Modul Praktek Laboratorium Komputer

Pascal

Di susun oleh :

Team Penyusun Modul Pascal

Akademi Manajemen Informatika & Komputer

BINA SARANA INFORMATIKA

Jakarta

2006

BAB I

PENGENALAN PASCAL

1.1. Sejarah Singkat Pascal

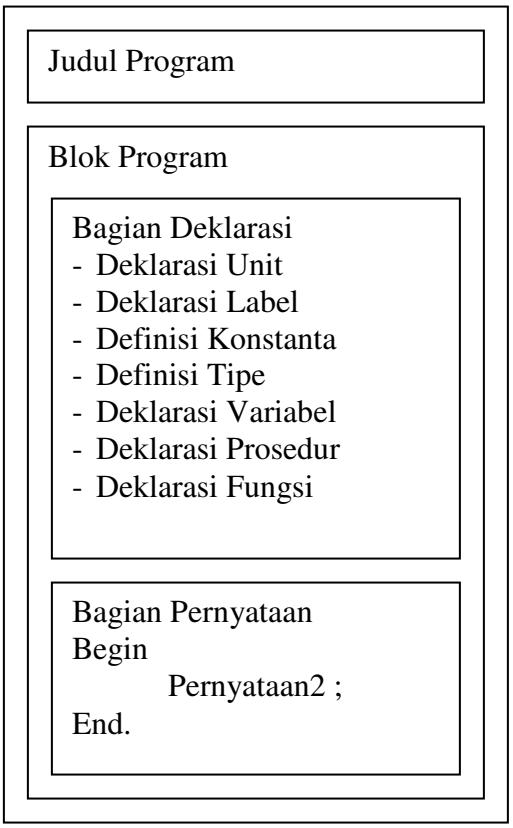
- Dirancang oleh ***Prof. Nicklaus Wirth*** dari Technical University di Zurich, Switzerland tahun 1971.
- Nama Pascal berasal dari ***Blaise Pascal***, nama ahli matematika dan philosopi dari Perancis (abad 17).
- Pengembangan dari bahasa Algol 60 dan Algol W (turunan Algol 60).
- Memiliki beberapa versi, seperti : *Turbo Pascal, Ms Pascal (Microsoft), Apple Pascal, UCSD (University of California at San Diego Pascal), dll.*
- Turbo Pascal yang dibuat oleh Borland Inc. adalah versi yang paling banyak digunakan karena menggunakan Compiler untuk menterjemahkannya dan juga mengikuti standard bahasa Pascal yang dibuat oleh Nicklaus Wirth dan K. Jensen.
- Pascal merupakan bahasa pemrograman tingkat tinggi (high level language) dan terstruktur (Structured Programming language).

1.2. Struktur Program Pascal

- **Judul Program (Program Heading)** bersifat optional (boleh digunakan/tidak), tetapi sebaiknya digunakan karena mencantumkan nama program.
- **Blok Program (Program Block)** atau Badan Program (Program Body), terdiri dari :
  - **Bagian deklarasi (Declaration Part)** : untuk menyiapkan elemen-elemen program, seperti seperti nama konstanta, variable, label, tipe, prosedur dan fungsi serta penggunaan unit.

- **Bagian Pernyataan (statement part)** : untuk menunjukkan suatu tindakan yang akan dikerjakan oleh program. Diawali ***Begin*** dan diakhiri ***End.***.

- Setiap akhir pernyataan diakhiri titik koma (;), kecuali untuk nama label.
- Akhir program diberi titik (.).



1.3. IDE (Integrated Developement Environment)

Langkah awal dari belajar Visual Basic adalah mengenal IDE (*Integrated Developement Environment*) Visual Basic yang merupakan Lingkungan Pengembangan Terpadu bagi programmer dalam mengembangkan aplikasinya.

Dengan menggunakan IDE programmer dapat membuat *user interface*, melakukan koding, melakukan testing dan *debuging* serta mengkompilasi program menjadi *executable*. Penguasaan yang baik akan IDE akan sangat membantu programmer dalam mengefektifkan tugas-tugasnya sehingga dapat bekerja dengan efisien.

Program Pascal Sederhana

- Hanya terdiri bagian Pernyataan saja.
- Program ini tidak melaksanakan apa-apa, karena tidak mengandung pernyataan (empty statement).

```
Begin
End.
```

Program Pascal Lengkap

```
Program Contoh_Lengkap(Input,Output) ;

Uses  CRT;
Label Akhir ;
Const Phi = 3.14 ;
Type  Bil_Nyata = Real ;
Var   Jari_jari : Bil_Nyata ;

Procedure Hitung_Luas ( Radius : Bil_Nyata ) ;
Begin
     Writeln (‘ Luas = ‘, 0.5 * Phi * Radius * Radius ) ;
End ;

Begin
   Clrscr ;
   Jari_jari := 10 ;
   Hitung_Luas ( Jari_jari ) ;
   Goto Akhir ;
   Writeln ( ‘ Lho, kok saya dilewati ! ‘ ) ;
   Akhir :
   Writeln ( ‘ Selesai ! ‘ ) ;
End.
```

Komentar Program

- Adalah keterangan yang diberikan untuk keperluan dokumentasi.
- Tidak menghasilkan tindakan (tidak mempengaruhi jalannya program).

- Boleh menggunakan tanda : **{** ini komentar **}** atau **(*** ini komentar ***)**

```
{ Ini awal Program }
Begin
     Writeln ( ‘ Bahasa ‘ ) ;
     Writeln ( ‘ Pascal ‘ ) ;
End.
(* Akhir Program *)
```

BAB II

INPUT-OUTPUT, VARIABEL DAN KONTANTA, RESERVED WORD

2.1. Perintah Input Output

2.1.1. Perintah Read dan Readln

- Digunakan untuk meminta masukan dari keyboard untuk diolah komputer.
- Tipe data yang dicetak dapat berupa Integer, Real, Character Striing ataupun Boolean.
- Perbedaan Read dan Readln adalah setelah meminta masukan. Jika Readln akan diakhiri dengan pindah baris, sedangkan pada Read tidak.

2.1.2. Perintah Write dan Writeln

- Digunakan untuk mencetak hasil proses. Tipe data yang dicetak dapat berupa Integer, Real, Character Striing ataupun Boolean.
- Perbedaan Write dan Writeln adalah setelah mencetak. Jika Writeln akan diakhiri dengan pindah baris, sedangkan pada Write tidak.

```
Program Contoh_Readln_&_Writeln;
Var
   Nama            : String [25];
   Nilai_Akhir     : Integer;
   Nilai_Rata2     : Real;
   Grade           : Char;
   Keterangan      : String [5];
Begin
   (* Input Data *)
   Write ('Masukkan Nama : ');Readln (Nama);
   Write ('Masukkan Nilai Akhir : ');Readln(Nilai_Akhir);
   Write ('Masukkan Nilai Rata2 : ');Readln(Nilai_Rata2);
   Write ('Masukkan Grade   : ');Readln(Grade) ;
   Write ('Masukkan Keterangan : ');Readln(Keterangan);
   (*Menampilkan Data *)
```

```
    Writeln ('Nama Siswa adalah ', Nama );
    Writeln ('Nilai Akhir adalah  ', Nilai_Akhir ) ;
    Writeln ('Nilai Rata-rata adalah ', Nilai_Rata2 ) ;
    Writeln ('Gradenya adalah ', Grade ) ;
    Writeln ('Keterangannya adalah ', Keterangan ) ;
End.
```

2.2. Identifier (Pengenal)

- Adalah nama yang dibuat oleh programmer yang berfungsi sebagai nama pengenal dari suatu elemen program, seperti nama-nama untuk judul program, variable, konstanta, label, prosedur, fungsi, dll.
- Syarat-syarat penamaan suatu identifier :
  - Karakter pertama huruf
  - Karakter kedua dan seterusnya boleh huruf, angka, garis bawah
  - Tidak boleh menggunakan karakter khusus (kecuali. Garis bawah), seperti : . , - * / @ ! > % dsb
  - Tidak boleh mengandung spasi/blank
  - Panjang nama bebas, tetapi hanya 63 karakter awal yang signifikan

    Contoh :

Identifier yang Benar :	Identifier yang salah :
SegiTiga	Segi Tiga
Segitiga	Segi 3
Segi3	Segi-Tiga
Segi_3	
Segi_Tiga	

2.3. Deklarasi Variabel

- Variabel adalah Suatu tempat di dalam memori komputer yang dapat menyimpan nilai/data yang berubah-rubah.
- Variabel bersifat sementara, jika komputer dimatikan semua variabel akan hilang. Variabel hanya dipakai saat program dijalankan.

```
Program Contoh_Variabel ;
Var
   Nama  : String [25] ;
   Nilai_Akhir      : Integer ;
   Nilai_Rata2      : Real ;
   Grade       : Char ;
      Keterangan    : String [5];
Begin
   Nama  := 'Andarii Maulana' ;
   Nilai_Akhir := 87 ;
   Nilai_Rata2      := 87.25 ;
   Grade := 'A' ;
   Keterangan := 'Lulus' ;
   Writeln ( 'Nama siswa adalah ', Nama ) ;
   Writeln ( 'Nilai Akhir adalah  ', Nilai_Akhir ) ;
   Writeln ( 'Nilai Rata-rata adalah ', Nilai_Rata2 ) ;
   Writeln ( 'Gradenya adalah ', Grade ) ;
   Writeln ( 'Keterangannya adalah ', Keterangan ) ;
End.
```

2.4. Deklarasi Konstanta

- Konstanta adalah Suatu nilai/data bersifat tetap (tidak dapat berubah) yang disimpan di dalam memori dan dapat diambil nilai/datanya bila dipanggil.
- Konstanta sering digunakan dalam rumus fisika dan matematika.

```
Program Contoh_Konstanta ;
Const
     Nama     = 'Andarii Maulana' ;
     Nilai_Akhir = 87 ;
     Nilai_Rata2   = 87.25 ;
     Grade    = 'A' ;
     Keterangan    = 'Lulus' ;
Begin
     Writeln ( 'Nama siswa adalah ', Nama ) ;
     Writeln ( 'Nilai Akhir adalah  ', Nilai_Akhir ) ;
     Writeln ( 'Nilai Rata-rata adalah ', Nilai_Rata2 ) ;
     Writeln ( 'Gradenya adalah ', Grade ) ;
     Writeln ( 'Keterangannya adalah ', Keterangan ) ;
End.
```

2.5. Reserved Word (Kata Tercadang)

- Adalah kata-kata yang sudah didefinisikan oleh Pascal dan mempunyai arti tertentu.
- Kata-kata tersebut tidak boleh digunakan sebagai identifier (Pengenal).

