan secara nila (*by Value*) atau secara acuan (*By reference*).

```

Program Utama;

  Procedure X;
  Var C,D : real;

  Begin
  .....
  End;

Var A, B : integer;

  Procedure Y;
  Begin
  .....
  End;

  Procedure Z;
  Begin
  .....
  End;

Begin
.....
End.

```

Keterangan :
 Berdasarkan gambar disamping, variabel A dan B bersifat global untuk prosedur Y, prosedur Z dan program utama. Tetapi tidak bersifat global untuk prosedur X. Sehingga prosedur X tidak dapat menggunakan variabel A dan B.
 Variabel C dan D hanya bersifat lokal untuk prosedur X saja dan tidak dapat digunakan pada modul yang lainnya (prosedur Y, Z dan program utama)

Contoh :

```

Program Procedure2;
uses wincrt;

Procedure Hitung;

```

```

var x,y :real;

Begin
write ('Nilai X : ');readln(x);
y:=x*x;
writeln('Nilai Y = X*X');
writeln('          = ',y:3:2);
end;

Begin
Hitung;
end.

```

Output :

```

Nilai X : 3
Nilai Y = X*X
          = 9

```

Pengiriman Parameter *by Value*

Pengiriman *by Value* merupakan pengiriman searah, yaitu dari parameter nyata ke parameter formal dan tidak dikirimkan balik dari parameter formal ke parameter nyata. Jika parameter dikirim *by Value*, parameter formal di prosedur akan berisi nilai yang dikirimkan yang kemudian bersifat lokal di prosedur. Jika nilai parameter formal di prosedur berubah, maka tidak akan mempengaruhi nilai parameter nyata. Parameter yang digunakan dengan pengiriman *by Value* disebut parameter nilai (*value parameter*)

Contoh :

```

Program Procedure3;
uses wincrt;

Procedure hitung(A,B :integer);
var C : integer;

Begin
C:=A+B;
writeln(#10,'Nilai C : ',c);
end;

Var X,Y : integer;

Begin
write('Nilai X : ');readln(X);
write('Nilai Y : ');readln(Y);
hitung(X,Y);
end.

```

Output :

```

Nilai X : 3
Nilai Y : 4
Nilai C : 7

```

Keterangan :

1. Judul prosedur di atas adalah hitung dengan bentuk penulisan Procedure hitung(A,B : integer), dengan :

- A dan B : parameter formal
 Integer : tipe parameter
2. Variabel lokal (variabel C) yang hanya dipakai di prosedur dan tidak termasuk parameter nilai harus didefinisikan tersendiri.
 3. Nilai-nilai parameter nyata X dan Y di program utama dikirimkan ke parameter formal A dan B di prosedur. Sehingga nilai parameter A dan B di prosedur akan berisi nilai yang sama dengan parameter X dan Y di modul utama.

```
Procedure hitung(A,B :integer);
```

```
    hitung(X,Y)
```

Contoh :

```
Program Procedure4;
uses wincrt;

    Procedure hitung(A,B,C :integer);
    Begin
    C:=A+B;
    writeln;
    writeln('A : ',A,' B : ',B,' C : ',C);
    end;

var X,Y,Z :integer;

Begin
write('X : ');readln(X);
write('Y : ');readln(Y);
write('Z : ');readln(Z);
hitung(X,Y,Z);
writeln('X : ',X,' Y : ',Y,' Z : ',Z);
end.
```

Output :

```
X : 3
Y : 4
Z : 5

A : 3 B : 4 C : 7
X : 3 Y : 4 Z : 5
```

Keterangan :

1. Parameter formal A akan berisi nilai yang sama dengan parameter nyata X.
2. Parameter formal B akan berisi nilai yang sama dengan parameter nyata Y.
3. Parameter formal C akan berisi nilai yang sama dengan parameter nyata Z, tetapi pada modul prosedur hitung parameter formal C akan berganti dengan nilai A+B.
4. Parameter formal dengan pengiriman *by Value* sifatnya lokal, maka perubahan nilai parameter di prosedur tidak akan mempengaruhi nilai parameter nyata di modul lain, sehingga parameter Z nilainya tidak berubah.

```
Procedure hitung(A,B,C :integer);
```

```
hitung(X,Y,Z)
```

Pengiriman Parameter *By reference*

Perubahan-perubahan yang terjadi pada nilai parameter formal di prosedur akan mempengaruhi nilai parameter nyata jika pengiriman parameter dilakukan *By reference*. Parameter ini disebut dengan variabel parameter.

