BAHAN AJAR
ALGORITMA DAN PEMROGRAMAN I

OLEH:

Budi Mulyono, S.Pd., M.Sc.

Drs. Purwoko, M.Si.

PENDIDIKAN MATEMATIKA
FAKULTAS KEGURUAN DAN ILMU PENDIDIKAN
UNIVERSITAS SRIWIJAYA

KATA PENGANTAR

Bahan ajar ini merupakan bahan yang digunakan untuk perkuliahan Algoritma dan Pemrograman 1. Bahasa pemrograman yang digunakan dalam buku ini adalah bahasa Turbo Pascal. Tujuan dirancangnya bahan ajar ini adalah untuk membekali mahasiswa tentang konsep dasar algoritma dan pemrograman, sehingga diharapkan mahasiswa dapat mengaplikasikan konsep tersebut dalam membuat sebuah program khususnya program yang berkaitan dengan pemecahan masalah-masalah matematika.

Dalam bahan ajar ini terdiri dari delapan bab, yaitu:

Bab 1 Algoritma

Bab 2 Tipe Data

Bab 3 Bahasa Pascal

Bab 4 Pengurutan

Bab 5 Pemilihan

Bab 6 Pengulangan

Bab 7 Prosedur

Bab 8 Fungsi

Penulis ucapkan terima kasih kepada semua pihak yang telah membantu sehingga dapat menyelesaikan bahan ajar ini. Akhirnya, penulis mengharapkan kritik dan saran untuk perbaikan kedepannya yang lebih baik.

Palembang, 2012

Penulis

DAFTAR ISI

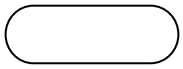
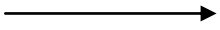
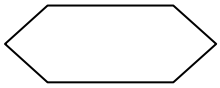
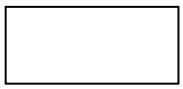
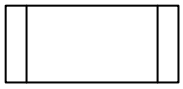
Bab 1
Algoritma

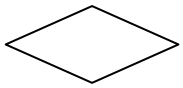
1. Pengertian Algoritma

Algoritma adalah urutan langkah-langkah logika yang menyatakan suatu tugas dalam menyelesaikan suatu masalah atau problem.

Notasi yang digunakan untuk menuliskan algoritma ada dua, yaitu dengan menyatakan langkah-langkah algoritma dengan untaian kalimat deskriftif, atau dengan mengunakan diagram alir (flowchart). Algoritma dengan flowchart dilakukan dengan simbol-simbol seperti yang diperlihatkan pada **Tabel 1**.

Tabel 1 Simbol-Simbol Flowchart.

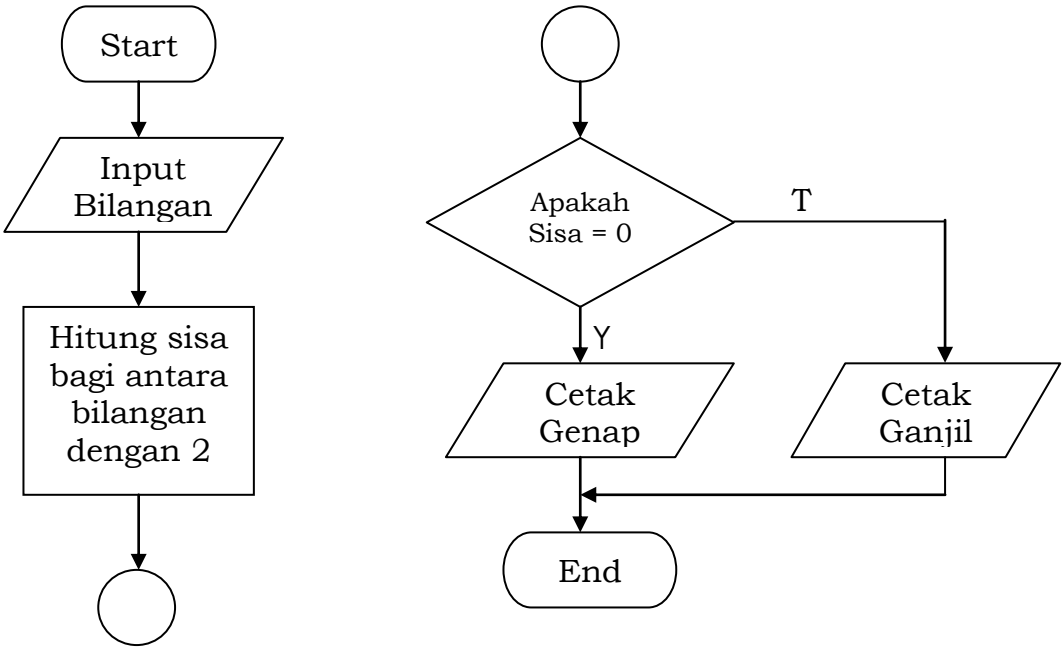
SIMBOL	NAMA	FUNGSI
	**TERMINATOR**	Permulaan/akhir program
	**GARIS ALIR (FLOW LINE)**	Arah aliran program
	**PREPARATION**	Proses inisialisasi/pemberian harga awal
	**PROSES**	Proses perhitungan/proses pengolahan data
	**INPUT/OUTPUT DATA**	Proses input/output data, parameter, informasi
	**PREDEFINED PROCESS (SUB PROGRAM)**	Permulaan sub program/proses menjalankan sub program

SIMBOL	NAMA	FUNGSI
	**DECISION**	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	**ON PAGE CONNECTOR**	Penghubung bagian-bagian flowchart yang berada pada satu halaman
	**OFF PAGE CONNECTOR**	Penghubung bagian-bagian flowchart yang berada pada halaman berbeda

Contoh 1.1.
Algoritma untuk menentukan jenis suatu bilangan apakah bilangan ganjil atau bilangan genap.

Dengan untaian kalimat deskriftif, algoritmanya adalah sebagai berikut:
- Bagi bilangan dengan bilangan 2
- Hitung sisa hasil bagi pada langkah 1.
- Bila sisa hasil bagi sama dengan 0 maka bilangan itu adalah bilangan genap tetapi bila sisa hasil bagi sama dengan 1 maka bilangan itu adalah bilangan ganjil.

Dengan flowchart, algoritma disajikan pada Gambar 1.



Gambar 1 Flowchart Mencetak Jenis Bilangan.

2. Struktur Dasar dan Notasi Algoritmik

Algoritma berisi langkah-langkah penyelesaian masalah. Langkah-langkah penyelesaian tersebut secara umum dibedakan menjadi tiga macam struktur, yaitu runtunan (*sequence*), pemilihan (*selection*), dan pengulangan (*repetition*).

Agar mudah dibaca, algoritma dituliskan dalam notasi algoritmik, tidak ada notasi yang standar untuk menuliskan algoritma, kita dapat menuliskan algoritma dengan notasi sendiri, karena teks algoritma tidak sama dengan teks program komputer. Namun, agar notasi algoritmik mudah ditranslasikan ke dalam notasi bahasa pemrograman, maka sebaiknya notasi algoritmik tersebut berupa *pseudo-code* yang berkoresponden dengan notasi bahasa pemrograman secara umum.

Beberapa notasi algoritmik yang sering digunakan diperlihatkan pada Bab 3.

