

**TOKI Handbook
(I)**

Ver 1.1

(c) 2004

Pada dasarnya, sebuah komputer tidak dapat mengerjakan apapun tanpa adanya perintah dari manusia. Perintah-perintah yang terstruktur dan sistematis untuk membuat komputer bekerja sesuai dengan apa yang diinginkan disebut **program**. Apa yang dapat dilakukan oleh program komputer? Komputer dapat diprogram untuk berbagai hal, misalnya diprogram untuk melakukan perhitungan suatu ekspresi matematika dan menampilkan hasilnya di layar monitor, diprogram untuk memainkan sebuah lagu, diprogram untuk mengurutkan data (misalnya mengurutkan data nama siswa, data nilai siswa), diprogram untuk permainan, diprogram untuk menggambar dan sebagainya. Program-program semacam itu dibuat oleh manusia, syarat utama dalam membuat program adalah perintah-perintah yang diberikan dalam program tersebut harus dimengerti oleh komputer.

Sayangnya, komputer hanya dapat mengerti sebuah bahasa yang disebut bahasa mesin, bahasa yang sangat berbeda dari bahasa manusia dan terlebih lagi akan amat menyulitkan untuk membuat sebuah program dalam bahasa mesin ini. Manusia menginginkan sebuah bahasa komputer yang sederhana yang dapat dimengerti dan mudah dipelajari oleh manusia sekaligus dapat dimengerti oleh komputer.

Bahasa komputer tersebut disebut bahasa pemrograman (*programming language*). Yang perlu diingat, konsep bahasa pemrograman adalah merubah / menerjemahkan perintah-perintah (program) yang diberikan oleh manusia ke dalam bahasa mesin yang dapat dimengerti oleh komputer. Jadi bahasa pemrograman adalah sarana interaksi antara manusia dan komputer. Seperti tujuan semula, bahasa pemrograman dibuat mudah dipelajari dan dimengerti agar manusia dapat mudah membuat program komputer dengan bahasa pemrograman ini (tak perlu menggunakan bahasa mesin untuk membuat program komputer).

Penerjemah bahasa pemrograman dibedakan menjadi tiga macam, yaitu:

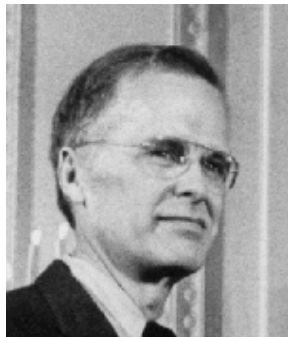
1. Assembler adalah program yang digunakan untuk menerjemahkan kode sumber dalam bahasa rakitan (assembly) ke dalam bahasa mesin
2. Kompiler adalah program penerjemah yang mengonversi semua kode sumber selain dalam bahasa rakitan menjadi kode objek. Hasil berupa kode objek inilah yang bisa dijalankan oleh komputer. Perlu diketahui, proses untuk melakukan penerjemahan ini biasa disebut kompilasi. Bahasa pemrograman yang menggunakan proses kompilasi adalah: Bahasa COBOL, Pascal, Bahasa C
3. Intepreter adalah program yang menerjemahkan satu per satu instruksi dalam kode sumber dan kemudian segera menjalankan instruksi yang telah diterjemahkan tersebut. Bahasa seperti BASIC pada awalnya menggunakan konsep intepreter ini.

Pada intinya, bahasa pemrograman digunakan untuk mempermudah manusia dalam berinteraksi dengan komputer. Syarat utama untuk membuat program komputer adalah dengan menggunakannya sesuai dengan kaidah-kaidah yang berlaku dalam bahasa pemrograman tersebut. Masing-masing bahasa pemrograman mempunyai ciri khas / kaidah tersendiri. Karena itu, sebelum membuat sebuah program dengan menggunakan bahasa pemrograman, sangat wajib untuk mengerti tentang aturan penulisan bahasa pemrograman tersebut.

Saat ini ada banyak bahasa pemrograman yang beredar di pasaran. Masing-masing memberikan kemudahan dan fasilitas untuk membuat sebuah program komputer yang sesuai dengan keinginan.

1. FORTRAN

FORTRAN kepanjangan dari Formula Translation. Pertama kali dikembangkan pada tahun 1956 oleh John Backus di IBM. Ditujukan untuk mempermudah pembuatan aplikasi matematika, ilmu pengetahuan dan teknik. Merupakan bahasa pemrograman tingkat tinggi pertama kali.



Gambar 1. John Backus, pencipta FORTRAN.

Keunggulan FORTRAN terletak pada dukungan untuk menangani perhitungan termasuk bilangan kompleks. Kelemahan bahasa ini terletak pada operasi masukan / keluaran yang sangat kaku. Selain itu kode sumbernya lebih sulit dipahami dibandingkan dengan bahasa pemrograman tingkat tinggi lainnya.

Contoh program dalam bahasa FORTRAN:

```
// JOB
// FOR
* ONE WORD INTEGERS
* IOCS (DISK, TYPEWRITER, KEYBOARD, PAPERTAPE)
  DIMENSION IEMG(10,15), IEMG1(13)
  DEFINE FILE 12(80,150,U,K)
  WRITE(1,10)
10 FORMAT('PAPERTAPE'// 'GIVE NUMBER EXPERIMENT (1-5 IN INT)')
  READ(6,30) M
30 FORMAT(I1)
  PAUSE 1
  DO 25 N=1,16
  DO 15 I=1,15
  READ(4,20) IEMG1
20 FORMAT(13I4)
  DO 15 J=4,13
  J3=J-3
15 IEMG(J3,I)=IEMG1(J)
  NE=N+(M-1)*16
25 WRITE(12'NE) IEMG
  CALL EXIT
  END
// DUP
*DELETE SJA1
*STORECI WS UA SJA1
*FILES(12,EMG)
```

2. COBOL

COBOL (Common Business Oriented Language) dikembangkan tahun 1959 dan tergolong sebagai bahasa tingkat tinggi. Sesuai dengan kepanjangan namanya, bahasa ini ditujukan untuk mempermudah pembuatan aplikasi di bidang bisnis. Sejauh ini bahasa ini masih banyak digunakan terutama di lingkungan komputer minikomputer dan mainframe.