Contoh :

```
Program Procedure5;
uses wincrt;

Procedure hitung(var A,B,C :integer);
begin
C:=A+B;
writeln('A : ',A,' B : ',B,' C : ',C);
end;

var X,Y,Z : integer;

Begin
write('Nilai X : ');readln(X);
write('Nilai Y : ');readln(Y);
writeln;
hitung(X,Y,Z);
writeln('X : ',X,' Y : ',Y,' Z : ',Z);
end.
```

Output :

```
Nilai X : 2
Nilai Y : 3

A : 2 B : 3 C : 5
X : 2 Y : 3 Z : 5
```

Keterangan :

1. Nilai parameter nyata Z akan mengikuti perubahan nilai dari parameter formal C
2. Pengiriman parameter *By reference* merupakan pengiriman dua arah atau bolak balik, sehingga perubahan nilai di parameter formal akan mempengaruhi nilai parameter nyata.

Ciri pengiriman parameter secara acuan

```
Procedure hitung(VAR A,B,C :integer);
```

```
hitung(X,Y,Z)
```

Dengan menggunakan komputer, praktekkan beberapa contoh program dibawah ini :

1. Program Prosedur_aktual;

```
Uses Wincrt;
Var Y:char;
    m:byte;

    Procedure Tampil(x:char;n:byte);
    Var i:integer;

    Begin
    for i := 1 to n do
    Write(x);
    Writeln;
    End;

Begin
Tampil('+',8);
Tampil('*',10);
Tampil('A',5);
Y:='B';
m:=11;
Tampil(Y,m);
End.
```

2. Program Prosedur_reference;

```
Uses Wincrt;
Var a,b,c : Integer;

    Procedure Coba(var x,y:integer;var z:integer);
    Begin
    x:=x+1;
    y:=y+1;
    z:=x+y;
    End;

Begin
a:=2;b:=3;c:=0;
Coba(a,b,c);
Writeln('a = ',a);
Writeln('b = ',b);
Writeln('c = ',c);
End.
```

3. Program Tukar_Nilai;

```
Uses WinCrt;
Type Larik = Array [1..100] of Integer;
Var A,B : Larik;
    i,x,m : Byte;

    Procedure Tukar;
    Var T:Integer;
    Begin
```

```

x:=0;
For i := 1 to m do
  Begin
    T:=A[i];
    A[i]:=B[i];
    B[i]:=T;
    Gotoxy(15+x,6);Write(A[i]);
    Gotoxy(15+x,7);Write(B[i]);
    x:=x+2;
  End;
End;

Procedure Input;
Var x:Byte;
Begin
  Randomize;
  x:=0;
  For i := 1 to m do
    Begin
      A[i]:=Random(10);
      B[i]:=Random(10);
      Gotoxy(15+x,12);Write(A[i]);
      Gotoxy(15+x,13);Write(B[i]);
      x:=x+2;
    End;
  End;

Begin
  Gotoxy(21,1);Write('Program Menukar Nilai Larik A & B');
  Gotoxy(21,2);Write('=====');
  Gotoxy(1,4);Write('Jumlah Data : ');Readln(m);
  Gotoxy(5,6);Write('Nilai A:');
  Gotoxy(5,7);Write('Nilai B:');
  Input;
  Gotoxy(1,9);Write('Setelah Di Tukar');
  Gotoxy(1,10);Write('=====');
  Gotoxy(5,12);Write('Nilai A:');
  Gotoxy(5,13);Write('Nilai B:');
  Tukar;
End.