Latihan

1) Buat algoritma untuk menentukan apakah suatu bilangan merupakan bilangan prima atau bukan, kemudian buat flowchart untuk program tersebut !
2) Buat algoritma untuk mencetak N buah bilangan prima yang pertama, kemudian buat flowchart untuk program tersebut !
3) Buat algoritma untuk menentukan jenis akar dari suatu persamaan kuadrat, kemudian buat flowchart untuk program tersebut !
4) Buat algoritma untuk menghitung jumlah N suku dari deret aritmatika berikut :

   $$Sn = 3 + 7 + 11 + \ldots\ldots + (4n-1)$$

5) Buat algoritma untuk menghitung nilai faktorial dari suatu bilangan, kemudian buat flowchart untuk program tersebut !
6) Buat flowchart untuk mencetak pasangan nilai X dan Y dimana hubungan antara X dan Y memenuhi persamaan $Y = X^3 - 2X + 1$ dan nilai x berubah dari –10 sampai 10 !

Bab 2
TIPE DATA

Program komputer bekerja dengan memanipulasi objek (data) di dalam memori. Objek yang akan diprogram memiliki tipe yang bermacam-macam. Tipe data dapat dikelompokkan menjadi dua, yaitu: tipe dasar dan tipe bentukan. Tipe dasar adalah tipe yang dapat langsung dipakai, sedangkan tipe bentukan merupakan tipe yang dibentuk dari tipe dasar, atau dari tipe bentukan lain yang sudah didefinisikan.

1. Tipe Dasar

Yang termasuk dalam tipe dasar adalah bilangan logika, bilangan bulat, karakter, bilangan riil, dan string.

Bilangan logika
Nama tipe : boolean
Ranah nilai : benar (true) atau salah (false)
Konstanta : true dan false
Operasi : not, and, or, dan xor

Bilangan Bulat
Nama tipe : integer
Ranah nilai :
Untuk kompilator Turbo Pascal, tipe integer direpresentasikan ke dalam empat tipe seperti pada Tabel **2**.

Tabel 2 Tipe Integer Bahasa Pascal.

Tipe	Rentang Nilai	Format
byte	0 ... 255	Unsigned 8-bit
shortint	-128 ... 127	Signed 8-bit
word	0 ... 65535	Unsigned 16-bit
integer	-32768 ...32767	Signed 16-bit
longint	-2147483648 ...	Signed 32-bit

2147483647

Konstanta : harus ditulis tanpa mengandung titik desimal
Operasi : ada dua operasi, yaitu operasi aritmatika, dan operasi perbandingan.

Operasi aritmatika
+ (tambah)
- (kurang)
* (kali)
div (bagi)
mod (sisa hasil bagi)

operasi perbandingan
< (lebih kecil)
≤ (lebih kecil atau sama dengan)
> (lebih besar)
≥ (lebih besar atau sama dengan)
= (sama dengan)
≠ (tidak sama dengan)

Bilangan Riil
Nama Tipe : real
Ranah Nilai : untuk kompilator Turbo Pascal, tipe real seperti pada Tabel 3.

Tabel 3 Tipe Real Bahasa Pascal.

Tipe	Rentang Nilai	Format
real	2.9×10^{-39} .. 1.7×10^{38}	6 byte
single	1.5×10^{-45} .. 3.4×10^{38}	4 byte
double	5.0×10^{-324} .. 1.7×10^{308}	8 byte
extended	3.4×10^{-4932} .. 1.1×10^{4932}	10 byte

Konstanta : harus ditulis dengan tanda titik desimal. Contoh: 0.59 , 0.0 , -14.87
Operasi : terdiri dari operasi aritmetika dan operasi perbandingan.

Operasi aritmatika
+ (tambah)
- (kurang)
* (kali)
/ (bagi)

Operasi perbandingan
< (lebih kecil)
≤ (lebih kecil atau sama dengan)
> (lebih besar)
≥ (lebih besar atau sama dengan)
≠ (tidak sama dengan)

Karakter
Nama Tipe : char
Ranah Nilai : semua huruf dalam alfabet, angka desimal, tanda baca, operator aritmatik dan lain-lain.
Konstanta : harus dapat diapit oleh tanda petik tunggal
Operasi : operasi yang berlaku untuk tipe karakter adalah operasi perbandingan berikut
= (sama dengan)
≠ (tidak sama dengan)
< (lebih kecil)
> (lebih besar)
≥ (lebih besar atau sama dengan)

String
Nama Tipe : string
Ranah Nilai : ranah nilai untuk tipe string adalah deretan karakter yang telah didefinisikan pada ranah karakter.
Konstanta : harus diapit oleh tanda petik tunggal
Operasi : ada dua operasi yang berlaku, yaitu:
Operasi Penyambungan (concatination), dengan operator +

Operasi Perbandingan
= (sama dengan)
≠ (tidak sama dengan)
< (lebih kecil)
> (lebih besar)
≥ (lebih besar atau sama dengan)
≤ (lebih kecil atau sama dengan)

2. Tipe Bentukan

Tipe bentukan adalah tipe yang didefinisikan sendiri oleh pemrogram, yang disusun dari satu atau lebih tipe dasar. Ada dua macam tipe bentukan, yaitu: tipe dasar yang diberi nama dengan tipe baru, dan tipe terstruktur.

Tipe Dasar yang diberi Nama Tipe Baru

Penggunaan tipe baru dapat dilakukan apabila pemrogram menginginkan nama yang lebih akrab dengannya sehingga lebih mudah diinterpretasikan oleh orang yang membaca teks algoritma. Pemberian nama baru untuk tipe dasar dilakukan dengan kata kunci *type*. Contoh: type BilanganBulat : integer.

Tipe Terstruktur

Tipe terstruktur adalah tipe yangberbentuk rekaman (*record*). Rekaman disusun dari satu atau lebih field dengan nama tipe tertentu. Berikut ini diberikan contoh pendefinisian tipe terstruktur

Contoh 2.1. (Pendefinisian tipe titik)
Titik dalam koordinat kartesius dinyatakan dengan (x,y), sehingga tipe titik mengandung field x dan y. Misalnya titik P(x,y).

Cara penulisan
type Titik : *record* <x : *real*, y : *real*>
atau
type Titik : *record* <x , y : *real*>

cara mengacu pada tiap field pada P adalah
P.x (untuk mengacu field x)
P.y (untuk mengacu field y)

Contoh 2.2. (Pendefinisian tipe bilangan kompleks)
Bilangan kompleks dinyatakan dengan a + bi, sehingga mengandung field a dan b. Misalnya bilangan kompleks K = a + bi.

Cara penulisan
type Kompleks : *record* <a : *real*, b : *real*>

cara mengacu pada tiap field pada K adalah
K.a (untuk mengacu field a)
K.b (untuk mengacu field b)

Contoh 2.3. (Pendefinisian tipe tanggal)
Tanggal terdiri dari field tanggal (dd), bulan(mm), dan tahun(yy).

Cara penulisan
type Tanggal : *record*
<dd : integer,
mm : integer,
yy : integer
>

Cara mengacu fieldnya jika ada variabel D yang bertipe *Tanggal.*
D.dd
D.mm
D.yy

Bab 3
BAHASA PASCAL

Bahasa Pascal merupakan salah satu bahasa pemrograman.

1. Translasi Notasi Algoritmik ke Bahasa Pascal

Translasi dari notasi algoritmik ke notasi bahasa Pascal untuk beberapa pernyataan disajikan pada Tabel **4**.