Keunggulan COBOL adalah:

- Sintaksnya yang menggunakan kata-kata bahasa Inggris sehingga mempermudah programmer.
- Kemudahan terhadap penanganan file.
- Kemudahan terhadap masukan / keluaran program

Contoh program dalam bahasa COBOL:

```
000100 IDENTIFICATION DIVISION.  
000200 PROGRAM-ID. HELLOWORLD.  
000300  
000400*  
000500 ENVIRONMENT DIVISION.  
000600 CONFIGURATION SECTION.  
000700 SOURCE-COMPUTER. RM-COBOL.  
000800 OBJECT-COMPUTER. RM-COBOL.  
000900  
001000 DATA DIVISION.  
001100 FILE SECTION.  
001200  
100000 PROCEDURE DIVISION.  
100100  
100200 MAIN-LOGIC SECTION.  
100300 BEGIN.  
100400     DISPLAY " " LINE 1 POSITION 1 ERASE EOS.  
100500     DISPLAY "Hello world!" LINE 15 POSITION 10.  
100600     STOP RUN.  
100700 MAIN-LOGIC-EXIT.  
100800     EXIT.
```

3. BASIC

BASIC adalah kepanjangan dari *Beginner All-purpose Symbolic Instruction Code*. Dikembangkan tahun 1965 di Dartmouth College. Penciptanya adalah John Kemeny dan Thomas Kurtz. Awalnya BASIC digunakan sebagai pengajaran dasar untuk bahasa pemrograman sederhana.

Keunggulan BASIC terletak pada kemudahannya untuk dipakai dan penggunaan bahasa Inggris yang mirip dengan kehidupan sehari-hari sebagai sintaksnya. BASIC merupakan bahasa pemrograman yang sangat populer sebelum Pascal dibuat.

Contoh program dalam bahasa BASIC

```
REM Program mencari rata-rata 3 buah bilangan  
  
INPUT "Masukkan tiga buah bilangan : ", a, b, c  
rata=(a+b+c)/3  
PRINT "Rata-rata ketiga bilangan adalah : "; rata
```

4. PASCAL

Sejarah perkembangan Pascal dimulai pada tahun 1960, yaitu ketika bahasa pemrograman ALGOL 60 digunakan sebagai *algorithmic language* yang digunakan untuk memecahkan masalah sehari-hari dengan menggunakan komputer. Nama Pascal sendiri diambil dari nama seorang ahli matematika dan ilmu pengetahuan bangsa Perancis, yaitu Blaise Pascal (1623-1662). Niklaus Wirth dari Sekolah Teknik Tinggi Zurich - Swiss, menjadi terkenal sebagai perancang bahasa Pascal, *compiler* pertama yang dilaksanakan pada tahun 1970, kemudian direvisi pada tahun 1973 bersama K. Jensen. Kemudian pada tahun 1983 bahasa Pascal dapat dibakukan secara resmi dengan adanya Pascal Standard dari ISO.

Keunggulan bahasa Pascal adalah keteraturan dalam pembuatan program dan kelengkapan struktur data.

Contoh program dalam bahasa Pascal:

```
PROGRAM CariMin;

{Mencari Bilangan terkecil dari dua buah bilangan}

VAR
  x,y,min:integer;

BEGIN
  WRITE('Bilangan pertama : ');READLN(x);
  WRITE('Bilangan kedua : ');READLN(y);
  IF x>y THEN
    Min:=y
  ELSE
    Min:=x;
  WRITE('Bilangan terkecil : ',min);
END.
```

5. Bahasa C

Bahasa C diciptakan oleh Brian W. Kernighan dan Dennis M. Ritchie pada tahun 1972 di laboratorium Bell AT&T. Bahasa ini menggabungkan kemampuan pengendalian mesin dalam aras rendah dan struktur data serta struktur kontrol aras tinggi. Jadi dapat disebut bahasa C adalah bahasa pemrograman yang menggabungkan kemudahan pengontrolan hardware dalam bahasa pemrograman tingkat rendah serta struktur kontrol dalam bahasa tingkat tinggi. Bahasa C ini digunakan untuk menyusun sistem operasi UNIX dan Linux.

Keunggulan bahasa C adalah:

1. Sifat portabilitas, yaitu kode sumber pada sebuah platform dapat ditransfer ke platform lain tanpa ada perubahan
2. Kemudahan akses terhadap hardware
3. Cepat dan efisien

Pada tahun 1983, Bjarne Stroustrup mengembangkan bahasa C yang pada mulanya disebut sebagai "a better C". Namun kemudian bahasa ini dikenal dengan nama C++ (C Plus plus) yang mengunggulkan kelebihanannya sebagai bahasa pemrograman berorientasi objek.

Contoh program dalam bahasa C:

```
/*Mencari Bilangan terkecil dari dua buah bilangan*/

#include <stdio.h>

main ()
{
  int x,y, min;
  printf ("Bilangan pertama : ");
  scanf("%lf",&x);

  printf ("Bilangan kedua : ");
  scanf("%lf",&y);

  if x>y
    min=x
  else
    min=y;
  printf("Bilangan terkecil : %lf\n",min);
}
```

6. Bahasa Java

Bahasa Java dikembangkan oleh Sun Microsystem pada tahun 1995. Merupakan bahasa yang berorientasi objek. Kode Java dikompilasi dalam format yang disebut bytecode. Bytecode ini dapat dijalankan di semua komputer yang telah dilengkapi dengan program Java Interpreter dan Java Virtual Machine.

Java sangat populer karena pada masa awal Internet menjadi populer, Java telah menyediakan sarana untuk membuat program (yang disebut sebagai applet) yang dapat berjalan pada *web browser* seperti Internet Explorer, Netscape Navigator.