```

4. Program Urut_Pecahan;

```

Uses Wincrt;
Var pmb, pny : array [1..10] of integer;
    i, j, n : integer;

Procedure Urut(x : integer);
Var t1, t2 : integer;
Begin
  For i := 1 to x-1 do
    For j := i+1 to x do
      Begin
        if ((pmb[i]/pny[i]) > (pmb[j]/pny[j])) then

```

```

        Begin
            t1:=pmb[i];
            t2:=pny[i];
            pmb[i]:=pmb[j];
            pny[i]:=pny[j];
            pmb[j]:=t1;
            pny[j]:=t2;
        End;
    End;
End;

```

```

Begin
Gotoxy(30,1);Write('Program Urut Pecahan');
Gotoxy(30,2);Write('=====');
Gotoxy(1,4);Write('Jumlah Data Pecahan: ');Readln(n);
For i := 1 to n do
    Begin
        Gotoxy(1,5+i);Write('Input Pecahan ke-',i,' :
');Readln(pmb[i]);
        Gotoxy(24,5+i);Write('/ ');Readln(pny[i]);
    End;
Urut(n);
Writeln;
Writeln('Hasilnya: ');
For i := 1 to n do
    Writeln(pmb[i], '/', pny[i]);
End.

```

5. Program Polinomial;

```

Uses Wincrt;
Type Larik = Array [1..10] of Integer;
var P1,P2,HP : Larik;
    i,n,m,o : Integer;

    Procedure Input(q:integer; var P:Larik);
    Begin
        for i := q+1 downto 1 do
            begin
                Write('koefisien dari pangkat ke-',i-1,' : ');
                Readln(P[i]);
            end;
        End;

    Procedure Tampil(q:integer; P:Larik);
    Begin
        for i := q+1 downto 1 do
            begin
                if P[i]<>0 then
                    if i=q+1 then
                        Write(P[i],'x^',i-1)
                    else if P[i]>0 then
                        begin
                            if i=1 then

```



```

        Write('+',P[i])
    else if i=2 then
        Write('+',P[i],'x')
    else
        Write('+',P[i],'x^',i-1);
    end
    else
    begin
    if i=1 then
    Write(P[i])
    else if i=2 then
        Write(P[i],'x')
    else
        Write(P[i],'x^',i-1);
    end;
    end;
End;

Begin
Clrscr;
Writeln('Program Penjumlahan 2 Polinomial');
Writeln('=====');
Write('Masukkan Jumlah Pangkat Tertinggi Polinomial Ke-1:
');Readln(n);
Input(n,P1);
Write('P1 = ');
Tampil(n,P1);
Writeln;Writeln;
Write('Masukkan Jumlah Pangkat Tertinggi Polinomial Ke-2:
');Readln(m);
Input(m,P2);
Write('P2 = ');
Tampil(m,P2);
if m>n then
    o:=m
    else
    o:=n;
Writeln;
Writeln;
Write('Hasil Polinomial (P1+P2): ');
for i := o+1 downto 1 do
    HP[i]:=P1[i]+P2[i];
Tampil(o,HP);
End.

```

Soal Latihan

1. Buatlah program pascal untuk menghitung jumlah, selisih, perkalian dua buah matriks dan tentukan juga transpose matriksnya. (*Gunakan modul Procedure*).
2. Buatlah program data nilai sejumlah mahasiswa yang dan tentukan nilai maksimum, minimum dan rata-rata nilai tersebut (*Gunakan modul Procedure*)

BAB 11

Fungsi dan Rekursi

Deklarasi Function (fungsi) dalam pascal hampir sama dengan deklarasi procedure, hanya fungsi harus dideklarasikan dengan tipenya. Tipe deklarasi ini menunjukkan tipe hasil dari fungsi. Bentuk deklarasi fungsi adalah sebagai berikut :

```
Function judul_fungsi(daftar_parameter): tipe;
```

Function (fungsi) juga memakai data atau variabel ketika dieksekusi, tetapi mempunyai kemampuan untuk menghasilkan nilai pada procedure atau program yang memanggilnya.