  Contoh : Program, Begin, End, If, For, While, Repeat, Write, Read, dsb.

2.6. Penggunaan Unit Crt

Unit yang mengatur kerja layar dan keyboard atau I/O. Harus menggunakan perintah uses crt untuk menggunakannya. Perintah yang terdapat dalam unit ini antara lain :

assigncrt	clreol	clrscr	delay	delline	gotoxy	highvideo
lowvideo	sound	insline	normvideo	textbackground	keypressed	nosound
textcolor	textmode	wherex	window	wherey	readky	

```
Program Hapus_Layar ;
Uses CRT ;
Begin
   Clrscr ;
   Writeln ( ‘ Bahasa ‘ ) ;
   Writeln ( ‘ Pascal ‘ ) ;
End.
```

BAB III

TIPE DATA SEDERHANA & OPERATOR

3.1. Tipe Data Sederhana

Tipe data menunjukkan suatu nilai yang dapat digunakan oleh variable. Tipe data sederhana terbagi menjadi beberapa bagian :

A. Tipe data Char (karakter)

- Terdiri dari satu huruf besar/kecil, angka (tidak untuk dihitung), atau karakter khusus
- Ditulis diantara 2 tanda petik tunggal.

  Contoh: ‘A’ ‘a’ ‘5’ ‘@’

B. Tipe data String (Untai)

- Berupa rangkaian karakter yang terletak diantara 2 tanda petik
- Panjang dari suatu string sebaiknya disebuntukan pada bagian deklarasi dengan tanda [n], jika tidak panjangnya dianggap 255 karakter.

  Contoh: ‘Budi’ ‘ Jl. Kramat Raya No. 18’ ‘3100413’

C. Tipe data Boolean

Berupa nilai logika, yaitu :

- True untuk menyatakan kondisi Benar
- False untuk menyatakan kondisi Salah

D. Tipe data Integer (Bil. Bulat)

- Adalah tipe bilangan yang tdk memiliki bagian desimal.

- Termasuk tipe numerik, yaitu dapat dioperasikan secara matematik.

Tipe	Ukuran Memori	Jangkauan
ShortInt	1 byte	-128 .. 127
Byte	1 byte	0 .. 255
Integer	2 byte	-32768 .. 32767
Word	2 byte	0 .. 65535
LongInt	4 byte	-2147483648 .. 2147483647

Tipe integer menyediakan konstanta standar MaxInt yang bernilai 32767 dan MaxLongInt yang bernilai 2147483647.

E. Tipe Data Real (Pecahan)

- Adalah tipe bilangan yang memiliki bagian desimal.

- Termasuk tipe numerik, yaitu dapat dioperasikan secara matematik.

Tipe	Ukuran Memori	Jangkauan	Digit Signifikan
Single	4 byte	1.5E-45 .. 3.4E+38	7 – 8
Real	6 byte	2.9E-39 .. 1.7E+38	11 – 12
Double	8 byte	5.0E-324 .. 1.7E+308	15 – 16
Extended	10 byte	1.9E-4951 .. 1.1E +4932	19 – 20

```
Program Contoh_Tipe_Data ;
Var
     Nilai_Akhir : Integer ;
     Nilai_Rata2    : Real ;
     Grade          : Char ;
     Keterangan     : String [5] ;
     Kondisi   : Boolean ;
Begin
     Nilai_Akhir := 87 ;
     Nilai_Rata2:= 87.25 ;
     Grade:= 'A' ;
     Keterangan:= 'Lulus' ;
     Kondisi:=  True ;
     Writeln('Nilai Akhir adalah  ', Nilai_Akhir);
     Writeln('Nilai Rata-rata adalah ', Nilai_Rata2);
```

```
        Writeln('Gradenya adalah ', Grade);
        Writeln('Keterangannya adalah ', Keterangan);
        Writeln('Kondisinya adalah ', Kondisi);
End.
```

3.2. Operator

Operator (tanda operasi) pada bahasa Pascal dkelompokkan dalam 9 kategori.

a. Assignment Operator (Operator pengerjaan)

- Menggunakan simbol titik dua diikuti tanda sama dengan (:=).

  Contoh :

```
A := B Nilai := 10      Grade := 'A'     Nama := 'Budi'
```

b. Binari Operator (operator Biner)

- Digunakan untuk mengoperasikan 2 buah operand untuk operasi aritmatika yang berhubungan dengan tipe Integer dan Real.
- Operand dapat berbentuk konstanta ataupun variable.

Operator	Operasi	Tipe Operand	Tipe Hasil
*	Perkalian	Real * Real Integer * Integer Real * Integer	Real Integer Real
DIV	Pembagian Bulat	Integer DIV Integer	Integer
/	Pembagian Real	Real / Real Integer / Integer Real / Integer	Real Real Real
MOD	Modulus (Sisa Pembagian)	Integer MOD Integer	Integer
+	Pertambahan	Real + Real Integer + Integer Real + Integer	Real Integer Real
-	Pengurangan	Real – Real Integer – Integer Real – Integer	Real Integer Real

```
Program Operator_Binari ;
```

```
Begin
    Writeln ( 15 * 5 ) ;
    Writeln ( 20 / 30 ) ;
    Writeln ( 20 DIV 3 ) ;
    Writeln ( 20 MOD 3 ) ;
End.
```

c. Unary Operator (Operator tunggal)

Berupa unary minus (untuk nilai negatif) dan unary plus (untuk nilai positif).

Contoh : `-5  +2.5  a+(-b)  a+(+b)`

d. Bitwise Operator

Digunakan untuk operasi bit per bit pada nilai integer.

– **Operator NOT**

Digunakan untuk pembalikan bitwise (nilai bit), 0 menjadi 1 dan 1 menjadi 0.

```
Program Operator_NOT ;
Begin
 Writeln (NOT 0) ;
End.
```

Proses :

- Nilai 0 disimpan di memori dalam bentuk : 0000000000000000
- NOT akan membalik 0 menjadi 1 : 1111111111111111

  Bit awal adalah sign bit yang menunjukkan :

  positif (bila 0) dan negatif (bila 1). 1111111111111111
- Nilai tsb dikurangi 1 : 1

  1111111111111110
- Semua nilai bit dibalik : 0000000000000001
- Hasilnya : -1 (bit awal 0, shg negatif)

- **Operator AND**

Digunakan untuk membandingkan 2 elemen, hasilnya akan benar jika keduanya benar.

A	B	A AND B
1	1	1
1	0	0
0	1	0
0	0	0

```
Program Operator_AND ;
Begin
     Writeln ( 12 AND 23 ) ;
End.
```

Proses :

- 12 Nilai Binarinya adalah : 0000000000001100

- 23 Nilai Binarinya adalah : 0000000000010111

- Hasilnya : 0000000000000100 → 4

- **Operator OR**

Digunakan untuk membandingkan 2 elemen, hasilnya akan benar jika salah satu atau keduanya benar.

A	B	A OR B
1	1	1
1	0	1
0	1	1
0	0	0

```
Program Operator_OR ;
Begin
     Writeln ( 12 OR 23 ) ;
End.
```

Proses :

- 12 Nilai Binarinya adalah : 0000000000001100

- 23 Nilai Binarinya adalah : 0000000000010111

- Hasilnya 0000000000011111 → 31

– **Operator XOR (Xclusive OR)**

Digunakan untuk membandingkan 2 elemen. Hasilnya akan benar bila salah satu saja benar.

A	B	A OR B
1	1	1
1	0	0
0	1	0
0	0	0

```
Program Operator_XOR ;
Begin
     Writeln ( 12 XOR 23 ) ;
End.
```

Proses :

- 12 Nilai Binarinya adalah : 0000000000001100

- 23 Nilai Binarinya adalah : 0000000000010111

- Hasilnya 0000000000011011 → 27

– **Operator SHL (Shift Left)**

Digunakan untuk menggeser (shift) sejumlah bit kekiri (Left) dengan bit 0.

```
Program Operator_Shl ;
Begin
 Writeln ( 5 Shl 6 ) ;
End.
```

Proses :

- 5 Nilai binarinya adalah : 0000000000000101

- Digeser 6 bit ke kiri menjadi : 0000000101000000 → 320

– **Operator SHR (Shift Right)**

Digunakan untuk menggeser (shift) sejumlah bit kekanan(Right) dengan bit 0.

```
Program Operator_Shl ;
Begin
 Writeln ( 160 Shl 6 ) ;
End.
```

Proses :

- 160 Nilai binarinya adalah : 0000000010100000

- Digeser 6 bit ke kiri menjadi : 0000000000000010 → 2

e. Relational Operator (Operator Relasi)

Digunakan untuk membandingkan hubungan antara 2 buah operand dan akan didapatkan hasil tipe Boolean, yaitu True atau False.

Operator	Operasi	Operator	Operasi
=	Sama dengan	<	Lebih kecil dari
< >	Tidak sama dengan	<=	Lebih kecilsama dengan dari
>	Lebih besar dari	IN	Seleksi dari anggota himpunan
>=	Lebih besar sama dengan dari		

```
Program Operator_Relasi ;
Var  a, b : Integer ;
Begin
     A := 5 ;   B := 3 ;
     Writeln ( A = B ) ;        Writeln ( A < > B ) ;
     Writeln ( A > B ) ;        Writeln ( A <= B ) ;
End.
```

f. Logical Operator (Operator Logika)

Ada 4 macam, yaitu : NOT, AND, OR dan XOR.