Contoh program dalam bahasa Java:

```
Public class SayHello {  
    Public static void main(String[] args {  
        System.out.println("Hello world!");  
    }  
}
```

1. Mengapa Pascal?

Mengapa Pascal yang dipilih sebagai bahasa pemrograman dalam TOKI? Dan mengapa compiler-nya adalah Free Pascal? Alasan di bawah ini setidaknya akan menjawab pertanyaan di atas,

- **Very Clean Language.** Bahasa Pascal adalah bahasa yang sangat mudah dibaca dan dikelola dibandingkan dengan bahasa C.
- **No Makefiles.** Tidak seperti kebanyakan bahasa pemrograman lain (Clipper, Java, C), Pascal tidak memerlukan makefiles. Ini dapat menghemat banyak waktu, compiler hanya melakukan proses kompilasi terhadap file-file memang perlu dikompilasi.
- **Pascal Compiler are FAST.** Begitu proses kompilasi dijalankan, maka hampir pada saat yang bersamaan program tersebut telah selesai dikompilasi, bahkan untuk program yang besar!
- **Each unit has it's own identifiers.** Pada Free Pascal, nama identifier (variabel, konstanta, tipe data, fungsi, prosedur) tidak harus berbeda untuk tiap unit. Walaupun dalam sebuah unit terdapat prosedur clrscr misalnya, maka dalam program yang menggunakan unit tersebut, nama clrscr masih dapat digunakan sebagai identifier. Hal ini berbeda dengan bahasa C yang mengharuskan identifier unik pada keseluruhan program.
- **Integrated Develoment Environment.** Free Pascal mempunyai IDE (editor) yang dapat bekerja dalam beberapa platform, di mana kita dapat langsung mengetikkan program, mengkompilasi serta melakukan debugging.
- **Great integration with Assembler.** Assembler adalah bahasa mesin yang dapat secara langsung mengakses hardware. Compiler Free Pascal mampu menggabungkan kemampuan dalam bahasa Assembler ini dengan bahasa Pascal.
- **Object Oriented Programming (OOP).** Bagi anda yang serius dengan pemrograman, tentu sangat tertarik dengan OOP ini (karena arah pemrograman di Windows adalah OOP). Free Pascal mensupport secara penuh OOP ini dalam bahasa Pascal.
- **Support Database.** Database yang disupport oleh Free Pascal adalah PostgreSQL, MySQL, Interbase, ODBC.
- **Smartlinking.** Free Pascal smart linker akan meninggalkan semua variabel-variabel yang tidak diperlukan sehingga program akan berukuran sangat kecil.
- **Compatible.** Kompabilitas Free Pascal terhadap compiler Pascal lain sangat tinggi. Hampir semua program di Turbo Pascal atau di Delphi dapat dikompilasi di Free Pascal.

2. Tentang Free Pascal

Free Pascal adalah compiler untuk bahasa Pascal. Free Pascal ini didistribusikan secara gratis di bawah lisensi GNU Public. Versi terakhir Free Pascal dapat di download di: <http://www.freepascal.org/>. Hingga bulan Juni 2004, versi Free Pascal yang paling stabil adalah versi 1.0.10. Sedangkan versi 1.9x masih dalam tahap pengembangan.

Free Pascal tersedia untuk berbagai macam prosesor, yaitu Intel x86, Motorola 680x0 (untuk versi 1.0 saja) dan PowerPC (mulai versi 1.9.2). Selain itu juga mendukung berbagai sistem operasi, yaitu: Linux, FreeBSD, NetBSD, MacOSX, DOS, Win32, OS/2, BeOS, SunOS (Solaris), QNX dan Classic Amiga.

3. Proses Instalasi Free Pascal

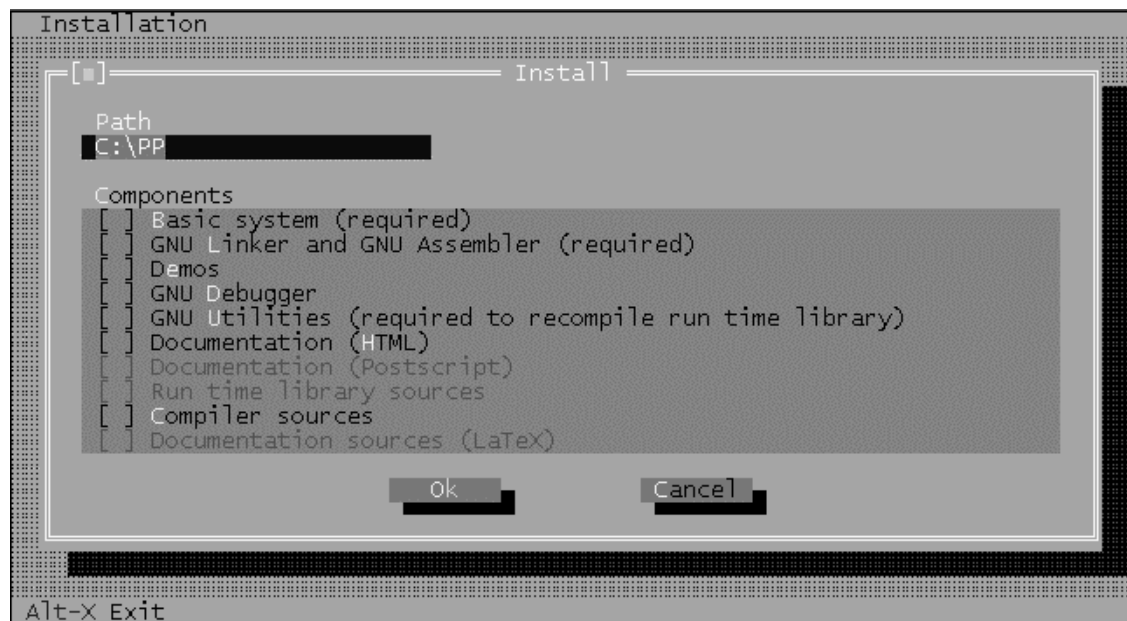
Sebelum melakukan instalasi Free Pascal, hardware yang dibutuhkan adalah:

- Prosesor Intel 80386 (atau yang lebih tinggi). Coprosesor tidak mutlak dibutuhkan, walaupun dapat memperlambat jalannya program jika menggunakan perhitungan floating point (perhitungan dengan angka di belakang desimal).
- Memori sedikitnya 4 Mb.
- Free Space di harddisk sedikitnya 3 Mb