Suatu function :

- Dimulai dengan kata kunci *function*
- Strukturnya sama dengan sebuah *procedure*
- Didalam fungsi, suatu nilai dihasilkan dengan nama *function*
- Suatu function dipakai pada sisi sebelah kanan pada suatu ekspresi
- Hanya menghasilkan tipe data sederhana

Contoh :

```
Program fungsil;
uses wincrt;

Function hitung(Var A,B : integer):integer;
begin
  hitung:=A+B;
end;

Var x,y : integer;

Begin
write('Nilai X : ');readln(x);
write('Nilai Y : ');readln(y);
writeln;
write(x,' + ',y,' = ',hitung(x,y));
end.
```

Output :

```
Nilai X : 2
Nilai Y : 3
2 + 3 = 5
```

Fungsi yang tanpa menggunakan parameter berarti nilai balik yang akan dihasilkan merupakan nilai yang pasti. Parameter digunakan untuk memberikan input pada fungsi dan fungsi akan memberikan hasil balik sesuai dengan parameter yang diberikan.

Pengiriman parameter dalam fungsi sama dengan dengan pengiriman parameter dalam prosedur yaitu *by Value* dan *By reference*.

Bentuk penulisan fungsi dengan pengiriman parameter *by Value* adalah sebagai berikut :

```
Function hitung (A, B : integer):integer;
```

Contoh :

```
Program fungsi2;
uses wincrt;

Function terbesar(X,Y :real):real;
begin
if X>Y then
    terbesar:=X
else
    terbesar:=Y;
end;
var nilai1, nilai2 : real;

Begin
Write('Nilai pertama : ');readln(nilai1);
write('Nilai kedua   : ');readln(nilai2);
writeln;
write('Nilai terbesar adalah
',terbesar(nilai1,nilai2):3:2);
end.
```

Output :

```
Nilai pertama : 32
Nilai kedua   : 70
Nilai terbesar adalah 70.00
```

Bentuk penulisan fungsi dengan pengiriman parameter *By reference* adalah sebagai berikut :

```
Function hitung ( var A, B : integer) : integer;
```

Pengiriman parameter *By reference* dalam fungsi akan mengakibatkan perubahan nilai parameter di fungsi dan juga akan mengakibatkan perubahan nilai parameter di modul yang mengirimkannya. Sehingga sama dengan prosedur, pengiriman parameter *by refence* dalam fungsi dapat dimanfaatkan sebagai hasil balik.

Contoh :

```
Program fungsi3;
uses wincrt;

Function hitung (var A,B,C : integer):integer;
begin
hitung:=A+B;
C:=A*B;
end;

Var X,Y,Z : integer;
```

```

Begin
write('Nilai X : ');readln(X);
write('Nilai Y : ');readln(Y);
writeln;
writeln(X,' + ',Y,' = ',hitung(X,Y,Z));
writeln(X,' * ',Y,' = ',Z);
end.

```

Output :

```

Nilai X : 2
Nilai Y : 3

2 + 3= 5
2 * 3= 6

```

Hubungan parameter tersebut dapat digambarkan sebagai berikut :

```

Function hitung (var A,B,C : integer):integer;
Begin

```

```

    hitung:= A + B;

```

```

    C:= A * B;

```

```

End.

```

```

writeln(X,' + ',Y,' = ',hitung(X,Y,Z));

```

```

writeln(X,' * ',Y,' = ',Z);

```

Rekursi dan Iterasi

Subprogram bisa memanggil dirinya (*Recursive Call*) dengan catatan bahwa memiliki syarat penghentian operasi (iterasi), sehingga perlu dilakukan kehati-hatian dalam penulisannya. Lihat contoh tentang faktorial suatu bilangan di bawah ini.