Bentuk operator ini samadengan bitwise operator, tetapi bekerja dengan nilai logika, yaitu True dan False.

```
Program Operator_Logika ;
Begin
   Writeln (Not True) ;  Writeln (True AND False) ;
   Writeln (True OR False) ; Writeln ( True XOR False) ;
End.
```

g. Addariess Operator (Operator Alamat)

- Operator ini berhubungan dengan alamat (addariess) di memori, yaitu :
- @ : Addariess-of Operator → alamat dari suatu nilai variable
- ^ : Indirection Operator → Nilai di alamat yang ditunjukkan.
- Operator ini akan digunakan pada pembahasan mengenai Pointer.

h. Set Operator (operator Himpunan)

Digunakan untuk operasi himpunan.

Operator	Operasi
+	Union
-	Perbedaan Himpunan
*	Perkalian Himpunan

Operator ini akan digunakan pada pembahasan mengenai Himpunan.

i. String Operator

Digunakan untuk operasi string dan hanya memiliki 1 operator saja, yaitu **+** yang digunakan untuk menggabungkan 2 buah nilai string.

```
Program Operator_String ;
Var Nama1, Nama2, Nama3 : String [15] ;
Begin
    Nama1 := ‘Arief’ ; Nama2 := ‘Budiman’ ;
    Nama3 := Nama1 + Nama2 := ‘Arief Budiman’ ;
    Writeln (Nama3) ;
End.
```

BAB IV

FUNGSI & PROSEDUR STANDAR

4.1. Fungsi Standar Aritmatika

a) Fungsi ABS (Absolut)

Digunakan untuk memutlakkan suatu nilai, yaitu nilai negatif dipositifkan dan nilai positif tetap positif.

B.U : ABS (X)

Argumen X dapat berupa Tipe Real atau Integer dan hasilnya sesuai tipe argumennya.

```
Program Fungsi_ABS ;
Var
   X : Real ;
Begin
   Write (‘Berapa nilai yang akan dimutlakkan :’) ;
   Readln (Nilai) ;
   Writeln (‘Nilai Mutlaknya = ‘, ABS(X) : 9 : 2 ) ;
End.
```

b) Fungsi EXP (Exponential)

Digunakan untuk menghitung nilai pangkat dari bilangan e, yaitu $e^{x.}$

B.u : EXP(X : real) : real ;

Argumen X dapat berbentuk tipe real atau integer dan hasilnya bertipe Real.

```
Program Fungsi_EXP ;
Var     X : Real ;
Begin
   Writeln ( ‘ Nilai yang diexponenkan : ‘) ; Readln (X) ;
   Writeln ( ‘ Nilai Exponennya = ‘, EXP(X) : 9 : 5 ) ;
End.
```

c) Fungsi LN (Logarithm Natural)

Digunakan untuk menghitung nilai logaritma alam (natural logarithm) dari nilai X.

B.u : LN(X : Real) : Real ;

Argumen X dapat berbentuk tipe real atau integer dan hasilnya bertipe Real.

```
Program Fungsi_EXP ;
Var
   X : Real ;
Begin
   Writeln ( ‘ Nilai yang diexponenkan : ‘) ; Readln (X) ;
   Writeln ( ‘ Nilai Exponennya = ‘, EXP(X) : 9 : 5 ) ;
End.
```

d) Fungsi SQR (SQuaRe) dan SQRT (SQuare Root)

SQR digunakan untuk mencari nilai pangkat kuadrat dan SQRT untuk mencari akar kuadrat.

B.u : SQR(X) SQRT(X)

Argumen X dapat bertipe Integer atau Real dengan hasil bertipe sesuai dengan argumennya.

```
Program SQR_dan_SQRT ;
Var     A, B : Real ;
   C, D: Integer ;
Begin
   A := 3 ;   B := SQR (A);
   Writeln (‘ Nilai kuadariat dari 3 adalah ‘, B) ;
   C := 25 ;  D := SQRT(B);
   Writeln (‘Akar kuadariat dari 25 adalah ‘, C ) ;
End.
```

e) Fungsi PI

Digunakan untuk mendapatkan nilai Pi, yaitu 3.141592653897932385.

```
Program Lingkaran ;
Uses CRT ;
Var
   r, L, K : Real ;
Begin
   Clrscr ;
   Write (‘Masukkan jari-jari Lingkaran (cm) : ‘ );
   Readln ( r ) ;
   L := PI * SQR( r ) ;
   K := 2 * PI * PI ;
   Writeln (‘ Luas Lingkaran adalah ‘, L :10:2, ‘Cm’);
   Writeln (‘ Keliling Lingkaran adalah : ‘), K:10:2 , ‘Cm’);
End.
```

f) Fungsi INT (Integer)

Digunakan untuk menghasilkan nilai integer, yaitu pembuatan ke bawah (nilai pecahan dibuang).

B. u : Int (X : Real) : Real ;

```
Program Fungsi_INT ;
Var     X : Real
Begin
   Angka := 450.654 ;
   Writeln (‘ Nilai pembulatannya adalah ‘, INT(Angka :9:2) ;
End.
```

g) Fungsi TRUNC dan FRAC

TRUNC digunakan untuk menghsilkan bilangan bulat dengan cara membuang bagian desimal suatu bilangan real. Fungsi FRAC mengambil bagian desimalnya.

B. u : FRAC(X : Real) : Real ;

TRUNC(X : X : Real) : LongInt ;

```
Program Fungsi TRUNC_dan_FRAC ;
Var     A, B : Real ;
Begin
   A := TRUNC (1.5) ;    Writeln (A) ;
   B := FRAC (1.5); Writeln(B);
End.
```

h) Fungsi ROUND

Digunakan untuk membulatkan nilai Real ke bilangan bulat yang terdekat.

B.u : ROUND(X) : Real) : LongInt ;

```
Program Fungsi_Round ;
Begin
   Nilai1 := ROUND(10/3) ;     Nilai2 := ROUND(20/3) ;
   Writeln (‘Hasil pembulatan 10/3 adalah ‘, Nilai1);
   Writeln (‘Hasil pembulatan 20/3 adalah’, Nilai2);
End.
```

i) Fungsi ORD dan CHR

Fungsi ORD : menghasilkan kode desimal suatu karakter pada table ASCII. Sedangkan fungsi CHR akan menghasilkan karakter ASCII dari suatu bilangan.

B.u : CHR(X : byte) : Char ;

ORD (X) : LongInt ;

```
Program Fungsi_ORD_dan CHR ;
Var
   X : byte ; Y : Char ;
Begin
   X := ORD('A') ;
   Writeln('Kode ASCII huruf A adalah ', X);
   Y := CHR(65);
   Writeln('Karakter dari kode ASCCI 65 adalah', Y) ;
End.
```

4.2. Fungsi Standar Operasi String

a) Copy

Fungsi Copy digunakan untuk menyalin atau mengcopy sejumlah karakter mulai posisi tertentu.

B U : Copy(*s, p, j*)

Ket : s = string yang akan disalin p = posisi awal penyalinan/copy

J = jumlah karakter yang disalin

b) Concat

Fungsi Concat digunakan untuk merangkai beberapa string (sama seperti +).

B U : Concat(s1,s2,…sn)

Ket : s1,s2,sn = string yang akan dirangakai

c) Pos

Fungsi Pos digunakan untuk mencari posisi string di dalam string lainnya, hasilnya berupa nilai byte, bila tidak ada hasilnya 0.

B U : Pos(s1,s2)

Ket : s1 = string/karaker yang akan dicari letaknya

s2 = string tempat pencarian

d) Length

Fungsi length digunakan untuk menghitung jumlah/panjang karakter yang ada pada suatu string.

B U : Length(s)

Ket : s = string/karaker yang akan dihitung

e) Upcase

Digunakan untuk merubah karakter menjadi huruf besar

B.U : UpCase(Ch :Char)

BAB V

TIPE & LABEL

5.1. Tipe

Pengenal (identifier) dari data yang digunakan harus diperkenalkan Tipe-nya. Jika ingin menggunakan tipe data dengan nama yang dibuat pemakai, maka harus disebutkan tipe data standarnya.

```
Program Contoh_Tipe ;
Type
    Bil_Bulat = Integer ;
Var
    Jumlah : Bil_Bulat ;
Begin
    Jumlah := 10 ;
    Writeln ( ' Jumlah : ', Jumlah ) ;
End.
```

5.2. Label

- Label harus dideklarasikan dahulu pada bagian deklarasi.
- Nama label boleh berupa string (Cth: Selesai) atau nilai integer (Cth : 100).
- Label digunakan sebagai arah tujuan dari perintah Goto.
- Goto adalah perintah untuk meloncat ke suatu statement tertentu.
- Perintah Goto harus diikuti nama label yang dituju.

```
Program Contoh_Label(Layar) ;
Label
   100, Selesai ;
Begin
   Writeln ( ' Bahasa ' ) ;
   Goto 100 ;
   Writeln ( ' Pemrograman ' ) ;
   100 :
   Writeln ( ' Tingkat ' ) ;
   Goto Selesai ;
   Writeln ( ' Tinggi ' ) ;
   Selesai :
End.
```

BAB VI

FORMAT KELUARAN

6.1. Format Tipe Data

a) Tampilan Default

Adalah tampilan yang mengikuti bentuk yang sudah ditetapkan Pascal.

- **Tipe Character, String, Boolean, Integer**

  Tipe2 data tsb akan ditampilkan dalam bentuk tidak mengandung blank di muka ataupun blank di belakang.

- **Tipe Real**

  Tampilan nilai real menempati posisi 18 digit dengan menggunakan bentuk Eksponential.

  B.U. : bd.ddddddddddEsYY

  Dengan : b : blank jika positif, - bila negatif

  d : digit

  s : + jika positif – bila Negatif

b) Tampilan Terformat

Untuk mengatur bentuk tampilan dari tampilan default ke bentuk yang diinginkan.