Tabel 4 Translasi Notasi Algoritmik ke Bahasa Pascal.

No.	Pernyataan	Algoritmik	Bahasa Pascal
1.	Penugasan	←	:=
2.	Pembacaan	read	read readln
3.	Penulisan	write	write writeln
4.	Tipe dasar	boolean integer real char string record <field1 : type, field2 : type, ...	boolean byte shortint word integer longint real double extended char string string[n] record field1 : type; field2 : type; ...

No.	Pernyataan	Algoritmik	Bahasa Pascal
		fieldN : type >	fieldN : type; end;
5.	Operator a. Aritmatika	 + - * / div mod	 + - * / div mod
	b. Perbandingan	< ≤ > ≥ = ≠	< <= > >= = <>
	c. Logika	not and or xor	not and or xor
	d. String	+ < ≥ > ≤ = ≠	+ < >= > <= = <>
6.	Komentar	{ komentar }	{ komentar } (* komentar *)
7.	Konstanta, type, benar, salah	const type true false	const type true false
8.	IF-THEN	if *kondisi* then *pernyataan* endif	if *kondisi* then *Pernyataan*; Bila pernyataan lebih dari satu maka penulisannya: if *kondisi* then begin

No.	Pernyataan	Algoritmik	Bahasa Pascal
			Pernyataan1; *Pernyataan2*; ... pernyataanN; end;
9.	IF-THEN-ELSE	if *kondisi* then *pernyataan 1* else *pernyataan 2* endif	if *kondisi* then *pernyataan 1* else *pernyataan 2;*
10.	CASE	case *ekspresi* *nilai* 1 : *pernyataan* 1 *nilai* 2 : *pernyataan* 2 *nilai* 3 : *pernyataan* 3 *nilai* 4 : *pernyataan* 4 . . . *nilai* n : *pernyataan* n otherwise : *pernyataan* x endcase	case *ekspresi of* *nilai* 1 : *pernyataan* 1; *nilai* 2 : *pernyataan* 2; *nilai* 3 : *pernyataan* 3; *nilai* 4 : *pernyataan* 4; . . . *nilai* n : *pernyataan* n; else *pernyataan* x; end;
11.	FOR (menaik)	for *pencacah* ← *nilai_awal* to *nilai_akhir* do *pernyataan* endfor	for *pencacah* := *nilai_awal* to *nilai_akhir* do *pernyataan;* untuk pernyataan yang lebih dari satu: for *pencacah* := *nilai_awal* to *nilai_akhir* do begin *pernyataan1;* *pernyataan2;* *...* *pernyataanN;* end;
12.	FOR (menurun)	for *pencacah* ← *nilai_akhir* downto *nilai_awal* do *pernyataan* endfor	for *pencacah* := *nilai_akhir* to *nilai_awal* do *pernyataan;* untuk pernyataan yang lebih dari satu: for *pencacah* := *nilai_akhir* to *nilai_awal* do

No.	Pernyataan	Algoritmik	Bahasa Pascal
			begin *pernyataan1;* *pernyataan2;* *...* *pernyataanN;* end;
13.	WHILE	while *kondisi* do *pernyataan* endwhile	while *kondisi* do *pernyataan;* untuk pernyataan yang lebih dari satu: while *kondisi* do begin *pernyataan1;* *pernyataan2;* *...* *pernyataanN;* end;
14.	REPEAT	repeat *pernyataan* until *kondisi*	repeat *pernyataan;* until *kondisi;*

2. Beberapa Statemen dalam Turbo Pascal

Statemen adalah perintah untuk mengerjakan 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 yyang berupa fungsi dan prosedur yang telah disediakan sebagai perintah standar Turbo Pascal.

1) Statemen-statemen yang digunakan untuk input/output
   - Read/Readln [prosedur]
     Untuk memasukkan data lewat keyboard ke dalam suatu variabel.
     Read, pada statemen ini posisi kursor tidak pindah ke baris selanjutnya.
     Readln, pada statemen ini posisi kursor akan pindah ke baris selanjutnya setelah di input.
   - ReadKey [fungsi]
     Untuk pembacaan sebuah karakter dari keyboard. Tipe data yang dihasilkan adalah char.
   - Write/Writeln
     Untuk menampilkan isi dari suatu nilai variabel di layar.

2) Statemen-statemen yang digunakan untuk pengaturan letak layar
   - ClrScr [prosedur]
     Untuk membersihkan layar
   - GotoXY [prosedur]
     Untuk menempatkan posisi kursor pada layar
   - DelLine [prosedur]
     Untuk menghapus sebuah baris pada posisi kursor dan menaikkan
   - InsLine [prosedur]
     Untuk menyisipkan sebuah baris pada posisi kursor dan mengeser ke bawah tampilan-tampilan baris dibawahnya
   - Delay [prosedur]
     Untuk menghentikan sejenak proses program

3) Statemen-statemen yang digunakan untuk memanifulasi string
   - ConCat [fungsi]
     Untuk menggabungkan dua atau lebih variabel string.
   - Copy [fungsi]
     Untuk mengambil satu atau beberapa karakter dari sebuah string.
   - Delete [prosedur]
     Untuk menghapus sebagian karakter dari sebuah string
   - Insert [prosedur]
     Untuk menyisipkan satu atau beberapa karakter ke dalam sebuah string
   - Length [fungsi]
     Untuk memberikan nilai panjang dari suatu string.
   - Pos [fungsi]
     Untuk mencari posisi sebuah bagian string (substring) di dalam sebuah string.
   - Str [prosedur]
     Untuk merubah nilai numerik ke dalam nilai string.
   - Val [prosedur]
     Untuk merubah nilai string ke dalam nilai numerik.
   - UpCase [fungsi]
     Untk memberikan huruf kapital dari argumen

4) Statemen-statemen untuk perhitungan aritmatik
   - Abs [fungsi]
     Untuk memberikan nilai mutlak dari suatu argumen
   - ArcTan [fungsi]
     Untuk memberikan nilai dari fungsi arctangent dari perhitungan goniometri.
   - Cos [fungsi]
     Untuk memberikan nilai dari fungsi cosinus.
   - Exp [fungsi]

Untuk menghitung nilai pangkat dari bilangan e

- Frac [fungsi]
  Untuk mendapatkan nilai pecahan dari suatu bilangan.
- Int [fungsi]
  Untuk memberikan nilai integer dari suatu variabel dengan membuang bilangan di belakang koma.
- Ln [fungsi]
  Untuk menghitung nilai logaritma alam dari suatu nilai.
- Sin [fungsi]
  Untuk memberikan nilai dari fungsi sinus
- Sqr [fungsi]
  Untuk menghitung nilai pangkat dua dari suatu bilangan
- Sqrt [fungsi]
  Untuk menghitung nilai akar dari suatu bilangan.

5) Statemen-statemen untuk transfer nilai dari suatu variabel
   - Chr [fungsi]
     Untuk merubah nilai dari byte ke bentuk karakter yang sesuai dengan kode ASCII.
   - Ord [fungsi]
     Untuk merubah nilai suatu variabel dari bentuk karakter ke bentuk longint.
   - Round [fungsi]
     Untuk membulatkan data tipe real ke data tipe longint.
   - TRUNC [fungsi]
     Untuk membulatkan ke bawah data tipe real ke data tipe longint.