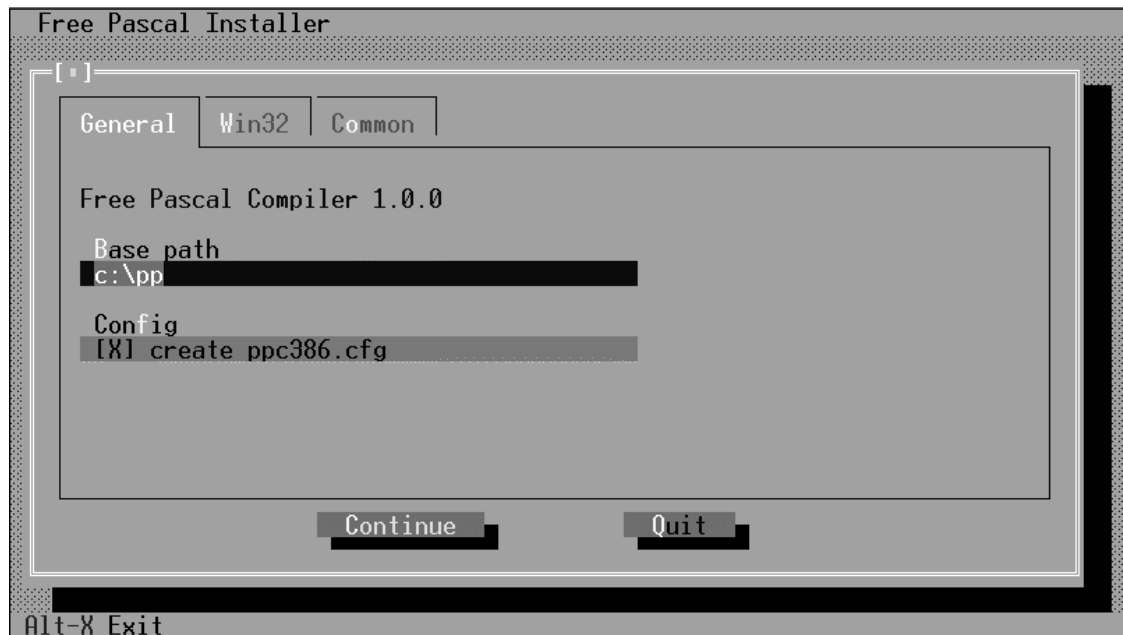
Untuk melakukan instalasi program Free Pascal, harus punya terlebih dulu versi terbaru dari FreePascal sesuai dengan platform untuk menjalankannya. Free Pascal didistribusikan dalam bentuk file Zip yang dapat dibuka/diekstrak dengan menggunakan WinZip atau program lain sejenis.

Berikut ini adalah langkah melakukan instalasi Free Pascal untuk DOS atau Windows.

- Cari dan jalankan file INSTALL.EXE di folder Free Pascal yang telah diekstrak tadi, lihat gambar 3.1 dan 3.2



Gambar 3.1 Tampilan awal proses instalasi



Gambar 3.2

- Program instalasi akan memberikan pilihan komponen-komponen yang hendak diinstal.
- Agar Free Pascal dapat dijalankan dari sembarang direktori dalam sistem, maka variabel path harus berisi: SET PATH=%PATH%;C:\PP\BIN\GO32V2 (pada DOS / ditambahkan di file AUTOEXEC.BAT). Atau SET PATH=%PATH%;C:\PP\BIN\WIN32 (untuk Windows) dan SET PATH=%PATH%;C:\PP\BIN\OS2 (untuk OS/2).

4. IDE (Integrated Developing Environment)

IDE menyediakan user interface yang nyaman untuk penulisan program, proses kompilasi serta debugging. IDE Free Pascal adalah aplikasi mode teks yang sama untuk setiap sistem operasi. Dimodelkan mirip dengan IDE di Turbo Pascal versi 7.0. Saat ini IDE Free Pascal ada untuk sistem operasi DOS, Windows, dan Linux. Untuk menjalankan IDE Free Pascal, jalankan file FP.EXE pada direktori C:\PP\BIN\GO32V2 (DOS) atau C:\PP\BIN\WIN32 (Windows).



Gambar 3.3. Tampilan awal IDE Free Pascal

Di atas sendiri disebut dengan *menu bar* dan di bawah adalah *status bar*. Area di antaranya disebut *desktop*.

Status bar menunjukkan shortcut keyboard yang akan sering digunakan dan memungkinkan dijalankan dengan mengkliknya. Untuk keluar dari IDE ini, dapat memilih menu File → Exit atau dengan shortcut Alt-X.

Catatan: Jika file FP.ANS ditemukan pada current directory (directory / folder tempat IDE dijalankan), maka tampilan desktop akan diambil dari file tersebut. File ini berisi perintah penggambaran untuk layar dalam standard ANSI.

5. Pengeditan pada IDE

Pengeditan teks pada IDE sama seperti pada editor lain.

- Insert Mode

Secara standard, mode pengeditan pada IDE Free Pascal adalah insert mode. Artinya semua teks yang diketikkan akan diletakkan di samping teks yang ada setelah posisi kursor.

Pada Overwrite mode, teks yang diketikkan akan menimpa teks yang ada.

Untuk mengubah dari insert mode ke overwrite mode atau sebaliknya, tekan Ins atau Ctrl-V.

- Blok

Teks dapat diblok dengan 3 cara.

1. Menggunakan mouse dengan melakukan drag pada teks yang hendak diblok
2. Menggunakan keyboard dengan menekan Ctrl-K B sebagai awal teks yang hendak diblok dan Ctrl-K K sebagai akhir dari blok.
3. Menggunakan keyboard dengan menahan tombol Shift, lalu mengarahkan kursor (dengan tombol panah) ke akhir dari blok.
4. Ctrl - K L → untuk memilih satu baris
5. Ctrl - K T → untuk memilih satu kata

Perintah-perintah yang berlaku saat ada pengeblokan adalah:

- Menggerakkan blok ke lokasi kursor (Ctrl - K V)
- Kopi blok ke lokasi kursor (Ctrl - K C)
- Hapus blok (Ctrl - K Y)
- Tulis blok ke file (Ctrl - K W)
- Membaca isi file (Ctrl - K R)
- Meng-indentasi (maju) blok (Ctrl - K I)
- Meng-unindentasi (mundur) blok (Ctrl - K U)
- Mencetak blok (Ctrl - K P)

- Bookmark (menandai)

Untuk menandai suatu posisi, Free Pascal dapat memberikan tanda hingga 9 bookmarks dalam sebuah file. Untuk menandai suatu posisi, dengan menggunakan Ctrl - K | [1-9], sedangkan untuk menuju ke tanda tersebut menggunakan Ctrl - Q | [1-9]