- **Parameter Char : n**

  Digunakan untuk tipe data karakter, membentuk tampilan char selebar n dengan blank dimuka sebanyak n-1.

```
Program Tampil_Karakter ;
Var   Huruf : Char ;
Begin
```

```
     Huruf := 'A' ;
     Writeln ('Tampilan huruf tanpa Format :' , Huruf) ;
     Writeln ('Tampilan huruf dengan Format :' , Huruf :5);
End.
```

- **Parameter String : n**

  Digunakan untuk tipe data string, membentuk tampilan String dengan lebar n karakter.

```
Program Tampil_Boolean ;
Var   Kondisi : Boolean ;
Begin
     Kondisi := True ;
     Writeln ('Tampilan huruf tanpa Format :' , Kondisi) ;
     Writeln ('Tampilan huruf dengan Format :' ,Kondisi:8);
End.
```

- **Parameter Boolean : n**

  Digunakan untuk tipe data Boolean, membentuk tampilan nilai boolean selebar n karakter rata sebelah kanan.

```
Program Tampil_Boolean ;
Var   Kondisi : Boolean ;
Begin
     Kondisi := True ;
     Writeln ('Tampilan huruf tanpa Format :' , Kondisi) ;
     Writeln ('Tampilan huruf dengan Format :',Kondisi:8);
End.
```

- **Parameter Integer : n**

  Digunakan untuk format tampilan nilai numeric bulat dengan lebar n digit rata sebelah kanan.

```
Program Tampil_Integer ;
Var   Jumlah : Integer ;
Begin
     Jumlah := 85 ;
     Writeln ('Tampilan angka tanpa Format :' , Jumlah) ;
     Writeln ('Tampilan angka dengan Format :' ,Jumlah:5);
End.
```

- **Parameter Real : n : m**

  Digunakan untuk membentuk format tampilan nilai pecahan dengan lebar n digit rata sebelah kanan, dengan m digit angka di belakang koma, tidak dalam bentuk eksponential.

  Ket : (Titik desimal dihitung 1 digit).

```
Program Tampil_Real ;
Var   Nilai : Real ;
Begin
     Nilai := 55.64 ;
     Writeln ('Tampilan angka tanpa Format :' , Nilai) ;
     Writeln ('Tampilan angka terformat :' , Nilai:10:2);
End.
```

6.2. Prosedur Pemberian Warna

a) Warna Foreground

B.u : Textcolor(color : byte)

Kode warna boleh menggunakan kode atau menyebuntukan warnanya

Warna	Kode	Warna	Kode	Warna	Kode
Black	0	Brown	6	LightRed	12
Blue	1	LightGray	7	LightMagenta	13
Green	2	DarkGray	8	Yellow	14
Cyan	3	LightBlue	9	White	15
Red	4	LightGreen	10	Blink	128
Magenta	5	LightCyan	11		

```
Program Tampil Warna_ForeGround ;
Uses CRT ;
Begin
     Clrscr ;
     TextColor (Blue);Writeln('Tulisan ini berwarna biru');
     TextColor (4); Writeln ('Kalau ini berwarna merah') ;
     TextColor (Brown+8) ; Writel ('Warna Kuning') ;
     TextColor (Red+Blink);Writeln(Warna Merah Berkedip');
End.
```

b) Warna BackGround

B.u : TextBackGround(color : byte)

Kode warna boleh menggunakan kode atau menyebuntukan warnanya

Warna	Kode	Warna	Kode	Warna	Kode
Black	0	Cyan	3	Brown	6
Blue	1	Red	4	LightGray	7
Green	2	Magenta	5		

```
Program Tampil_BackGround ;
Uses CRT ;
Begin
   Clrscr ;
   TextColor (Yellow) ; TextBackGround (red) ;
   TextColor (7) ; TextBackGround (0) ;
   Writeln (‘ Kembali ke Normal ‘ ) ;
End.
```

6.3. Prosedur Penentuan Lokasi Kursor

Prosedur2 standar pengaturan layar ini bila digunakan harus menyebuntukan dahulu unit standar CRT.

a) Prosedur CLRSCR (Clear Screen)

Digunakan untuk membersihkan layar dari tampilan2 sebelumnya dan meletakkan kursor di ujung kiri atas layar.

```
Program Coba_Clrscr  ;
Uses CRT ;
Begin
   Writeln ( ‘Bahasa’ ) ;
      Clrscr ;
      Writeln ( ‘Pascal’ ) ;
End.
```

b) Prosedur GOTOXY

Digunakan untuk meletakkan kursor di posisi layar yang ditunjukkan oleh nilai : X (kolom 1 s.d 80) dan Y (Baris 1 s.d 25).

```
Program Coba_GotoXY  ;
Uses CRT ;
Begin
   Clrscr ;
   GotoXY (20, 15) ;
      Writeln ( ‘Pascal’ ) ;
End.
```

c) Prosedur CLREOL (Clear End Of Line)

Digunakan untuk menghapus semua karakter dalam satu baris disebelah kanan posisi kursor.

```
Program Coba_Clreol ;
Uses CRT ;
Var   Nilai : Integer ;
Begin
  Clrscr ;   GotoXY(10, 15) ;
  Writel ('Masukkan sebuah nilai : ') ; Readln ( Nilai ) ;
  GotoXY ( 10, 15 ) ; ClrEol ;
  Writeln (' Anda Pintar !' ) ;
End.
```

d) Prosedur DELLINE (Delete Line)

Digunakan untuk menghaspus sebuah baris di posisi kursor dan menggeser naik tampilan baris2 di bawahnya.

```
Program Tampil_Delline ;
Uses CRT ;
Var     Tekan : Char ;
Begin
  Clrscr ;
  GotoXY (5, 8) ; Writeln ('Satu' ) ;
  GotoXY (5, 9) ; Writeln ('Dua') ;
  GotoXY (5,10) ; Writeln ('Tiga') ;
  GotoXY (5, 20) ;
     Write ('Tekan Sembarang Tombol') ; Read (Tekan);
     GotoXY (5,20) ; ClrEol ;
     GotoXY (5,9 ; DelLine
End.
```

e) Prosedur INSLINE (Insert Line)

Digunakan untuk menyisipi sebuah baris pada posisi kursor dan menggeser ke bawah tapilan baris2 di bawahnya.

```
Program Tampil_Insline ;
Uses CRT ;
Var     Tekan : Char ;
Begin
  Clrscr ;
  GotoXY (5, 8) ; Writeln ('Satu' ) ;
  GotoXY (5, 9) ; Writeln ('Dua') ;
  GotoXY (5,10) ; Writeln ('Tiga') ;
  GotoXY (5, 20) ;
     Write ('Tekan Sembarang Tombol') ; Read (Tekan);
```

```
      GotoXY (5,20) ; ClrEol ;
      GotoXY (5,9 ; InsLine
End.
```

f) Prosedur Delay

Digunakan untuk menghentikan sejenak proses program selama nilai argumen tempo, yaitu dalam ukuran millisecond (1/1000 detik).

```
Program Tampil Warna_ForeGround ;
Uses CRT ;
Begin
   Clrscr ;
   TextColor (Blue) ; Writeln ('Tulisan ini berwarna biru') ;
   TextColor (4); Writeln ('Kalau ini berwarna merah') ;
   TextColor (Brown+8) ; Writel ('Warna Kuning') ;
   TextColor (Red + Blink); Writeln (Warna Merah Berkedip') ;
 End.
```

BAB VII

BRANCHING

(PENYELEKSIAN KONDISI)

Untuk menyeleksi kondisi di dalam pascal, menggunakan statement sebagai berikut :

7.1. Statemen IF

a) Struktur If …Then

B. u : `If` *Kondisi* `Then` *Statemen*

Kondisi adalah syarat yang diseleksi.

Bila kondisi benar (terpenuhi), maka statemen akan dikerjakan.

```
Program Seleksi_IF_1 ;
Var
   Nilai : Real ;
   Ket : String [5] ;
Begin
   Ket : 'Gagal' ;
   Write ('Berapa Nilai yang didapat ? '); Readln (Nilai) ;
   If Nilai > 60 Then Ket := 'Lulus' ;
   Writeln ('Keterangannya : ', Ket ) ;
End.
```

b) Struktur If …Then …Else

B. u : `If` *kondisi* `Then`

Statemen1 ;

`Else`

Statemen2 ;

Bila kondisi benar (terpenuhi), maka statemen1 akan dikerjakan, sedangkan bila kondisi salah (tidak terpenuhi), maka statemen2 yang akan dikerjakan.

```
Program Seleksi_IF_2 ;
Var
   Nilai : Real ;
   Ket : String [5] ;
```

```
Begin
  Write (‘Berapa Nilai yang didapat ? ‘); Readln (Nilai) ;
  If Nilai > 60 Then
          Ket := ‘Lulus’
  Else
       Ket := ‘Gagal’ ;
  Writeln (‘Keterangannya : ‘, Ket ) ;
End.
```

Ket : Statemen di atas Else jangan diberi titk koma (;)

c) Struktur IF Tersarang

Adalah suatu Statemen IF yang berada dalam lingkungan statemen If yang lain.