6) Statemen-statemen untuk memanipulasi data
   - Pred [fungsi]
     Untuk memberikan nilai sebelum nilai argumen dalam urutannya dalam ASCII.
   - Succ [fungsi]
     Untuk memberikan nilai sesudah nilai argumen dalam urutannya dalam ASCII.
   - Inc [fungsi]
     Untuk menambah nilai suatu variabel.
   - Dec [fungsi]
     Untuk mengurangi nilai suatu variabel.

7) Statemen-statemen tambahan
   - TextColor [prosedur]
     Untuk mengatur warna dari karakter-karakter ke layar.
   - TextBackGround [prosedur]
     Untuk mengatur warna latar belakang dari karakter-karakter di layar.

- Window [prosedur]
  Untuk membuat suatu jendela yang terletak pada layar.
- TextMode [prosedur]
  Untuk mengatur lebar layar, 80 kolom atau 40 kolom.
- Sound [prosedur]
  Untuk mengaktifkan suara (*beep*) pada internal speaker.

Bab 4
RUNTUNAN

Runtunan merupakan struktur algoritma yang terdiri dari satu atau lebih instruksi yang tiap instruksi dikerjakan satu per satu, tiap instruksi dilaksanakan tepat sekali, tidak ada instruksi yang berulang, urutan instruksi yang dilaksanakan pemroses sama dengan urutan instruksi sebagaimana yang tertulis di dalam teks algoritma, akhir dari instruksi merupakan akhir algoritma.

Contoh 4.1. (Mencetak Pesan ”Matematika OK”)
Tuliskan algoritma untuk mencetak pesan “Matematika OK”

Versi 1:

Program Matematika_OK1
{Program untuk mencetak “Matematika OK”}

Deklarasi
{Tidak ada}

Algoritma:
    write(‘Matematika OK’)

PASCAL:

```
program Matematika_OK1;
{ Menampilkan kata matematika OK }


uses crt;

begin
clrscr;

    write('Matematika OK');
end.
```

Versi 2:

Program Matematika_OK2
{Program untuk mencetak “Matematika OK”}

Deklarasi
pesan : string

Algoritma:
pesan ← ‘Matematika OK’
write (pesan)

PASCAL:

```
program Matematika_OK2;
{ Menampilkan kata matematika OK }

uses crt;

var
    pesan : string;

begin
clrscr;
    read(pesan);

    write(pesan);
end.
```

Versi 3:

Program Matematika_OK3
{Program untuk mencetak “Matematika OK”}

const pesan = ‘Matematika OK’

Algoritma:
write (pesan)

PASCAL:

```
program Matematika_OK3;
{ Menampilkan kata matematika OK }


uses crt;

const pesan = 'Matematika OK';

begin
clrscr;

      write(pesan);
end.
```

Contoh 4.2. (Mempertukarkan nilai dari dua peubah)
Tuliskan algoritma yang membaca dua buah nilai untuk peubah X dan Y, lalu mempertukarkan nilai kedua peubah tersebut, kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Pertukaran
{Mempertukarkan nilai X dan Y. Nilai X dan Y dibaca terlebih dahulu}

Deklarasi
      X, Y, temp : integer

Algoritma:
      read (X, Y)   {baca nilai X dan Y terlebih dahulu}

      {proses pertukaran}
      temp ← X   {simpan nilai X di tempat penampungan sementara, temp}
      X   ← Y   {X diisi dengan Y}
      Y   ← temp   {isi Y dengan nilai X semula yang tadi disimpan di temp}
      write (X, Y)   {cetak nilai X dan Y setelah pertukaran}

PASCAL:

```
program Pertukaran;
{ Mempertukarkan nilai X dan Y. Nilai X dan Y dibaca terlebih dahulu }

(* Deklarasi *)
uses crt;

var
      X, Y, temp : integer;

(* Algoritma *)

begin
clrscr;

      { baca nilai X dan Y }
      write('X = '); readln(X);
      write('Y = '); readln(Y);

      { proses pertukaran }
      temp := X;
      X := Y;
      Y := temp;

      { tulis nilai X dan Y setelah pertukaran }
      writeln('X = ',X);
      writeln('Y = ',Y);
end.
```

Contoh 4.3. (Menghitung luas segitiga)
Tuliskan algoritma yang membaca peubah alas dan tinggi, lalu menghitung luas segitiga tersebut. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Luas_Segitiga
{Menghitung luas segitiga, dengan alas dan tinggi dibaca terlebih dahulu}

Deklarasi
      alas, tinggi, luas : real

Algoritma:
      read (alas, tinggi)
      luas← (alas * tinggi)/2
      write (luas)

PASCAL:

```
program Luas_Segitiga;
{ Menghitung luas segitiga dengan panjang alas dan tinggi dibaca terlebih dahulu }

(* Deklarasi *)
uses crt;

var
	alas, tinggi, luas : real;


(* Algoritma *)

begin
clrscr;

	{ baca nilai alas dan tinggi }
	write('Alas = '); readln(alas);
	write('Tinggi = '); readln(tinggi);

	{ proses menghitung luas }
	luas := 0.5 * alas * tinggi;

	{ tulis hasil perhitungan}
	writeln('Luas Segitiga = ', luas)

end.
```

Contoh 4.4. (Menghitung titik tengah dari dua buah titik)
Tuliskan algoritma yang membaca dua buah titik, lalu menghitung titik tengah dari kedua titik tersebut. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Titik_Tengah
{Menghitung titik tengah dari dua buah titik P1 = (x1, y1) dan P2 = (x2, y2). Titik tengah dicetak ke piranti keluaran}

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

Deklarasi

	P1, P2, P3 : Titik

Algoritma:

```
read (P1.x, P1.y)
read (P2.x, P2.y)
P3.x ← (P1.x + P2.x)/2
P3.y ← (P1.y + P2.y)/2
write (P3.x, P3.y)
```

PASCAL:

```
program Titik_Tengah;
{ Menentukan titik tengah antara titik P1(x,y) dan P2(x,y) }

(* Deklarasi *)
uses crt;

type Titik = record
               x : real;
               y : real;
               end;

var

      P1, P2, P3 : Titik;

(* Algoritma *)

begin
clrscr;

      { baca P1 }
      write('P1.x = '); readln(P1.x);
      write('P1.y = '); readln(P1.y);

       { baca P2}
      write('P2.x = '); readln(P2.x);
      write('P2.y = '); readln(P2.y);

      { proses menghitung titik tengah }
      P3.x := (P1.x + P2.x)/2;
      P3.y := (P1.y + P2.y)/2;

      { tulis hasil perhitungan}
      writeln('Titik Tengah Antara P1(',P1.x,',',P1.y,') dan P2(',P2.x,',',P2.y,')
adalah P3(',P3.x,',',P3.y,')');
end.
```

Dari hasil output contoh 4.4. coba dibuatkan tampilan yang lebih sederhana.

Latihan

1) Tuliskan algoritma untuk menghitung luas bangun geometri (bujursangkar, lingkaran, segitiga, trapesium, dan sebagainya). Data masukan dibaca dari piranti masukan dan luas bangun ditampilkan sebagai keluaran. Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.
2) Dibaca tiga buah bilangan bulat $x, y,$ dan z. Tuliskan algoritma untuk mempertukarkan tripel (x, y, z) menjadi (y, z, x). Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.

Bab 5
PEMILIHAN

Pemilihan merupakan struktur algoritma yang terdiri dari satu kasus, dua kasus, tiga kasus atau lebih.