6. Syntax Highlight

IDE Free Pascal mampu memberikan Syntax highlight (warna yang berbeda untuk tiap-tiap sintaks). Syntax highlight ini dapat diatur di menu Options → Environment → Colors. Pada kotak dialog color, grup syntax harus dipilih terlebih dulu. Yang dapat diubah syntax highlight adalah:

- **Whitespace**, warna teks yang kosong / spasi antar kata.
- **Comments**, komentar dalam Free Pascal
- **Reserved Word**, kata-kata yang dikenal oleh Free Pascal
- **Strings**, ekspresi untuk string
- **Numbers**, angka dalam notasi desimal

- **Hex Number**, angka dalam notasi heksa desimal
- **Assembler**, blok untuk bahasa Assembler
- **Symbol**, simbol-simbol yang dikenali
- **Directive**, compiler directive
- **Tabs**, karakter tab

7. Code Completion

Code completion artinya editor akan memberikan kemungkinan perintah yang akan diketik untuk dilengkapi. Ini dilakukan dengan melakukan pengecekan teks yang diketikkan, lalu editor akan memberikan daftar kata kunci yang mungkin dimaksud oleh user. Tekan Enter untuk menerima daftar kata kunci tersebut.

Code completion dapat diatur di kotak dialog Code Completion lewat Options → Preferences → Codecompletion. Daftar kata kunci dapat dikelola di sini.



Gambar 3.4. Kotak dialog untuk mengelola kata kunci

8. Code Templates

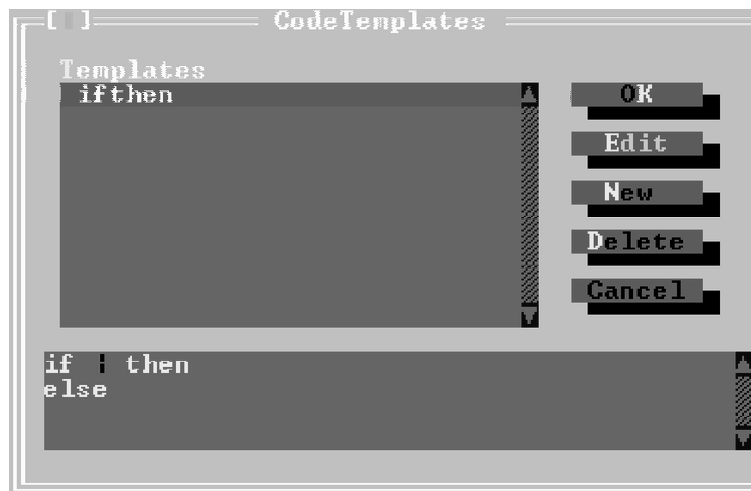
Code Templates adalah sebuah cara untuk memasukkan kode-kode perintah secara sekaligus. Setiap code templates diidentifikasi dengan nama yang unik. Nama untuk code templates dapat diasosiasikan dengan kode-kode perintah.

Misalnya nama `ifthen` dapat diasosiasikan dengan kode-kode perintah berikut:

```
IF | THEN
Begin
End
```

Code templates dapat dimasukkan dengan mengetikkan namanya, lalu menekan Ctrl-J ketika kursor berada di posisi nama template.

Code templates ini dapat diatur di menu options → preferences → Codetemplates.



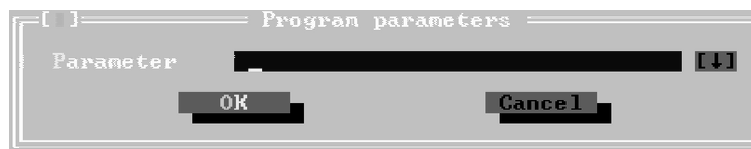
Gambar 3.5. Kotak dialog untuk mengatur codetemplates

9. Menjalankan Program

Program yang telah dikompilasi dapat dijalankan langsung lewat IDE, yaitu dengan :

- Pilih menu Run → Run atau
- Tekan Ctrl – F9

Bila ada parameter yang perlu diberikan untuk program, dapat ditambahkan pada Run → Parameter seperti pada gambar 3.6



Gambar 3.6. Kotak dialog untuk menambahkan parameter

Saat program di-run, akan terus berjalan hingga:

1. Program berhenti dengan normal
2. Terjadi error
3. Karena ada breakpoint dalam program atau
4. Program di reset oleh user.

Alternatif terakhir hanya mungkin terjadi apabila program dikompilasi dengan debug information

Untuk menjalankan program hingga baris tertentu, dapat dilakukan dengan:

- Pilih Run → Goto Cursor atau
- Tekan F4 pada posisi yang diinginkan

Ini juga dapat dilakukan hanya apabila program dikompilasi dengan debug information.

Program dapat dijalankan dengan mengeksekusi baris demi baris. Tekan F8 untuk menjalankan sebuah perintah. Penekanan F8 secara berturut-turut akan menjalankan program baris demi baris. Jika diinginkan agar eksekusi masuk dalam subrutin, maka dapat dilakukan dengan menekan F7.

Saat masuk dalam sebuah subrutin, pilih Run → Until return untuk menjalankan program hingga akhir subrutin tersebut.

Jika program memang harus dihentikan sebelum berhenti dengan normal, maka dapat melakukan:

- Pilih Run → Program reset atau
- Tekan Ctrl – F2

Program akan segera berhenti. Namun perlu diingat bahwa file-file yang terbuka (oleh perintah assign, reset) tidak secara otomatis tertutup.