Contoh.

```

Program faktorial;
Uses wincrt;
Var x : integer;

function factorial (n:integer):integer;
begin
if n<2 then { ini adalah syarat penghentian operasi
}
    factorial:=1;

```

```
        else
            factorial:=n*factorial(n-1); { ini bagian
            rekursi}
        end;

begin
writeln('Masukkan nilai : '); readln(x);
writeln(x,'! adalah ',factorial(x));
end.
```

Output :

```
Masukkan nilai : 4
4! adalah 24
```

Bab 12

Sorting

Beberapa metode *sorting* mengurutkan data yang dikenal antara lain adalah:

1. *Bubble Sort* (sederhana tetapi lambat)
2. *Quick Sort* (cepat tetapi rumit)
3. *Shell Sort* (agak cepat dan tidak terlalu rumit)
4. *Selection Sort*
5. *Insert Sort*
6. *Merge Sort*

Bubble Sort

Algoritma *Bubble Sort* adalah salah satu algoritma pengurutan yang paling simple, baik dalam hal pengertian maupun penerapannya. Ide dari algoritma ini adalah mengulang proses perbandingan antara tiap-tiap elemen array dan menukarnya apabila urutannya salah. Perbandingan elemen-elemen ini akan terus diulang hingga tidak perlu dilakukan penukaran lagi. Algoritma ini termasuk dalam golongan algoritma comparison sort, karena menggunakan perbandingan dalam operasi antar elemennya. Berikut ini adalah gambaran dari algoritma bubble sort.

Teknik ini menyusun data yang diinginkan secara berurutan dengan membandingkan elemen data yang ada, misalkan kita akan menyusun data secara (ascending) cacah naik. Maka algoritma utamanya adalah seperti ini.

```
for i:=1 to Jumlah_data-1 do
  for j:=i+1 to Jumlah_data do
    if Data[i]>Data[j] then
      begin
        t:=Data[i];
        Data[i]:=Data[j];
        Data[j]:=t;
      end;
```

Quick Sort

Ditemukan oleh E. Hoare. Dengan menggunakan metode rekursi samapi habis. Prinsipnya membagi data menjadi dua bagian yang sama (kiri dan kanan). Dimana data tengah menjadi pivot (pusat operasi). Kumpulkan data dengan nilai lebih kecil dari pivot disebelah kiri pivot, dan di kanan untuk yang lebih besar. Dimungkinkan bagian kiri dan kanan pivot tidak sama besarnya. Untuk itu tiap bagian di bagi menjadi dua lagi dan mempunyai pivot yang baru .

Quick Sort juga disebut juga dengan partition Exchange sort. Disebut *Quick Sort*, karena terbukti mempunyai 'average behaviour' yang terbaik di antara metode pengurutan yang ada. Disebut Partition Exchange Sort karena konsepnya membuat partisi-partisi, dan sort dilakukan per partisi.

Teknik mempartisi tabel:

1. pilih $x \in \{a_1, a_2, \dots, a_n\}$ sebagai elemen pivot,
2. pindai (scan) tabel dari kiri sampai ditemukan elemen $a_p \leq x$
3. pindai tabel dari kanan sampai ditemukan elemen $a_q \geq x$
4. pertukarkan $a_p \leftrightarrow a_q$

5. ulangi (ii) dari posisi p+1, dan (iii) dari posisi q-1, sampai kedua pemindaian bertemu di tengah
6. tabel.

Dalam algoritma *Quick Sort*, pemilihan pivot adalah hal yang menentukan apakah algoritma *Quick Sort* tersebut akan memberikan performa terbaik atau terburuk. Berikut beberapa cara pemilihan pivot :

1. Pivot = elemen pertama, elemen terakhir, atau elemen tengah tabel. Cara ini hanya bagus jika
2. elemen tabel tersusun secara acak, tetapi tidak bagus jika elemen tabel semula sudah terurut. Misalnya, jika elemen tabel semula menurun, maka semua elemen tabel akan terkumpul di upatabel kana.
3. Pivot dipilih secara acak dari salah satu elemen tabel. Cara ini baik, tetapi mahal, sebab memerlukan biaya (cost) untuk pembangkitan prosedur acak. Lagi pula, ita tidak mengurangi kompleksitas waktu algoritma.
4. Pivot = elemen median tabel. Cara ini paling bagus, karena hasil partisi menghasilkan dua bagian tabel yang berukuran seimbang (masing-masing = $n/2$ elemen). Cara ini memberkan kompleksitas waktu yang minimum. Masalahnya, mencari median dari elemen tabel yang belum terurut adalah persoalan tersendiri.