1. Satu Kasus

Bentuk pernyataan untuk pemilihan satu kasus adalah:

```
if kondisi then
        pernyataan
endif
```

Contoh 5.1. (Mencetak "Ganjil" jika data masukan adalah bilangan ganjil)
Tuliskan algoritma yang membaca data masukan, dan mencetak pesan "ganjil" jika data tersebut adalah bilangan ganjil. Kemudian translasikan ke dalam bahasa Pascal.

```
Program Cetak_Ganjil
{Mencetak pesan "Ganjil" dari data yang dimasukkan}

Deklarasi
        x       : integer

Algoritma:
        read (x)
        if x mod 2 = 1 then
                write ('Ganjil')
        endif
```

PASCAL:

```
program Cetak_Ganjil;
{ Mencetak pesan ganjil jika data masukan berupa bilangan ganjil }


uses crt;

(* Deklarasi *)
var
      x : integer;


(* Algoritma *)
begin
clrscr;

      read (x);
      if x mod 2 = 1 then
            write('Ganjil');
end.
```

Contoh 5.2. (Menentukan nilai mutlak dari sebuah bilangan riil)
Tuliskan algoritma untuk menentukan nilai mutlak dari sebuah bilangan riil yang dimasukkan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Nilai_Mutlak
{Menentukan nilai mutlak dari bilangan riil yang dimasukkan}

Deklarasi
      x   : real

Algoritma:
      read (x)
      if x < 0 then
            x ← -x
      endif
      write (x)

PASCAL:

```
program Nilai_Mutlak;
{ Menentukan nilai mutlak dari data yang dimasukkan }


uses crt;

(* Deklarasi *)
var
      x : integer;


(* Algoritma *)
begin
clrscr;

      read (x);
      if x < 0 then
            x := -x;
      write('Nilai Mutlaknya adalah ',x);
end.
```

2. Dua Kasus

Bentuk pernyataan untuk pemilihan dua kasus adalah:

```
if kondisi then
      pernyataan 1
else
      pernyataan 2
endif
```

Contoh 5.3. (Mencetak jenis bilangan ganjil atau genap dari data masukan) Tuliskan algoritma yang mencetak pesan jenis bilangan yang dimasukkan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Jenis_Bilangan1
{Mencetak pesan jenis bilangan ganjil atau genap dari data yang dimasukkan}

Deklarasi
      x     : integer

Algoritma:
      read (x)
```

```
if x mod 2 = 1 then
        write (‘Ganjil’)
else
        write (‘Genap’)
endif
```

PASCAL:

```
program Jenis_Bilangan1;
{ Mencetak pesan jenis bilangan ganjil atau genap dari data masukkan }

uses crt;

var
    x : integer;

begin

clrscr;

    write('Masukkan bilangan : '); readln(x);

    if (x mod 2 = 1) then
        write('Ganjil')
    else
        write('Genap');

end.
```

Contoh 5.4. (Menentukan bilangan terbesar dari dua buah bilangan)
Tuliskan algoritma untuk menentukan bilangan terbesar dari dua bilangan yang dimasukkan terlebih dahulu. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Bilangan_Terbesar
{Menentukan bilangan terbesar dari dua bilangan yang dimasukkan}

Deskripsi
    A, B : integer

Algoritma:
    read (A, B)
    if A > B then
        write (‘Bilangan terbesar adalah ‘, A)
```

```
else
        write (‘Bilangan terbesar adalah ‘,B)
endif
```

PASCAL:

```
program Nilai_Terbesar;
{ Menentukan nilai terbesar dari dua nilai yang dimasukkan }

uses crt;

var
      A, B : integer;

begin

clrscr;

      write('Masukkan bilangan A: '); readln(A);
      write('Masukkan bilangan B: '); readln(B);

      if A > B then
            write('Bilangan terbesar adalah A = ', A)
      else
            write('Bilangan terbesar adalah B = ', B);

end.
```

Contoh 5.5. (Menentukan Nilai fungsi tangga)
Tuliskan algoritma untuk menentukan nilai dari fungsi tangga di bawah ini. Kemudian translasikan dalam bahasa Pascal.

$$f(x) = \begin{cases} x, & x < 0 \\ x - 1, & x \geq 0 \end{cases}$$

Program Fungsi_Tangga
{Menentukan nilai fungsi tangga $f(x) = x$ untuk $x < 0$ dan $f(x) = x - 1$ untuk $x \geq 0$, dengan x dibaca dari masukkan }

Deskripsi
      x, f(x) : real

Algoritma:
      read (x)
      if x < 0 then

```
            f(x) ← x
        else
            f(x) ← x - 1
        endif
        write (f(x))
```

PASCAL:

```
program Fungsi_Tangga;
{ Menentukan nilai fungsi tangga f(x)=x untuk x<0 }
{ f(x) = x - 1 untuk x>=0 }

uses crt;

var
    x, fungsi_x : real;

begin
clrscr;

    write('Masukkan nilai x : '); readln(x);

    if x < 0 then
        fungsi_x := x
    else
        fungsi_x := x - 1;
    write('Nilai f(x) = ', fungsi_x);

end.
```

3. Tiga Kasus atau Lebih

Bentuk pernyataan untuk pemilihan tiga kasus adalah:

```
if kondisi 1 then
        pernyataan 1
else
        if kondisi 2 then
                pernyataan 2
        else
                if kondisi 3 then
                pernyataan 3
                endif
        endif
endif
```

untuk kasus yang lebih dari tiga, mirip dengan bentuk di atas hanya ditambah berapa pernyataannya.

Contoh 5.6. (Menentukan jenis bilangan)
Tuliskan algoritma untuk menentukan jenis bilangan apakah negatif, positif, atau nol dari data masukan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Jenis_Bilangan2
{Menentukan jenis bilangan negatif, positif, atau nol, dari data masukan }

Deskripsi
        x       : real

Algoritma:
        read (x)
        if x < 0 then
                write ('Bilangan negatif')
        else
                if x > 0 then
                        write ('Bilangan positif')
                else
                        if x = 0 then
                                write ('Bilangan Nol')

                        endif
                endif
        endif
```

PASCAL:

```
program Jenis_Bilangan2;
{ Mencetak pesan jenis bilangan negatif, positif atau nol dari data masukkan }

uses crt;

var
      x : integer;

begin

clrscr;

      write('Masukkan bilangan : '); readln(x);
```

```
    if x < 0 then
        write('Bilangan Negatif')
    else
        if x > 0 then
            write('Bilangan Positif')
        else
            write('Bilangan Nol');

end.
```

Contoh 5.7. (Menentukan kuadran titik di bidang kartesius)
Tuliskan algoritma untuk menentukan kuadrat mana titik berada, di mana titik dibaca dari piranti masukan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Kuadran_Titik
{Menentukan kuadran dari sebuah titik P(x, y) yang dibaca dari piranti masukan}

Deskripsi
    type Titik : record <x : real, y : real>
    P   : Titik

Algoritma:
```
    read (P.x, P.y)
    if (P.x > 0) and (P.y > 0) then
        write ('Kuadran I')
    else
        if (P.x < 0) and (P.y > 0) then
            write ('Kuadran II')
        else
            if (P.x < 0) and (P.y < 0) then
                write ('Kuadran III')
            else
                if (P.x > 0) and (P.y < 0) then
                    write ('Kuadran IV')
                endif
            endif
        endif
endif
```

PASCAL:

```
program Kuadran_Titik;
{ Menentukan letak sebuah titik yang dibaca, apakah berada di kuadran I, II,
III, atau IV }

uses crt;

type Titik = record
              x : real;
              y : real;
             end;

var

       P : Titik ;

begin

clrscr;
       { Masukkan koordinat titiknya }
       write('Masukkan titik ');
       write('P.x = '); readln(P.x);
       write('P.y = '); readln(P.y);

       { Menentukan posisi titik }

       if (P.x > 0) and (P.y > 0) then
              write('Titik berada di Kuadran I')
       else
              if (P.x < 0) and (P.y > 0) then
                     write('Titik berada di Kuadran II')
              else
                     if (P.x < 0) and (P.y < 0) then
                            write('Titik berada di Kuadran III')
                      else
                            if (P.x > 0) and (P.y < 0) then
                                   write('Titik berada di Kuadran IV')
end.
```

4. STRUKTUR CASE

Penulisan IF-THEN-ELSE untuk kasus yang lebih banyak, dirasakan kurang efektif dalam hal penulisan, sehingga untuk lebih menyederhanakan penulisan digunakan struktur CASE. Bentuk penulisan dengan CASE adalah

```
case ekspresi
        nilai 1         : pernyataan 1
        nilai 2         : pernyataan 2
        nilai 3         : pernyataan 3
        nilai 4         : pernyataan 4
        .
        .
        .
        nilai n         : pernyataan n
        otherwise  : pernyataan x
endcase
```

Contoh 5.8. (Menu untuk empat persegi panjang)
Tuliskan algoritma untuk melakukan perhitungan terhadap hal yang berkaitan dengan persegi panjang sesuai dengan keinginan yaitu luas, keliling, dan panjang diagonal. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Persegi_Panjang
{Menentukan luas, keliling, atau diagonal persegi panjang berdasarkan pilihan}

Deskripsi
        NoPilih : integer;
        Panjang, lebar, luas, keliling, diagonal : real;

Algoritma:
        {Mencetak Menu untuk dipilih}
        write (' Menu Persegi Panjang ')
        write (' 1. Hitung Luas')
        write (' 2. Hitung Keliling')
        write (' 3. Hitung Panjang Diagonal')
        write (' 4. Keluar')
        write (' Masukkan pilihan anda (1/2/3/4) : ')
        read (NoPilih)

        case NoPilih
                1       : read (panjang, lebar)
                          luas  ← panjang * lebar
                          write (luas)

                2       : read (panjang, lebar)
                          keliling  ← 2 * (panjang + lebar)
                          write (keliling)
```

```
        3       : read (panjang, lebar)
                  diagonal  ← sqrt(panjang* panjang + lebar * lebar)
                  write (diagonal)

        4       : write ('Keluar')
    endcase
```

PASCAL:

```
program Persegi_Panjang;
{ Menentukan luas, keliling, atau diagonal persegi panjang berdasarkan pilihan
}

uses crt;

var
    NoPilih : integer;
    panjang, lebar, luas, keliling, diagonal : real;

begin

clrscr;
    { Mencetak Menu untuk dipilih }
    writeln('Menu Persegi Panjang');
    writeln('1. Hitung Luas');
    writeln('2. Hitung Keliling');
    writeln('3. Hitung Panjang Diagonal');
    writeln('4. Keluar');
    write('Masukkan pilihan anda 1/2/3/4 : ');
    readln(NoPilih);

    case NoPilih of
    1     : begin
            write('Masukkan panjang : '); readln(panjang);
            write('Masukkan lebar : '); readln(lebar);
            luas := panjang * lebar ;
            write('Luas Persegi panjang = ', luas);
            end;

    2     : begin
            write('Masukkan panjang : '); readln(panjang);
            write('Masukkan lebar : '); readln(lebar);
            keliling := 2*panjang + 2*lebar;
            write('Keliling Persegi Panjang = ', keliling);
            end;
```

```
      3       : begin
                write('Masukkan panjang : '); readln(panjang);
                write('Masukkan lebar : '); readln(lebar);
                diagonal := sqrt(panjang*panjang + lebar*lebar);
                write('Panjang diagonal = ', diagonal);
                end;

      4       : write('Keluar');

      end;
end.
```

Latihan

1) Buatlah algoritma yang membaca sebuah bilangan bulat positif lalu menentukan apakah bilangan tersebut merupakan kelipatan 4. Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.
2) Tulislah algoritma yang membaca tiga buah bilangan bulat, lalu mengurutkan tiga buah bilangan tersebut dari nilai yang kecil ke nilai yang besar. Keluaran adalah tiga buah bilanga terurut. Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.
3) Tulislah algoritma yang membaca panjang (*integer*) tiga buah sisi sebuah segitiga. $a, b,$ dan c, yang dalam hal ini $a \leq b \leq c$, lalu menentukan apakah ketiga sisi tersebut membentuk segitiga siku-siku, segitiga lancip, atau segitiga tumpul. Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.

Bab 6
PENGULANGAN

Struktur pengulangan secara umum terdiri dari dua bagian:
1. Kondisi pengulangan, yaitu ekspresi *boolean* yang harus dipenuhi untuk melaksanakan pengulangan.
2. Badan (*body*) pengulangan, yaitu bagian algoritma yang diulang.

Sturktur pengulangan biasanya diserta dengan bagian:
1. Inisialisasi, yaitu aksi yang dilakukan sebelum pengulangan dilakukan pertama kali
2. Terminasi, yaitu aksi yang dilakukan setelah pengulangan selesai dilaksanakan

Bentuk struktur pengulangan secara umum adalah

```
<inisialisasi>
awal pengulangan
        badan pengulangan
akhir pengulangan
<terminasi>
```

Ada tiga bentuk pernyataan pengulangan yang akan digunakan dalam buku ini yaitu:
1. Pernyataan FOR
2. Pernyataan WHILE
3. Pernyataan REPEAT

1. Pernyataan FOR

Pernyataan FOR ada dua macam, yaitu menaik (*ascending*) atau menurun (*descending*)

Bentuk umum pernyataan FOR menaik:

```
for pencacah ← nilai_awal to nilai_akhir do
        pernyataan
endfor
```

Bentuk umum pernyataan FOR menurun:

```
for pencacah ← nilai_akhir downto nilai_awal do
    pernyataan
endfor
```

Contoh 6.1. (Mencetak angka 1 sampai 20)
Tuliskan algoritma untuk mencetak angka 1 sampai 20. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Cetak1Sampai20_1
{Mencetak angka dari 1 sampai 20}

Deskripsi
    i : integer

Algoritma:
    for i ← 1 to 20 do
        write (i)
    endfor
```

PASCAL:

```
program Cetak1Sampai20_1;
{ Mencetak angka 1 sampai 20 }

uses crt;

var
    i : integer;

begin

clrscr;

    for i:=1 to 20 do
        write(i);
end.
```

Contoh 6.2. (Mencetak angka 1 sampai N)
Tuliskan algoritma untuk mencetak angka 1 sampai N, dengan N dibaca dari piranti masukan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Cetak1SampaiN_1
{Mencetak angka dari 1 sampai N}

Deskripsi
    N, i : integer

Algoritma:
    read (N)
    for *i* ← *1* to *N* do
        write (i)
    endfor

PASCAL:

```
program Cetak1SampaiN_1;
{ Mencetak angka 1 sampai N }

uses crt;

var
    i, N : integer;

begin

clrscr;
    read(N);
    for i:=1 to N do
        write(i);
end.
```

Contoh 6.3. (Menghitung deret 1 + 2 + 3 + ... + N)
Tuliskan algoritma untuk menghitung 1 sampai 20. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program HitungDeret1SampaiN_1
{Menghitung deret dari 1 sampai N}

Deskripsi
    N, i, jumlah : integer

Algoritma:
    read (N)
    jumlah ← 0
    for *i* ← *1* to *N* do
        jumlah ← jumlah + i

```
endfor
write (jumlah)
```

PASCAL:

```
program Deret1SampaiN_1;
{ Menghitung deret 1+2+3+...+N }

uses crt;

var
      i, N, jumlah : integer;

begin

clrscr;

      read(N);
      jumlah:=0;
      for i:=1 to N do
            jumlah := jumlah+i;
            write('Jumlah Deret = ', jumlah);
end.
```

Contoh 6.4. (Mencetak angka 20 sampai 1)
Tuliskan algoritma untuk mencetak angka 20 sampai 1. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Cetak20Sampai1
{Mencetak angka dari 20 sampai 1}

Deskripsi
      i : integer

Algoritma:
```
for i ← 20 downto 1 do
      write (i)
endfor
```

PASCAL:

```
program Cetak20Sampai1_1;
{ Mencetak angka 20 sampai 1 }

uses crt;

var
      i : integer;

begin

clrscr;
      for i:=20 downto 1 do
            write(i);
end.
```

2. Pernyataan WHILE

Bentuk umum pernyataan WHILE adalah

```
while kondisi do
      pernyataan
endwhile
```

Contoh 6.5. (Mencetak angka 1 sampai 20)
Tuliskan algoritma untuk mencetak angka 1 sampai 20. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Cetak1Sampai20_2
{Mencetak angka dari 1 sampai 20}

Deskripsi
      i : integer

Algoritma:
      i ← 1
    while i ≤ 20 do
          write (i)
          i ← i + 1
    endwhile
```

PASCAL:

```
program Cetak1Sampai20_2;
{ Mencetak angka 1 sampai 20 }

uses crt;

var
      i : integer;

begin

clrscr;
      i := 1;
      while i <= 20 do
            begin
            write(i);
            i := i + 1;
            end;
end.
```

Apa yang terjadi jika pada contoh 6.5 tidak ada “begin-end” dalam struktur while, coba diskusikan penyebabnya!

Contoh 6.6. (Mencetak angka 1 sampai N)
Tuliskan algoritma untuk mencetak angka 1 sampai N, dengan N dibaca dari piranti masukan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Cetak1SampaiN_2
{Mencetak angka dari 1 sampai N}

Deskripsi
      N, i : integer

Algoritma:
      read (N)
      i ← 1
      while $i \leq$ N do
          write (i)
          i ← i + 1
      endwhile

PASCAL:

```
program Cetak1SampaiN_2;
{ Mencetak angka 1 sampai N }

uses crt;

var
      i, N : integer;

begin

clrscr;
      read(N);
      i := 1;
      while i <= N do
            begin
            write(i);
            i := i + 1;
            end;
end.
```

3. Pernyataan REPEAT

Bentuk umum pernyataan REPEAT adalah

```
repeat
      pernyataan
until kondisi
```

Contoh 6.7. (Mencetak angka 1 sampai 20)
Tuliskan algoritma untuk mencetak angka 1 sampai 20. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

```
Program Cetak1Sampai20_3
{Mencetak angka dari 1 sampai 20}

Deskripsi
      i : integer

Algoritma:
      i ← 1
      repeat
            write (i)
            i ← i + 1
      until i > 20
```

PASCAL:

```
program Cetak1Sampai20_3;
{ Mencetak angka 1 sampai 20 }

uses crt;

var
      i : integer;

begin

clrscr;
      i := 1;
      repeat
            begin
            write(i);
            i := i + 1;
            end;
      until i > 20
end.
```

Contoh 6.8. (Mencetak angka 1 sampai N)
Tuliskan algoritma untuk mencetak angka 1 sampai N, dengan N dibaca dari piranti masukan. Kemudian translasikan algoritma tersebut dalam bahasa Pascal.

Program Cetak1SampaiN_3
{Mencetak angka dari 1 sampai N}

Deskripsi
      N, i : integer

Algoritma:
      read (N)
      i ← 1
      repeat
          write (i)
          i ← i + 1
      until i > N

PASCAL:

```
program Cetak1SampaiN_3;
{ Mencetak angka 1 sampai N }

uses crt;

var
	i, N : integer;

begin

clrscr;
	read(N);
	i := 1;
	repeat
		begin
		write(i);
		i := i + 1;
		end;
	until i > N
end.
```

Latihan

1) Buatlah algoritma untuk menghitung jumlah N buah bilangan ganjil pertama (yaitu 1 + 3 + 5 + ...). Catatan: N adalah bilangan bulat tidak negatif. Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.
2) Buatlah algoritma untuk menghitung jumlah bilangan ganjil dari 1 sampai N (1 + 3 + 5 + … + N). Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.
3) Tuliskan algoritma untuk menampilkan semua solusi bilangan bulat tidak negatif dari persamaan berikut:

   $$x + y + z = 25$$

   Yang dalam hal ini, $x \geq 0, y \geq 0,$ dan $z \geq 0$. Kemudian, translasikan algoritma tersebut dalam bahasa Pascal.

Bab 7
PROSEDUR

1. Struktur Prosedur

Prosedur adalah modul program yang mengerjakan tugas/aktivitas yang spesifik dan menghasilkan suatu efek netto (Liem, 1996).

Struktur prosedur adalah:

```
procedure NamaProsedur (deklarasi parameter, jika ada)
{ Spesifikasi prosedur, berisi penjelasan tentang apa yang dilakukan oleh prosedur ini.
K. Awal      : keadaan sebelum prosedur dilaksanakan.
K. Akhir     : keadaaan setelah prosedur dilaksanakan. }

Deklarasi
{ semua nama yang dipakai di dalam prosedur dan hanya berlaku lokal di dalam prosedur didefinisikan di sini }

Algoritma :
{ badan prosedur, berisi urutan instruksi }
```

Contoh 7.1. Menghitung luas segitiga
Tuliskan prosedur yang membaca panjang alas dan tinggi segitiga, menghitung luas segitiga dengan rumus luas = (alas x tinggi)/2, dan mencetak luas segitiga tersebut.

```
Procedure LuasSegitiga
{ menghitung luas segitiga dengan rumus L = (alas  x tinggi)/2 }
{ K. Awal : sembarang }
{ K. Akhir : luas segitiga tercetak }

Deklarasi
        alas   : real
        tinggi : real
        luas   : real

Algoritma:
```

read(alas, tinggi)
luas ← (alas * tinggi)/2
write (luas)

2. Pemanggilan Prosedur

Prosedur bukan program yang berdiri sendiri, instruksi-instruksi dalam prosedur hanya akan dieksekusi jika prosedur tersebut diakses. Untuk mengakses prosedur cukup dengan memanggil prosedur tersebut. Cara memanggil prosedur hanya dengan menuliskan nama prosedur tersebut. Misalnya untuk memanggil prosedur pada contoh 7.1. cukup dengan menuliskan *LuasSegitiga.*

3. Nama Lokal, Nama Global, dan Lingkup

Nama lokal merupakan nama-nama (konstanta, peubah, tipe, dan lain-lain) yang dideklarasikan di dalam prosedur. Nama lokal hanya dikenal di dalam prosedur yang bersangkutan.