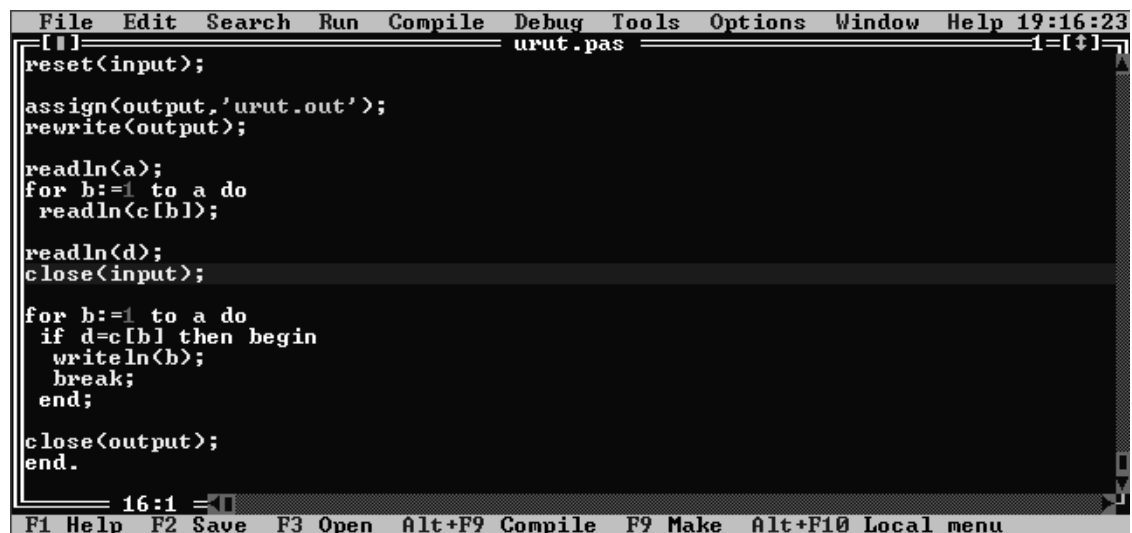
10. Mendebug Program

Dengan melakukan debugging program dapat dimungkinkan:

- Menjalankan program baris demi baris
- Menjalankan program hingga baris tertentu (breakpoint)
- Melihat isi dari variabel memori sementara program berjalan

Breakpoint akan mengakibatkan program berhenti pada baris di mana breakpoint diset / ditentukan. Pada saat ini, proses dikembalikan pada IDE dan dimungkinkan untuk dijalankan kembali.

Untuk mengeset / menentukan breakpoint, pada baris yang diinginkan pilih menu **Debug → Breakpoint** atau dengan menekan Ctrl – F8. Untuk menghapus breakpoint, dapat dengan menekan kembali Ctrl – F8.



```
File Edit Search Run Compile Debug Tools Options Window Help 19:16:23
urut.pas
reset(input);
assign(output,'urut.out');
rewrite(output);

readln(a);
for b:=1 to a do
  readln(c[b]);

readln(d);
close(input);

for b:=1 to a do
  if d=c[b] then begin
    writeln(b);
    break;
  end;

close(output);
end.
```

16:1

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

Gambar 3.7. Tampilan breakpoint

Fasilitas Watch adalah fasilitas untuk melihat isi variabel memori pada saat program dihentikan (oleh breakpoint). Watch ditampilkan di window baru dengan memilih menu Debug → Add Watch atau dengan menekan Ctrl – F7. Gambar 3.8 menunjukkan kotak dialog untuk watch.



Gambar 3.8. Kotak dialog watch.

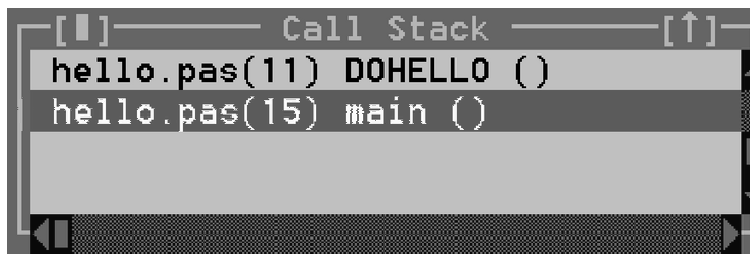
Setelah memasukkan nilai pada kotak dialog watch, tekan Enter. Akan muncul window baru, yaitu window watch (gambar 3.9).



Gambar 3.9. Window watch

Untuk berpindah antar window (dari watch ke window program atau sebaliknya, dapat menggunakan tombol F6 atau Shift-F6). Sedangkan untuk mengatur window, gunakan menu Window → Cascade.

Call stack adalah sebuah fasilitas di IDE untuk melihat eksekusi subrutin (termasuk parameternya) yang saat ini dijalankan program. Untuk menampilkan call stack lewat menu Debug → Call Stack atau dengan menekan Ctrl – F3.



Gambar 3.10. Window untuk call stack

Dengan menekan spasi pada window call stack, baris program akan diarahkan pada baris pemanggilan subrutin.

11. Free Pascal vs Turbo Pascal

Berikut adalah hal-hal yang diizinkan di Turbo Pascal namun tidak berlaku di Free Pascal, yaitu:

1. Duplikasi label untuk case tidak diizinkan. Ini merupakan bug pada Turbo Pascal, dan Free Pascal masih tetap mempertahankannya.
2. Parameter list yang dideklarasikan pada function atau procedure sebelumnya, harus benar-benar persis untuk function atau procedure yang sesungguhnya.
3. Variabel Mem, MemW, MemL dan Port tidak berlaku lagi di Free Pascal.
4. PROTECTED, PUBLIC, PUBLISHED, TRY, FINALLY, EXCEPT, RAISE merupakan kata-kata tercadang (*reserved word*)
5. Kata tercadang FAR dan NEAR tidak lagi berlaku lagi, karena Free Pascal adalah compiler 32 bit
6. INTERRUPT hanya berlaku untuk mode DOS
7. File yang dibuka dengan perintah rewrite hanya dapat ditulisi saja, tidak dapat dibaca. Untuk dapat membaca, maka harus diberikan perintah reset.

Berikut adalah tambahan Free Pascal dibandingkan Turbo Pascal:

1. Ada lebih banyak reserved word. Lihat session IV – nomer 5
2. Function dapat mengembalikan tipe kompleks seperti record dan array
3. Function dapat dihandle dalam function itu sendiri, contoh:

```
function a : longint;
```

```

begin
  a:=12;
  while a>4 do
    begin
      {...}
    end;
  end;
end;

```

Contoh di atas dapat dijalankan dalam Turbo Pascal, namun compiler mengasumsikan bahwa perintah while a>4 do adalah pemanggilan fungsi secara rekursif (pada Turbo Pascal). Untuk Free Pascal, pemanggilan rekursif pada kasus di atas adalah dengan menambahkan () (kurung buka tutup) pada a.

```

function a : longint;
begin
  a:=12;
  { pemanggilan secara rekursif }
  if a()>4 then
    begin
      {...}
    end;
end;

```

4. Pemanggilan EXIT dapat dengan memberikan nilai kembali

```

function a : longint;

begin
  a:=12;
  if a>4 then
    begin
      exit(a*67); {function menghasilkan a*67 }
    end;
end;

```

5. Free Pascal mendukung adanya fungsi yang *overloading*. Artinya kita bisa mendefinisikan nama fungsi yang sama dengan parameter yang berbeda, contoh:

```

procedure DoSomething (a : longint);
begin
  {...}
end;

procedure DoSomething (a : real);
begin
  {...}
end;

```

6. Mendukung nama file panjang. Mulai Windows 95, nama file dapat sepanjang 255 karakter. Free Pascal dapat mendukung penyimpanan dengan nama file panjang.