B.u :

```
If Kondisi1 Then
      If Kondisi2 Then
            Statemen1
      Else
            Statemen2 ;
```

```
If Kondisi1 Then
Begin
      If Kondisi2 Then
            Statemen1
            Else
            Statemen2
End ;
```

```
If Kondisi1 Then
      If Kondisi2 Then
            Statemen1;
      Else
            Statemen2;
Else
      Statemen3 ;
```

```
If Kondisi1 Then
Begin
      If Kondisi2 Then
      Begin
        If Kondisi3 Then
              Statemen1
            Else
              Statemen2 ;
         End ;
      End ;
End ;
```

```
Program Seleksi_IF_3 ;
Var
     Nilai : Real ;
     Grade : Char ;
Begin
     Write (‘Berapa Nilai yang didapat ? ‘);
     Readln (Nilai) ;
     If Nilai > 90 Then
          Grade := ‘A’;
     Else If Nilai > 75 Then
                Grade := ‘B’;
           Else If  Nilai > 60 Then
                   Grade := ‘C’
                Else If Nilai > 40 Then
                     Grade := ‘D’;
                Else
                     Grade := ‘E’;
                  EndIf
           EndIf
     EndIf
     EndIf
     Writeln (‘Keterangannya : ‘, Ket ) ;
End.
```

7.2. Statemen Case

Digunakan untuk memilih dengan kemungkinan lebih dari 2.

a) Statemen Case – Of

B.u : **Case** <*variable*> **Of**

<*Konstanta1*> : <*Pernyataan*>

<*Konstanta2*> : <*Pernyataan*>

…

<*Konstanta_n*> : <*Pernyataan*>

End;

```
Program Case_1 ;
Var
    Ukuran : Char ;
    Banyak : Integer ;
    Harga, Jumlah : Real ;
Begin
     Write(‘Ukuran Jaket (S?M?L) : ‘); Readln(Ukuran) ;
     Write(‘Banyak Jaket : ‘); Readln(Banyak);
```

```
        Case Ukuran Of
              'S' : Harga := 1000 ;
              'M' : Harga := 1250 ;
              'L' :Harga := 15000 ;
        End ;
        Jumlah := Banyak * Harga ;
        Writeln('Jumlah dibayar : Rp ', Jumlah:8:0);
  End.
```

b) Statemen Case Of - Else

B.u : **Case** <*variable*> **Of**

<*Konstanta1*> : <*Pernyataan*>

<*Konstanta2*> : <*Pernyataan*>

…

Else

<*Konstanta_n*> : <*Pernyataan*>

End;

```
Program Case_2 ;
Var   Ukuran : Char ;
      Banyak : Integer ;
      Harga, Jumlah : Real ;
Begin
      Write('Ukuran Jaket (S?M?L) : '); Readln(Ukuran) ;
      Write('Banyak Jaket : '); Readln(Banyak);
      Case Ukuran Of
            'S' : Harga := 1000 ;
            'M' : Harga := 1250 ;
            'L' :Harga := 15000 ;
      End ;
      Jumlah := Banyak * Harga ;
      Writeln('Jumlah dibayar = Rp ', Jumlah:8:0);
End.
```

BAB VIII

LOOPING

(PERINTAH PERULANGAN)

Iterasi / perulangan (Loop) dalam bahasa Pascal terdiri dari 3 macam, yaitu : For … Do, While … Do dan Repeat … Until.

8.1. For … Do

- Digunakan untuk mengulang statemen atau satu blok statemen berulang kali sejumlah yang ditentukan.
- Perulangan For dapat berbentuk perulangan positif, perulangan Negatif dan perulangan tersarang.

a) Perulangan Positif

Adalah perulangan dengan penghitung (counter) dari kecil ke besar atau pertambahannya positif.

B.u : `For` *Variabel_Kontrol* `:=` *Nilai_Awal* `To` *Nilai_Akhir* `Do`

Statement `;`

Ket : - Variabel_Kontrol, Nilai_Awal dan Nilai_Akhir harus bertipe sama, yaitu Integer.

- Jika Statement hanya 1, maka boleh ditulis dalam blok (Diawali Begin dan diakhiri End;) boleh tidak. Sedangkan jika blok statement lebih dari 1, maka statement2 tsb harus diletakkan dalam blok.

Contoh 1 :

```
Program Perulangan_FOR_1 ;
Var  I : Integer ;
Begin
```

```
     FOR I := 1 To 5 Do Writeln ('Pascal');
End.
```

Contoh 2 :

```
Program Perulangan_FOR_2 ;
Var  I : Integer ;
Begin
     FOR I := 1 To 5 Do
          Begin
               Writeln ('Pascal');
          End ;
End.
```

Contoh 3 :

```
Program Perulangan_FOR_3 ;
Var  I : Integer ;
Begin
     FOR I := 1 To 5 Do
          Begin
               Writeln (I );
               Writeln ('Pascal');
          End ;
End.
```

b) Perulangan Negatif

Adalah perulangan dengan penghitung (counter) dari besar ke kecil atau pertambahannya negatif.

B.u : **For** *Variabel_Kontrol* := *Nilai_Awal* **DownTo** *Nilai_Akhir* **Do**

Statement ;

```
Program Perulangan_FOR_4;
Var  I : Integer ;
Begin
     FOR I := 1 To 5 Do Writeln ('Pascal');
End.
```

c) Perulangan Tersarang (Nested Loop)

Adalah perulangan yang berada dalam perulangan lainnya. Perulangan yang lebih dalam akan diproses terlebih dahulu sampai habis, kemudian perulangan yang luar

baru bertambah, mengerjakan perulangan yang lebih dalam lagi mulai dari awal, dan seterusnya.

```
Program Perulangan_FOR_5 ;
Var  I, J : Integer ;
Begin
     For I := 1 To 3 Do
          Begin
          For J := 1 To 3 DO
               Write (I:8, J :3) ;
          Writeln ;
          End ;
End.
```

8.2. While … Do

Digunakan untuk melakukan proses perulangan suatu statemen atau blok statemen terus menerus selama kondisi ungkapan logika pada While masih bernilai logika benar.

B.u : **While** *ungkapan_logika* **Do**

Statemen;

```
Program Perulangan_While ;
Var  I : Integer ;
Begin
     I := 0 ;
     While I < 5 Do
     Begin
          Writeln (I) ;
          I := I + 1 ;
     End ;
End.
```

Perulangan While-Do Tersarang

Adalah suatu perulangan While-Do yang ada didalam perulangan While_Do yang lain.

```
Program Perulangan_While ;
Var  I, J : Integer ;
Begin
     I := 1 ;
     While I < 3 Do
     Begin
          J := 1 ;
          While J < 2     Do
          Begin
               Writeln ( I : 5 , J : 5 );
               J := J + 1 ;
          End ;
```

```
            I := I + 1 ;
      End ;
End.
```

8.3. Repeat … Until

Digunakan untuk mengulang (Repeat) statemen satau blok statemen sampai (Until) kondisi yang diseleksi di Until tidak terpenuhi.

B.u : **Repeat**

Statemen;

Until *Ungkapan_logika*;

```
Program Perulangan_Repeat ;
Var   I : Integer ;
Begin
      I := 0 ;
      Repeat
            I := I + 1 ;
            Writeln (I) ;
      Until I = 5;
End.
```

Repeat-Until Tersarang

Adalah suatu perulangan Repeat – Until yang berada didalam perulangan Repeat-Until yang lain.

```
Program Perulangan_Repeat_2 ;
Var   I, J : Integer ;
Begin
      I := 0 ;
      Repeat
            I := I + 1 ;
            J := 0 ;
            Repeat
                  J := J + 1 ;
                  Writeln (I : 5, J : 5) ;
            Until J = 3 ;
      Until j = 3 ;
End.
```

Perbandingan Repeat-Until dengan While-Do

Struktur Repeat – Until	Struktur While-Do

Var I : Integer ;
Begin
I := 10 ;
Repeat
Writeln (I) ;
I := I + 1 ;
Until I > 5 ;
End.
	Var I : Integer ;
	Begin
	I := 10 ;
	While I < 5 Do
	Begin
	Writeln (I) ;
	I := I + 1 ;
	End;
	End.

1) Paling sedikit Statemen2 didalam perulangan Repeat-Until diproses sekali, karena seleksi kondisi ada di bawah, sedangkan pada struktur While-Do paling sedikit dikerjakan nol kali, karena seleksi kondisi terletak di atas.
2) Pada While-Do blok statemen diawali dengan Begin dan End untuk menunjukkan batas perulangannya, sedangkan pada Repeat-Untiltidak diperlukan Begin dan End krn batasnya jelas (Diawali Repeat dan diakhiri End).
3) Pada While-Do perulangan dilaksanakan terus selama kondisi ungkapan bernilai Benar, sedangkan pada Repeat-Until akan dilaksanakan terus selama kondisi ungkapan bernilai salah.

Fungsi Standard Pada Perulangan

a) Fungsi INC (Increment)

Digunakan untuk meningkatkan nilai suatu angka.

B.u : INC(X[n:LongInt)

b) Fungsi DEC (Decrement)

Digunakan untuk menurunkan nilai suatu angka.

B.u : INC(X[n:LongInt)

BAB IX

ARRAY

Array (Larik) adalah tipe tersruktur yang terdiri dari sejumlah komponen yang mempunyai tipe yang sama. Array ada 2 jenis:

1) Array berdimensi satu.
2) Array berdimensi 2 /dimensi banyak.

9.1. Array Berdimensi Satu (One Dimensional Array)

Bentuk Umum :

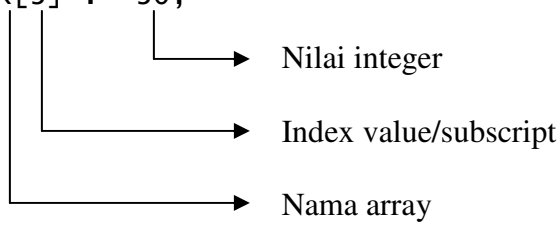
```
Nama_array = ARRAY [Tipe index] of tipe data;
```

Contoh Penulisan :

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

Sebagian dari elemen-elemen dari X tersebut adalah :

```
X[1] := 10;
X[2] := 20;
X[3] := 30;
```



Contoh :

```
Program Array_1_dimensi;
Var
     NilaiPrak       : array[1..20] of real;
     I, JumlahData   : Byte;
Begin
```

```
      Write ('Masukkan banyakya data : ');
      Readln(JumlahData);
      For I := 1 to JumlahData do
      Begin
            Write ('Nilai ke ',I,' : ");
            Readln(NilaiPrak[I])
      End;
      Readln;
End.
```

9.2. Array Berdimensi Dua (Two/multi Dimensional Array)

Bentuk umum :

Nama_array : **ARRAY**[*tipe-indeks1,type-indeks2*] **of** *tipe data*

Contoh penulisan :

```
      tabel  : array [1..3,1..2] of byte
```

Contoh :

```
Program Array_2_dimensi;
Var
      Matrik : array[1..3,1..2] of shortint;
      I, J : Byte;
Begin
      Matrik[1,1] := -11;
      Matrik[2,1] := -76;
      Matrik[3,1] := 8;
      Matrik[1,2] := -1;
      Matrik[2,2] := 11;
      Matrik[3,3] := 18;
      For I := 1 to 3 do
      Begin
            For J := 1 to 2 do
                  Write(Matrik[I,J]):5);
            Writeln;
      End;
      Readln;
End.
```

BAB X

PROSEDUR

10.1. Deklarasi Prosedur

Prosedur adalah suatu program terpisah dalam blok tersendiri yang berfungsi sebagai sub program (program bagian) dan diawali dengan kata cadangan Procedure.