Nama global merupakan nama-nama (konstanta, peubah, tipe, dan lain-lain) yang dideklarasikan di dalam program utama. Nama-nama global dapat digunakan dibagian mana saja dari program, baik di dalam program utama maupun dalam prosedur.

Kapan menggunakan nama lokal atau nama global, tergantung kepada penggunaan nama tersebut. Bila nama peubah tersebut digunakan di seluruh bagian program, maka nama peubah tersebut harus dideklarasikan global. Bila nama peubah hanya digunakan di dalam prosedur saja, maka nama peubah tersebut dideklarasikan secara lokal, namun apabila dideklarasikan secara global masih tetap benar.

4. Parameter

Parameter adalah nama-nama peubah yang dideklarasikan pada bagian header prosedur.

Terdapat tiga jenis parameter formal yang disertakan di dalam prosedur, yaitu:
1) Parameter masukan (*input parameter*), yaitu parameter yang nilainya berlaku sebagai masukan untuk prosedur.
2) Parameter keluaran (*output parameter*), yaitu parameter yang menampung keluaran yang dihasilkan oleh prosedur.
3) Parameter masukan/keluaran (*input/output parameter*), yaitu parameter yang berfungsi sebagai masukan sekaligus keluaran bagi prosedur tersebut.

Contoh 7.2. (Prosedur menghitung luas segitiga dengan parameter masukan)
Dari contoh 7.1. dituliskan menjadi parameter masukkan akan menjadi

```
Procedure LuasSegitiga (input alas, tinggi : real)
{ menghitung luas segitiga dengan rumus L = (alas  x tinggi)/2 }
{ K. Awal : alas dan tinggi sudah terdefinisi nilainya }
{ K. Akhir : luas segitiga tercetak }

Deklarasi
        luas   : real

Algoritma:
        luas ← (alas * tinggi)/2
        write (luas)
```

Contoh 7.3. (Prosedur menghitung luas segitiga dengan parameter keluaran)
Dari contoh 7.1. dituliskan menjadi parameter keluaran akan menjadi

```
Procedure LuasSegitiga (input alas, tinggi : real, output luas : real)
{ menghitung luas segitiga dengan rumus L = (alas  x tinggi)/2 }
{ K. Awal : alas dan tinggi sudah terdefinisi nilainya }
{ K. Akhir : luas berisi luas segitiga }

Deklarasi
        { tidak ada }

Algoritma:
        luas ← (alas * tinggi)/2
```

Latihan

1) Tulislah prosedur untuk menghitung jumlah *N* buah bilangan genap pertama (bilangan genap dimulai dari 0). Prosedur menerima (parameter) masukan *N* dan memberikan (parameter) keluaran jumlah *N* buah bilangan genap pertama.
2) Tulislah prosedur yang menghasilkan nilai rata-rata sekumpulan data bilangan bulat yang dibaca secara berulang-ulang dari papan ketik (akhir pembacaan adalah 9999). Prosedur memiliki parameter keluaran, yaitu nilai rata-rata yang dihasikan.
3) Ulangi soal nomor 2 tetapi prosedur menghasilkan nilai terkecil.

Bab 8
FUNGSI

1. Struktur Fungsi

Fungsi adalah program yang memberikan/mengembalikan (return) sebuah nilai dari tipe tertentu (tipe dasar atau tipe bentukan).

Struktur fungsi adalah:

function *NamaFungsi* (input *deklarasi parameter, jika ada*) → *tipe*
{ spesifikasi fungsi, menjelaskan apa yang dilakukan dan yang dikembalikan oleh fungsi. }

Deklarasi
{ semua nama yang dipakai di dalam fungsi dan hanya berlaku lokal didefinisikan di sini }

Algoritma:
{ badan fungsi, berisi instruksi-instruksi untuk menghasilkan nilai yang akan dikembalikan oleh fungsi }

    return *ekspresi*   { pengembalian nilai yang dihasilkan fungsi }

Contoh 8.1. (Fungsi $F(x) = 2x^2 + 3x - 5$)
Tuliskan fungsi untuk menghasilkan nilai $F(x) = 2x^2 + 3x - 5, \quad x \in \boldsymbol{R}$

function F (input x : real) → real
{ Mengembalikan nilai $F(x) = 2x^2 + 3x - 5, \quad x \in \boldsymbol{R}$ }

Deklarasi
{ tidak ada }

Algoritma:
    return 2*x*x + 3*x - 5

Contoh 8.2. (Fungsi perpangkatan x^m)
Tuliskan fungsi untuk menghitung perpangkatan x^m, $m \geq 0$, $m \in \boldsymbol{R}$.

function Pangkat (input x : real, input m : integer) → real
{ Mengembalikan nilai perpangkatan x^m }

Deklarasi
    p : real
    i : integer

Algoritma:
    p ← 1
    for i ← 1 to m do
        p ← p * x
    endfor
    return p

Contoh 8.3. (Fungsi faktorial)
Tuliskan fungsi untuk menghitung nilai faktorial dari bilangan bulat tidak negatif.

function Faktorial (input n : integer) → integer
{ Mengembalikan nilai n!, untuk n≥0 }

Deklarasi
    f : integer
    i : integer

Algoritma:
    f ← 1
    for i ← 1 to n do
        f ← f * i
    endfor
    return f

Contoh 8.4. (Fungsi titik tengah)
Tuliskan fungsi untuk menghitung titik tengah dari dua buah titik.

function TitikTengah (input P1, P2 : Titik) → Titik
{ Mengembalikan titik tengah dari P1 dan P2 }

Deklarasi

Pt : Titik

Algoritma:
Pt.x ← (P1.x + P2.x)/2
Pt.y ← (P1.y + P2.y)/2
return Pt

Dari contoh 8.4., perhatikan tipe Titik, coba diskusikan dimana tipe ini di deklarasikan.

2. Pemanggilan Fungsi

Fungsi diakses dengan cara memanggil namanya dari program utama, diikuti dengan daftar parameter aktual (bila ada). Karena fungsi menghasilkan nilai, maka nilai tersebut dapat diperlakukan dengan dua cara. Pertama, nilai yang dikembalikan oleh fungsi ditampung di dalam sebuah peubah yang bertipe sama dengan tipe fungsi.

Peubah ← NamaFungsi (parameter aktual, jika ada);

Latihan

1) Buatlah fungsi jarak yang menerima masukan dua buah titik $P_1(x, y)$ dan $P_2(x, y)$ dan menghitung jarak kedua titik tersebut.
2) Buatlah fungsi phytagoras yang menerima tiga buah bilangan bulat $a, b, c,$ dan menentukan apakah ketiga bilangan tersebut merupakan tripel phytagoras.

DAFTAR PUSTAKA

Liem, Inggriani. 1996. *Diktat Kuliah Algoritma dan Pemrograman Prosedural*. Bandung : Teknik Informatika ITB.

Lien, Inggriani. 1996. Program Kecil. Bandung : Teknik Informatika ITB.

Munir, Renaldi. 2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C Edisi ke-3*. Bandung : Informatika.

Sanjaya, Alwin. 2003. *Cepat Mahir Bahasa Pascal*. IlmuKomputer.com