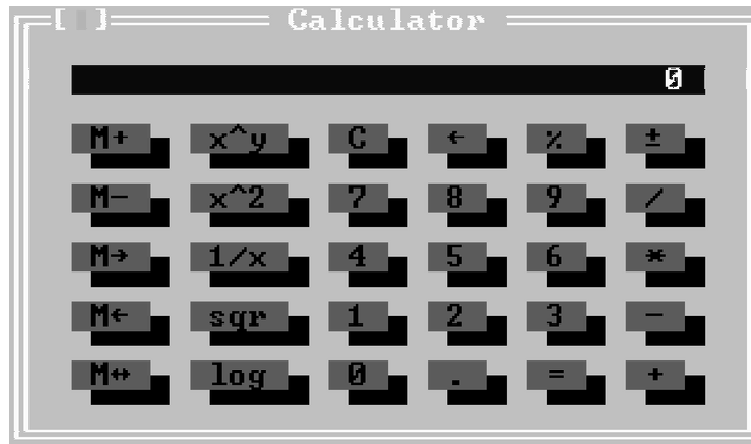
12.Fasilitas Tambahan

Tabel ASCII dapat ditampilkan langsung lewat IDE, yaitu pada menu Tools → Ascii Table. Untuk memasukkan karakter ASCII pada editor, dapat langsung melakukan double click atau tekan Enter.



Gambar 3.11. Tabel ASCII di Free Pascal

Calculator juga tersedia lewat menu Tools → Calculator. Fasilitas calculator pada free pascal tidak mensupport tanda kurung. Hasil perhitungan dari kalkulator ini dapat langsung diletakkan pada editor dengan menekan Ctrl – Enter.



Gambar 3.12. Tampilan fasilitas calculator

Calculator ini mendukung operasi standard dalam matematika, seperti pada tabel 3.1

Tabel 3.1. Operasi pada fasilitas calculator

Operasi	Tombol di calculator	Penekanan keyboard
Penambahan dua bilangan	+	+
Pengurangan dua bilangan	-	-
Perkalian dua bilangan	*	*
Pembagian dua bilangan	/	/
Menghapus bilangan terakhir	←	Backspace
Menghapus bilangan	C	C
Merubah tanda	±	
Kalkulasi persen	%	%
Hasil	=	Enter
Pemangkatan	X ^y	
Inverse	1/x	
Akar pangkat dua	Square root	
Logaritma natural	Log	
Pangkat dua	X ²	

13. Daftar Shortcut pada IDE

Shortcut akan berguna untuk mempercepat pengeditan pada IDE. Berikut adalah daftar shortcut yang dikenal IDE Free Pascal.

- Shortcut untuk window IDE dan Help

Perintah	Shortcut	Alternatif
Help	F1	
Topik help terakhir	Alt - F1	
Mencari kata pada help	Ctrl - F1	
Indeks help	Shift - F1	
Menutup window aktif	Alt - F3	
Zoom / Unzoom window	F5	
Menggerakkan window	Ctrl - F5	
Berpindah ke window berikutnya	F6	
Berpindah ke window sebelumnya	Shift - F6	
Menu	F10	
Menu lokal	Alt - F10	
Daftar Window	Alt - 0	
Mengaktifkan window	Alt - [no. Urut window]	
Keluar dari IDE	Alt - X	

- Shortcut untuk me-RUN dan debugging

Perintah	Shortcut	Alternatif
Merest program	Ctrl - F2	
Menampilkan Call stack	Ctrl - F3	
Jalankan sampai posisi kursor	Ctrl - F4	
Melihat tampilan layar	Alt - F5	
Trace / penelusuran program	F7	
Menambah Watch	Ctrl - F7	
Step over	F8	
Make	F9	
Menjalankan program	Ctrl - F9	
Mengkompilasi program aktif	Alt - F9	
Message	F11	
Pesan Compiler	F12	

- Shortcut untuk navigasi kursor

Perintah	Shortcut	Alternatif
Ke kiri	Panah kiri	Ctrl - S
Ke kanan	Panah kanan	Ctrl - D
Ke atas	Panah atas	Ctrl - E
Ke bawah	Panah bawah	Ctrl - X
Satu huruf ke kiri	Ctrl - panah kiri	Ctrl - A
Satu huruf ke kanan	Ctrl - panah kanan	Ctrl - F
Scroll satu baris ke atas	Ctrl - W	
Scroll satu baris ke bawah	Ctrl - Z	
Satu halaman ke atas	Page Up	Ctrl - R

Satu halaman ke bawah	Page Down	
Awal baris	Home	Ctrl - Q S
Akhir baris	End	Ctrl - Q D
Awal baris program	Ctrl - Home	Ctrl - Q E
Akhir baris program	Ctrl - End	Ctrl - Q X
Posisi kursor terakhir	Ctrl - Q P	

- Shortcut untuk pengeditan

Perintah	Shortcut	Alternatif
Hapus karakter	Del	Ctrl - G
Hapus di sebelah kiri kursor	Backspace	Ctrl - H
Hapus baris	Ctrl - Y	
Hapus sampai baris terakhir	Ctrl - Q Y	
Hapus kata	Ctrl - T	
Menyisipkan baris	Ctrl - N	
Mode Insert / overwrite	Ins	Ctrl - V

- Shortcut untuk blok

Perintah	Shortcut	Alternatif
Ke awal blok	Ctrl - Q B	
Ke akhir blok	Ctrl - Q K	
Blok baris	Ctrl - K L	
Cetak blok	Ctrl - K P	
Pilih kata	Ctrl - K T	
Hapus teks yang diblok	Ctrl - Del	Ctrl - K Y
Kopi blok	Ctrl - K C	
Pindah blok	Ctrl - K V	
Kopi blok ke clipboard	Ctrl - Ins	
Pindah blok ke clipboard	Shift - Del	
Indentasi blok satu kolom	Ctrl - K I	
Un-indentasi blok satu kolom	Ctrl - K U	
Menyisipkan teks dari clipboard	Shift - Ins	
Tulis blok ke file	Ctrl - K W	
Menjadikan blok huruf besar	Ctrl - K N	