Bentuk Umum Prosedur :

```
Procedure nama (daftar_parameter);
Bagian deklarasi;
Bagian pernyataan;
```

10.2. Parameter dalam Procedure

a) *Parameter Bersifat Lokal* artinya bahwa nilai yang terdapat didalam suatu modul program hanya dapat digunakan pada modul atau unit program yang bersangkutan saja sehingga tidak dapat digunakan pada modul atau unit program lain.

Contoh program

```
Procedure Kali;
Var
     A, B : Byte;
Begin
     Write ('Isi nilainya : '); Readln (A);
     B := A * A;
End;
Begin
     Kali;
     Writeln ('Nilai B = ', B);
End.
```

b) *Parameter Bersifat Global* adalah kebalikan dari lokal. Agar nilainya dapat digunakan untuk beberapa atau semua modul/unit program maka nilai tersebut harus dideklarasikan diatas modul yang akan menggunakannya.

Contoh program

```
Var
     A, B : Byte;
Procedure Kali;

Begin
     Write ('Isi nilainya : '); Readln (A);
     B := A * A;
End;
Begin
     Kali;
     Writeln ('Nilai B = ', B);
End.
```

Istilah Di Dalam Parameter

a) Actual parameter (parameter nyata) yaitu parameter yang dikirimkan dari modul utama ke modul prosedur

b) Formal Parameter (parameter formal) yaitu parameter yang ada dan dituliskan pada judul prosedur

c) Parameter Passing yaitu proses Pemanggilan data lewat parameter nyata ke parameter formal.

d) By Value yaitu Pemanggilan parameter secara nilai

e) By Reference yaitu Pemanggilan parameter secara acuan

f) Value Parameter yaitu parameter-parameter yang digunakan dalam Pemanggilan secara nilai

10.3. Pemanggilan Parameter

a) Pemanggilan secara nilai (by Value)

Pemanggilan parameter secara nilai bersifat searah yaitu dari parameter nyata ke parameter formal. Bila nilai parameter formal berubah, maka nilai parameter nyata tidak berubah.

Contoh program

```
Procedure Hitung (X, Y, Z : Byte);
Begin
   Z : = X + Y;
   Writeln (‘Nilai X = ‘, X);
   Writeln (‘Nilai Y = ‘, Y);
   Writeln (‘Nilai Z = ‘, Z);
End;
Var
   A, B, C : Byte;
Begin
   A := 5; B:= 7; C:=3;
   Hitung (A, B, C);
   Writeln (‘Nilai A = ‘,A,’Nilai B = ‘,B,’Nilai C = ‘,C);
   Readln;
End.
```

b) Pemanggilan secara Acuan (by reference)

Pemanggilan parameter secara acuan bersifat dua arah (bolak – balik). Bila nilai parameter formal berubah, maka nilai parameter nyata ikut berubah.

Contoh program

```
Procedure Hitung (Var X, Y, Z : Byte);
Begin
   Z : = X + Y;
   Writeln (‘Nilai X = ‘, X);
   Writeln (‘Nilai Y = ‘, Y);
   Writeln (‘Nilai Z = ‘, Z);
End;
Var
   A, B, C : Byte;
Begin
   A := 5; B:= 7; C:=3;
   Hitung (A, B, C);
   Writeln (‘Nilai A = ‘,A,’Nilai B = ‘,B,’Nilai C = ‘,C);
   Readln;
End.
```

c) Pemanggilan parameter sebagian secara nilai dan sebagian secara acuan

Contoh program

```
Procedure Hitung (X, Y : Byte; Var Z : Byte);
Begin
   Z : = X + Y;
End;
Var
   A, B, C : Byte;
Begin
   A := 5; B:= 7; C:=3;
   Hitung (A, B, C);
   Writeln ('Nilai X = ', X);
   Writeln ('Nilai Y = ', Y);
   Writeln ('Nilai Z = ', Z);
   Readln;
End.
```

BAB XI

FUNGSI

11.1. Deklarasi Fungsi

Fungsi secara garis besar sama dengan procedure yang membedakannya adalah nama fungsi harus dideklarasikan dengan type datanya.

Bentuk Umum

```
Function Identifier  (daftar-parameter):
Type;
```

Contoh Penulisan :

```
Function  faktorial (var fak, hasil  : I nteger) : integer;
```

11.2. Parameter pada Fungsi

Sifat parameter dalam fungsi sama dengan sifat parameter dalam prosedur, yaitu bersifat lokal dan global.

Contoh program (parameter bersifat lokal)

```
Function Kali: Byte;
Var
     A, B : Byte;
Begin
     Write ('Isi nilainya : '); Readln (A);
     B := A * A;
End;
Begin
     Kali;
     Writeln ('Nilai B = ', B);
End.
```

Contoh program (parameter bersifat global)

```
Var
     A, B : Byte;
Function Kali: Byte;
Begin
```

```
      Write (‘Isi nilainya : ‘); Readln (A);
      B := A * A;
End;
Begin
      Kali;
      Writeln (‘Nilai B = ‘, B);
End.
```

11.3. Pemanggilan Fungsi

Parameter dalam fungsi (idem dengan prosedur), yaitu dapat dilakukan pemanggilan secara nilai (by Value) atau secara acuan (by reference)

Contoh program (by Value)

```
Function Hitung (X, Y, Z : Byte): Byte;
Begin
   Z : = X + Y;
   Writeln (‘Nilai X = ‘, X);
   Writeln (‘Nilai Y = ‘, Y);
   Writeln (‘Nilai Z = ‘, Z);
End;
Var
   A, B, C : Byte;
Begin
   A := 5; B:= 7; C:=3;
   Hitung (A, B, C);
   Writeln (‘Nilai A = ‘,A,’Nilai B = ‘,B,’Nilai C = ‘,C);
   Readln;
End.
```

Contoh program (by Reference)

```
Function Hitung (Var X, Y, Z : Byte): Byte;
Begin
   Z : = X + Y;
   Writeln (‘Nilai X = ‘, X);
   Writeln (‘Nilai Y = ‘, Y);
   Writeln (‘Nilai Z = ‘, Z);
End;
Var
   A, B, C : Byte;
Begin
   A := 5; B:= 7; C:=3;
   Hitung (A, B, C);
   Writeln (‘Nilai A = ‘,A,’Nilai B = ‘,B,’Nilai C = ‘,C);
   Readln;
End.
```

BAB XII

UNIT

Unit adalah kumpulan dari konstanta, tipe data, variabel, prosedur dan fungsi.

12.1. Struktur Unit

Struktur unit terdiri dari :

```
Judul unit (unit header)
Bagian penghubung (interface section/interface part)
Bagian penerapan (implementation section/implementation part)
Bagian initialisasi (initialization section/initialization part)
End
```

Keterangan :

- Judul Unit menentukan nama unit yaitu nama yang digunakan dalam klausa USES. Contoh : UNIT Barang (barang adalah nama unit yang dimaksud)
- Bagian Penghubung adalah bagian untuk mendeklarasikan konstanta, tipe, variabel, prosedur dan fungsi yang bersifat publik. Bagian in diawali dg dengan kata tercadang INTERFACE.
- Bagian penerapan adalah bagian yang mendefinisikan tubuh dari semua prosedur dan fungsi publik. Bagian in diawali dengan kata tercadang implementation.
- Bagian inisialisasi adalah bagian terakhir dari suatu unit.bagian in diawali dengan kata tercadang Begin dan diakhiri dengan kata tercadang END.

12.2. Pembuatan Unit pada Pascal

```
Interface
     Uses  crt;
Var
     namaperusahaan   : string;
```

```
Procedure hapuslayar;
Function kapital (st : string) : string;

Procedure hapuslayar;
Begin
     clrscr;
end;

Function kapital  ( st : string) : string;
Var
     I        : integer;
     temp  : string;
begin
     temp  := '  ';
     for I  :=  1  to length  (st)  do
          temp  :=  temp + upcase(st[I]);
     kapital  := temp;
end.
```

12.3. Prosedur & Fungsi dalam File Unit

Pascal menyediakan beberapa unit standar, diantaranya :

- **Unit System**

  Merupakan pustaka/library dari proses pengerjaan pascal yang mendukung semua proses yang dibutuhkan pada saat pengerjaan program. Secara otomatis digunakan di dalam program, sehingga boleh tidak disebutkan. Perintah yang terdapat dalam unit ini antara lain :

break	abs	arctan	continue	cos	exp	exit
frac	int	halt	runerror	chr	ord	round
trunc	concat	insert	str	freemem	memavail	assigned
ofs	sptr	fillchar	lo	paramstr	sizeof	upcase
blockread	close	ln	sin	sqrt	inc	low
pred	copy	length	val	getmem	new	cseg
ptr	sseg	hi	move	random	swap	append
blockwrite	eof	pi	sqr	dec	high	odd
succ	delete	pos	dispose	maxavail	addr	dseg
seg	exclude	include	paramcount	randomize	typeof	assign
chdir	eoln	erase	filepos	filesize	flush	mkdir
getdir	ioresult	read	readln	rename	reset	rewrite
rmdir	seek	seekof	seekeoln	settextbuf	truncate	write
writeln						