Algoritmanya adalah sebagai berikut :

```
pusat:=A[(awal+akhir) div 2];
kiri:=awal;
kanan:=akhir;
While kiri<=kanan Do
  Begin
    While A[kiri]<pusat Do
      Inc(kiri);
    While A[kanan]>pusat Do
      Dec(kanan);
    If kiri<=kanan Then
      Begin
        Ganti(A[kiri],A[kanan]);
        Inc(kiri);
        Dec(kanan);
        Inc(baca);
      End;
  End;
If kanan>awal Then
  Urut(awal,kanan);
If akhir>kiri Then
  Urut(kiri,akhir);
```

Shell Sort

Prinsipnya hampir sama dengan *Bubble Sort* tetapi dioptmisisasi sehingga lebih cepat. Ditemukan oleh Donald Shell. prinsipnya adalah membandingkan data dengan jarak tertentu dalam array. Algoritma utamanya adalah sebagai berikut :

```
baca:=0;
For i:= (m Div 2) Downto 1 Do
  For j:= 1 To m-i Do
```

```

If A[j]>A[j+i] Then
  Begin
  Ganti(A[j],A[j+i]);
  Inc(baca);
  End;

```

Selection Sort

Algoritma utamanya adalah sebagai berikut :

```

baca:=0;
For i:= 1 To m-1 Do
  Begin
  tempat:=i;
  For j:= i+1 To m Do
  If A[tempat]>A[j] Then
    tempat:=j;
  Ganti(A[i],A[tempat]);
  Inc(baca);
  End;

```

Insert Sort

Algoritma utamanya adalah sebagai berikut :

```

baca:=0;
For i:= 2 To m Do
  Begin
  G:=A[i];
  j:=i-1;
  A[0]:=G;
  While G<A[j] Do
    Begin
    A[j+1]:=A[j];
    Dec(j);
    Inc(baca);
    End;
  A[j+1]:=G;
  End;

```

Merge Sort

Algoritma utamanya

```

Begin
i:=awal;
k:=awal;
j:=tengah+1;
Repeat
  If A[i]<A[j] Then
    Begin
    B[k]:=A[i];
    Inc(i);
    End
  Else
    Begin
    B[k]:=A[j];
    Inc(j);
    End;
  Inc(k);
  Inc(baca);

```



```
Until (i>tengah) Or (j>akhir);
If i>tengah Then
    For t:= j To akhir Do
        Begin
            B[k+t-j]:=A[t];
        End
    Else
        For t:= i To tengah Do
            Begin
                B[k+t-i]:=A[t];
            End;
        End;
    End;
```

Soal Latihan

1. Jelaskan Alur logika (dengan disertai contoh data) metode sorting data di atas?
2. Buatlah program pascal untuk mengurutkan data dengan metode sorting data di atas!

Daftar Pustaka

- _____, *Bahasa Pemrograman Pascal*, Universitas Sebelas Maret, Surakarta, <http://miwan.ueuo.com/materi/Pascal.pdf>, Diakses 15 September 2010
- Fachrie Lantera, Kompleksitas Algoritma Quick Sort, <http://www.informatika.org/~rinaldi/Matdis/2008-2009/Makalah2008/Makalah0809-019.pdf>, Diakses 25 September 2010
- Hendarsyah Decky, *Kumpulan Program Pascal*, <http://ilmukomputer.com>, Diakses tanggal 20 September 2010
- Kadir A, 2002, *Pemrograman Pascal Buku 1*, Penerbit Andi, Yogyakarta.
- Kadir A, 2002, *Pemrograman Pascal Buku 2*, Penerbit Andi, Yogyakarta.
- Jogiyanto HM, 2002, *Turbo Pascal Versi 5.0*, Penerbit Andi, Yogyakarta.
- Ryan Rheinadi , Analisis Algoritma Bubble Sort, <http://webmail.informatika.org/~rinaldi/Matdis/2009-2010/Makalah0910/MakalahStrukdis0910-032.pdf>, Diakses 25 September 2010
- Sahid, 2004, *Pemrograman Komputer dengan Turbo Pascal versi Windows 1.5 (Edisi Revisi)*, Lab. Komputer Jurdik Matematika FMIPA UNY