- **Unit Crt**

Unit yang mengatur kerja layar dan keyboard atau I/O. Harus menggunakan perintah uses crt untuk menggunakannya. Perintah yang terdapat dalam unit ini antara lain :

assigncrt	clreol	clrscr	delay	delline	gotoxy	highvideo
lowvideo	sound	insline	normvideo	textbackground	keypressed	nosound
textcolor	textmode	wherex	window	wherey	readky	

```
Program Hapus_Layar ;
Uses CRT ;
Begin
     Clrscr ;
     Writeln ( ‘ Bahasa ‘ ) ;
     Writeln ( ‘ Pascal ‘ ) ;
End.
```

- **Unit Dos**

Unit ini berkaitan dengan dos. Harus menggunakan perintah uses dos untuk menggunakannya. Perintah yang terdapat dalam unit ini antara lain :

getdate	packtime	settime	intr	diskfree	fsearch	findnext
envcount	exec	dosversion	setcbreak	getftime	setdate	unpacktime
msdos	disksize	fsplit	getfattr	envstr	keep	getcbreak
setverivy	gettime	settime	getintvec	setintvec	fexpand	findfirst
setfattr	gatenv	swapvectors	getverivy			

```
Program Sisa_Isi_Disk ;
Uses DOS ;
Begin
   Writeln ( DiskFree(0), ‘byte isi disk’ ) ;
End.
```

- **Unit Graph**

Unit yang berorientasi pada pembuatan grafik. Harus menggunakan perintah uses graph untuk menggunakannya. Perintah yang terdapat dalam unit ini antara lain :

arc	bar	bar3d	circle	cleardevice	clearviewport	closegraph
detectgraph	drawpoly	ellipse	fillpoly	floodfill	getarccoords	getcolor
getbkcolor	getmaxy	getpixel	getx	gety	getaspectratio	getmaxx
initgraph	line	linerel	lineto	moverel	moveto	setcolor

rectangle setpallete textheight textidth sector putimage outtext

- **Unit Printer**

  Unit yang mengatur kerja printer. Harus menggunakan perintah uses printer untuk menggunakannya.

```
Program Contoh_Cetak ;
Uses Printer ;
Begin
   Writeln ( Lst, ‘Bahasa ‘ ) ;
   Writeln ( Lst, ‘Pascal ‘ ) ;
End.
```

- **Unit Windows**

  Merupakan suatu unit yang digunakan untuk menggantikan unit dos.

12.4. Mengkompilasi Unit

Unit dapat dikompilasi seperti halnya program biasa. Hasil dari kompilasi tersebut menghasilkan extention .TPU (Turbo Pascal Unit).

BAB XIII

REKURSI

Rekursi (recursion) adalah proses dari suatu sub program baik berupa fungsi atau prosedur yang memanggil dirinya sendiri.

Contoh program

```
Program Rekursi_dalam_pascal;
Procedure Rekursi;
Begin
     If A < 10 then
     Begin
          Write (‘Pascal’);
          A := A + 1;
          Rekursi;
     End;
End;
Var
     X : Byte;
Begin
     X := 3;
     Rekursi(X);
End.
```

Istilah Indefinite didalam Rekursi adalah proses rekursi yang terus dilakukan tanpa berhenti (rekursi yang tidak berujung).

Contoh program

```
Program Rekursi_Indefinite;
Procedure Rekursi;
Begin
     Write (‘Pascal ‘); Write;
     Rekursi;
End;
Begin
     Rekursi;
End.
```

Berikut ini adalah contoh program faktorial dengan menggunakan rekursi ;

```
Program faktorial;
Function factorial(A:integer):longint;
Begin
     If(A = 1) then
          factorial := 1;
```

```
      Else
            factorial := A * factorial(A-1);
End;

Var
      X : byte;
Begin
      Writeln('Factorial Sequence');
      Write('Berapa factorial : ');Readln(X);
      Writeln(X,' factorial ',' = ',factorial(X));
End.
```

BAB XIV

TIPE DATA SKALAR & SET (HIMPUNAN)

Skalar adalah tipe data yang didefinisikan oleh pemakai dengan menunjukan kumpulan dari nilai yang urutannya sudah pasti

14.1. Tipe Data Skalar

Tipe data skalar (Scalar Type) atau disebut juga tipe data terbilang (Enumerated Type) atau disebut juga tipe data skalar terdeklarasi (Declared Scalar Type) menunjukkan kumpulan dari nilai urutannya sudah pasti. Nilai dari tipe yang dideklarasikan ini akan diwakili dengan pengenal-pengenal yang kan menjadi suatu nilai konstanta.

14.2. Deklarasi tipe data skalar

Deklarasi dan penggunaan Tipe Data skalar

```
Type
Nama_tipe = (Pengenal_1, pengenal_2, … , pengenal_n);
```

Contoh Penulisan :

```
Type Materi = (Ppn, Pascal, Visual Basic, Visual Foxpro);
```

14.3. Penggunaan Tipe Data Skalar

Setelah tipe data skalr dideklarasikan di bagian deklarasi tipe, maka suatu variabel dapat dideklarasikan dengan tipe data skalar ini sebagai berikut :

```
Type
     Namahari = (Senin, Selasa, Rabu, Kamis, Jumat, Sabtu)
Var
     Hari : Namahari;
```

Variabel Hari telah dideklarasikan sebagai tipe yang dideklarasikan sendiri, yaitu bertipe Namahari. Namahari adalah tipe data skalar. Setelah variabel Hari dideklarasikan dengan tipe data skalar ini, selanjutnya dapat digunakan dalam program.

Contoh :

```
Type
     Namahari = (Senin, Selasa, Rabu, Kamis, Jumat, Sabtu)
Var
     Hari : Namahari;
Begin
     For Hari := Senin To Sabtu Do
     Writeln ('Pascal');
End.
```

14.4. Fungsi dan Prosedur Skalar

Fungsi Skalar

- Fungsi Standar Pred (Predecessor) adalah Fungsi standar untuk menghasilkan nilai sebelumnya (urutan nilai) dari suatu nilai ordinal.

  Contoh : `Pred(3) = 2`

- Fungsi Standar Succ (Succesor) adalah Fungsi standar untuk menghasilkan nilai berikutnya (urutan nilai) dari suatu nilai ordinal.

  Contoh : `Succ(4) = 5`

- Fungsi Standar Ord (Ordinal) adalah Fungsi standar untuk menghasilkan nilai interger yang merupakan urutan dari suatu tipe ordinal bersangkutan.

  Contoh program

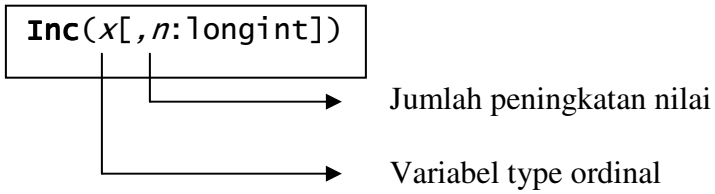
```
Program Skalar;
Uses crt;
Type
     Day = (senin, selasa, rabu, kamis, jumat, sabtu);
Var
     Hari : Day;
```

```
Begin
     Clrscr;
     For Hari := selasa to jumat do
     Write ('Pascal');
     Readln;
End.
```

Prosedur Skalar

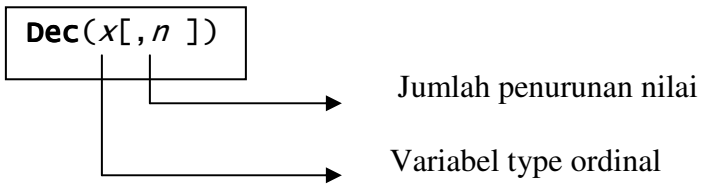
- Prosedur Standar INC (increment) digunakan untuk peningkatan (penambahan) nilai dari suatu nilai ordinal.

Bentuk Umum



- Prosedur Standar DEC (Decrement) digunakan untuk penurunan (pengurangan) nilai dari suatu nilai ordinal.

Bentuk Umum



14.5. Deklarasi Set (Himpunan)

Set (himpunan) adalah suatu kumpulan dari obyek yang mempunyai urutan yang dapat dianggap sebagai satu kesatuan.

Deklarasi set

```
Tipe Set Of  Tipe ordinal
```

14.6. Element Set

Yang menunjukan elemen-elemen didalam ungkapan set adalah Pembentuk set (set construktor) yang terdiri dari satu atau lebih elemen atau jangkauan dari elemen yang dipisahkan dengan koma dan diletakan diantara kurung bracket ("[" dan "]").

Contoh :

[2,3,5,7,11] nilai integer 2,3,5,7,11

[1..5] nilai integer 1 s/d 5

['0'..'9','a'..'f'] karakter '0' s/d '9' & karakter 'a' s/d 'f'

14.6. Operasi Set

- *Union (sum)* adalah operasi penjumlahan terhadap dua buah set dengan menggunakan operator set (+).

  Contoh :

```
A := [1,2,3]; B := [3,4];
C := A + B;
```

  Hasilnya : [1,2,3,4]

- *Set Difference* adalah operasi pengurangan dari suatu set terhadap set yang lain dengan menggunakan operator (-).

  Contoh :

```
A := [1,2,3]; B := [3,4];
C := A - B;
```

  Hasilnya : [1,2]

- *Intersection (product)* adalah operasi perkalian dua buah set dengan menggunakan operator set (*).

  Contoh :

```
A := [1,2,3]; B := [3,4];
C := A * B;
```

Hasilnya : [3]

Penyeleksian Set

- **Set Equality (kesamaan Set)**

  Bila seluruh anggota set pertama = anggota set ke dua. Lambangnya =.

  Contoh : `A := [1,2,3]; B := [3,4]; A = B`

- **Set Inequality (ketidaksamaan set)**

  Bila ada satu atau lebih anggota yang berbeda dari ke dua. Lambangnya <>.

  Contoh : `A := [1,2]; B := [1,2,3]; A <> B`

- **Set Inclusion (keterlibatan set)**

  Bila seluruh anggota dari salah satu set termasuk ke dalam set lainnya. Lambangnya >=, <=.

  Contoh :

  `A := [1,2]; B := [1,2,3]; A <= B, artinya A terlibat dlm B`

  `A := [1,2,3]; B := [1,2]; A >= B, artinya B terlibat dlm A`

- **Set membership (keanggotaan set)**

  Untuk mengecek karakter/angka, apakah termasuk dalam sebuah himpunan atau tidak. Lambangnya **in**.

  Contoh program

```
Program Union;
     Uses Crt;
Type
     Digit = set of 0..7;
Var
     A, B, C : Digit
```

```
     I : Byte;
Begin
     Clrscr;
     A := [0,2,4]; B := [1,3,5];
     C := A + B;
     Write ('Himpunan C : ');
     For I := 0 to 7 do
     If (I in C) then write (I:4);
     Readln;
End.
```

BAB XV

RECORD

Record adalah kumpulan type data terstruktur yang masing-masing data dapat mempunyai tipe data berbeda-beda.

15.1. Deklarasi Record

```
RECORD
      daftar_field_1  : type_1;
      daftar_field_2  : type_2;

      daftar_field_n  : type_n;
END;
```

15.2. Penggunaan Record

```
TYPE
      Recbarang = Record
                  Nama       : string;
                  Kualitas   : Char;
                  Harga      :  Longint
      End;
```

Contoh Program

```
Program Lingkaran;
Type
      Hasil = Record
            JariJari : real;
            Keliling : real;
            Luas     : real;
      End;
Var
      Lingkaran : Hasil;
Begin
      Write('Jari-jari lingkaran ? '); Readln(Lingkaran.JariJari);
      Lingkaran.Keliling := 2 * PI * Lingkaran.Jari-Jari;
      Lingkaran.Luas := PI * sqr(Lingkaran.JariJari);
      Writeln('Keliling lingkaran = ', Lingkaran.Keliling:7:2);
      Writeln('Luas lingkaran = ', Lingkaran.Luas:7:2);
      Readln;
End.
```

Statement With

Statement With didalam record berfungsi untuk mempermudah dan mempersingkat penggunaan field didalam suatu record sehingga pengenal record tidak selalu harus ditulis.

Contoh Program

```
Program Lingkaran;
Type
     Hasil = Record
          JariJari : real;
          Keliling : real;
          Luas     : real;
     End;
Var
     Lingkaran : Hasil;
Begin
     With Lingkaran Do
     Begin
          Write('Jari-jari lingkaran ? '); Readln(JariJari);
          Keliling := 2 * PI * Jari-Jari;
          Luas := PI * sqr(JariJari);
          Writeln('Keliling lingkaran = ',Keliling:7:2);
          Writeln('Luas lingkaran = ',Luas:7:2);
     End;
     Readln;
End.
```

BAB XVI

FILE

Sebuah file terdiri dari urutan komponen yang mempunyai tipe yang sama. Jumlah komponen dalam file sifatnya fleksibel, yaitu bisa ditambah dan dikurangi sewaktu-waktu.

16.1. Prosedur Standar File

- Assign adalah prosedur yang digunakan untuk menghubungkan nama dari external file kedalam suatu file variable dengan sintak :

```
Assign(f;nama;string);
```

- Rewrite adalah prosedur yang digunakan untuk membuka file yang baru atau yang belum pernah ada di disk dengan sintak :

```
Rewrite(f[ : file; recsize : word]);
```

- Reset adalah prosedure yang digunakan untuk membuka file yang telah ada dengan sintak :

```
Reset(f[ : file recsize : word]);
```

- Close adalah prosedure yang digunakan untuk menutup file yang telah dibuka dengan prosedure standar Rewrite, Reset atau Append (prosedur standar Append khusus untuk membuka file teks) dengan sintak :

```
Close(f);
```

- Erase adalah prosedur yang digunakan untuk menghapus suatu external file dengan sintak :

```
Erase(f);
```

- Rename adalah prosedur yang digunakan untuk mengganti nama dari suatu external file dengan sintak :

```
Rename(f; newname : string);
```

- Append adalah suatu prosedure yang digunakan untuk membuka file yang telah ada untuk keperluan menambah data ke dalam file dengan sintak :

```
Append(var f : text);
```

16.2. Fungsi Standar File

- EOLN adalah fungsi standar yang digunakan untuk mengetahui apakah posisi file berada di end-of-line market atau tidak dengan sintak :

```
Eoln[(var f : text)] : boolean;
```

- SEEKEOF adalah fungsi standar yang akan menghasilakan status akhir dari file dengan sintak :

```
SeekEof[(var f : text)] : boolean;
```

- SEEKEOLN adalah fungsi standar yang akan menghasilkan status akhir dari baris dengan sintak :

```
SeekEoln[(var f : text)] : boolean;
```

16.2. File Bertype

Prosedur Standar file bertype

- Write adalah prosedur standar yang digunakan untuk merekamkan satu atau lebih nilai kedalam file, dengan sintak :

```
Write(f, v1[, v2,…,vn]);
```

- Read adalah prosedur standar yang digunakan untuk membaca satu atau lebih nilai dari file, dengan sintak :

```
Read(f, v1[, v2,…,vn]);
```

- Seek adalah prosedur yang digunakan untuk mengarahkan penunjuk file ke suatu komponen tertentu didalam file, dengan sintak :

```
Seek(f;n : longint);
```

- Truncate adalah prosedur yang digunakan untuk menghapus sejumlah komponen atau record mulai dari posisi record tertentu dengan sintak :

```
Truncate(f);
```

Fungsi standar file bertype

- Filepos adalah fungsi standar yang digunakan untuk menghasilkan letak posisi sekarang dari penunjuk file dengan sintak :

```
FilePos(f) ; longint;
```

- Filesize adalah fungsi standar yang digunakan untuk menghasilkan jumlah dari komponen atau record yang sudah ada di file dengan sintak :

```
FileSize(f) : longint;
```

BAB XVII

VARIABEL DINAMIS

17.1. Pengertian Variabel Dinamis

Variabel dinamis adalah variabel yang dapat dibuat dan dialokasikan dengan prosedur standar New. Dengan prosedur standar New ini, maka variabel dinamis telah diletakkan di dalam heap dan posisi dari variabel dinamis yang berada di heap ditunjukkan oleh variabel pointer.

Variabel pointer menunjukkan alamat letak dari suatu variabel, dan variabel dinamis menunjukkan isi nilai variabelnya di alamat yang ditujukkan oleh variabel pointer.

17.2. Deklarasi Pointer.

Variabel dinamis tidak dapat dideklarasikan. Variabel dinamis hanya dapat ditunjukkan oleh variabel khusus yang berisi alamat memori yang digunaan oleh variabel dinamis tersebut. Variabel khusus ini disebut variabel pointer. Variabel pointer dapat dideklarasikan dengan tipe data pointer dengan simbol ^.

Contoh :

```
Type
     Catatankaryawan = Record
          Kode : String[5];
          Nama : String[25];
          Gaji : Real;
        End;
     Penunjukkaryawan = ^Catatankaryawan;
Var
     Datakaryawan : Penunjukkaryawan;
```

Pada contoh ini Datakaryawan adalah variabel pointer yang menunjuk paa letak record Catatankaryawan. Deklarasi ini dapat juga dituliskan sebagai berikut :

```
Type
     Catatankaryawan = Record
          Kode : String[5];
          Nama : String[25];
          Gaji : Real;
        End;
     Penunjukkaryawan = ^Catatankaryawan;
Var
     Datakaryawan : Penunjukkaryawan;
```

17.3. Penggunaan Pointer

Berikut ini adalah contoh penggunaan pointer dalam pembuatan program sederhana.

```
Type
     Tipestring = string[15];
     Penunjukstring = ^Tipestring;
Var
     Nama : Penunjukstring;
Begin
     Nama^ := 'Turbo Pascal';
     Writeln(Nama^);
End.
```

Daftar Pustaka

Ali, Sriyanto, dkk. 1994. Tuntunan Praktis Pemrograman Membedah PC Dengan Turbo Pascal. Elex Media Komputindo. Jakarta.

H. M., Jogiyanto. 1994. Turbo Pascal, Jilid 1. Andi Offset. Yogyakarta.

H. M., Jogiyanto. 1994. Turbo Pascal, Jilid 2. Andi Offset. Yogyakarta.

H. M., Jogiyanto. 1994. Pascal Tingkat Lanjutan. Andi Offset. Yogyakarta.

H. M., Jogiyanto. 1993. Teori Dan Aplikasi Pemrograman Komputer Bahasa Turbo Pascal. Andi Offset. Yogyakarta.

Hanafi, Mamduh M, Drs, MBA. 1995. Pemrograman Terstruktur Turbo Pascal. BPFE. Yogyakarta.

Martina, Inge. 1996. Turbo Pascal 5.5/6.0. Elex Media Komputindo. Jakarta.

Nugroho, Eko, Ir. 1992. Bahasa Pemrograman Pascal. Andi Offset. Yogyakarta.

Kata Pengantar

Bahasa Pascal adalah bahasa pemrograman tingkat tinggi yang banyak dipakai dalam bidang teknis dan sains. Bahasa Pascal mengutamakan pemrograman terstruktur sehingga program mudah dibuat dan mudah dilacak jika ada perbaikan. Produk bahasa Pascal yang banyak digunakan adalah Turbo Pascal.

Modul ini akan membahas pemakaian Turbo Pascal secara cepat dan mudah karena dibuat berdasarkan satuan acara perkuliahan yang terdapat di AMIK BSI, dan juga di dalam modul ini terdapat contoh-contoh program yang dapat langsung dipraktekkan di komputer.

Semoga modul Pascal ini berguna bagi pembaca atau siswa yang ingin mendalami bahasa pemrograman Pascal. Kritik dan saran yang membangun sangat diperlukan bagi sempurnanya modul Pascal ini.

Jakarta, Independence Day 2003

Tim Penyusun Modul