- Shortcut untuk merubah blok

Perintah	Shortcut	Alternatif
Mengawali blok	Ctrl - K B	
Mengakhiri blok	Ctrl - K K	
Menghilangkan blok	Ctrl - K Y	
Menambah blok satu karakter ke kiri	Shift - Panah kiri	
Menambah blok satu karakter ke kanan	Shift - Panah kanan	
Menambah blok ke akhir baris	Shift - End	
Menambah blok ke baris di atasnya	Shift - Panah atas	
Menambah blok ke baris di bawahnya	Shift - Panah bawah	
Menambah blok satu kata di kiri	Ctrl - Shift - Panah kiri	
Menambah blok satu kata di kanan	Ctrl - Shift - Panah kanan	
Menambah blok satu halaman ke atas	Shift - Page Up	
Menambah blok satu halaman ke bawah	Shift - Page Down	
Menambah blok hingga ke awal program	Ctrl - Shift - Home	
Menambah blok hingga ke akhir program	Ctrl - Shift - End	

- Shortcut lain-lain

Perintah	Shortcut	Alternatif
Simpan file	F2	Ctrl - K S
Buka file	F3	
Cari	Ctrl - Q F	
Cari lagi	Ctrl - L	
Cari dan ganti	Ctrl - Q A	
Menandai	Ctrl - K [0-9]	
Menuju ke tanda	Ctrl - Q [0-9]	
Undo	Alt - Backspace	

Sebelum membahas dengan detail tentang bahasa Pascal (dalam hal ini adalah compiler Free Pascal), perlu diketahui terlebih dulu struktur program dalam bahasa Pascal. Bentuk ini harus dipenuhi agar program dapat dikompilasi oleh compiler.

1. Struktur Bahasa Pascal

Bentuk umum bahasa Pascal adalah sebagai berikut:

```
PROGRAM      nama(file1,file2,file3);
CONST       deklarasionstanta;
VAR         deklarasivariabel;
TYPE        deklarasi tipe;
LABEL       deklarasilabel;
FUNCTION    deklarasi fungsi;
PROCEDURE   deklarasi prosedur;

BEGIN
  statement1;
  statement2;
  statement3;
  ...
END.
```

Program Pascal terdiri dari 3 bagian pokok, yaitu:

1. Nama Program

Nama program adalah hanya sekedar menuliskan judul dari program, tidak mempunyai arti apa-apa dalam proses kompilasi. Judul program dapat diikuti oleh file-file data yang berhubungan dengan program tersebut.

Pada Turbo Pascal, dapat ditambahkan klausa uses yang menunjukkan bahwa program menggunakan unit.

2. Deklarasi

Bagian ini berisi deklarasi pengenalan maupun data yang dipergunakan di dalam program. Walaupun tampaknya membuang-buang waktu dan tidak berguna, namun sesungguhnya merupakan bagian terpenting dari rangka penyusunan sebuah program yang terstruktur. Struktur program sangat penting dalam pembuatan program yang panjang, karena bagian ini akan mengingatkan programmer tentang variabel, tipe data, konstanta, fungsi, prosedur yang digunakan dalam program. Selain itu, orang lain yang membaca program akan lebih dapat mengerti jalannya program dengan deklarasi ini.

Bagian deklarasi ada 6 macam, yaitu:

- Deklarasi CONST

Deklarasi ini gunanya untuk mendeklarasikan nama konstanta tertentu. Nama konstanta adalah merupakan suatu pengenalan (identifier) yang nilainya tidak dapat berubah dalam program.

Contoh:

```
CONST
  Pi=3.14
  Titikkoma='';
```

- Deklarasi VAR

Deklarasi ini gunanya adalah untuk menyatakan variabel yang digunakan dalam program. Variabel adalah suatu pengenalan (identifier) yang nilainya dapat berubah.

Contoh:

```
VAR
  Data:array[1..100] of byte;
  Umur: 0..100;
```

- **Deklarasi TYPE**

Deklarasi type dipergunakan untuk menyusun suatu bentuk tipe data yang baru sebagai hasil penggabungan dari tipe-tipe yang sudah ada.

Contoh:

```
TYPE
  Data=array[1..100] of byte;
  Hari=(Senin, Selasa, Rabu, Kamis, Jumat, Sabtu);
VAR
  Nilai:Data;
  HariKerja:Senin..Jumat;
```

- **Deklarasi LABEL**

Deklarasi label menjelaskan adanya label atau tujuan yang bisa melompatkan jalannya program dengan statement goto.

- **Deklarasi FUNCTION**

Function adalah bagian dari program yang melakukan tugas tertentu dan menghasilkan suatu nilai.

Sintaks penulisan:

```
function namafunction:tipehasil;
    Atau
function namafunction(daftarparameter):tipehasil;
```

Contoh:

```
function UpCaseStr(S: string): string;
var
  I: Integer;
begin
  for I := 1 to Length(S) do
    if (S[I] >= 'a') and (S[I] <= 'z') then
      Dec(S[I], 32);
  UpCaseStr := S;
end;
begin
  writeln(UpCaseStr('this is text'));
end.
```

- **Deklarasi PROCEDURE**

Procedure adalah bagian dari program yang melakukan aksi tertentu, seringkali aksi tersebut dilakukan berdasarkan parameter.

Sintaks penulisan:

```
procedure namafunction;
    Atau
procedure namafunction(daftarparameter);
```

Contoh:

```
procedure WrStr(X, Y: integer; S: string);
var
  SaveX, SaveY: Integer;
begin
  SaveX := WhereX;
  SaveY := WhereY;
  GotoXY(X, Y);
  Write(S);
  GotoXY(SaveX, SaveY);
end;

begin
  WrStr(10,20,'This is text');
```

end.

3. Aturan penulisan program Pascal

Suatu bahasa pemrograman selalu mempunyai aturan penulisan program. Hal ini menunjukkan konsistensi kompiler dalam melakukan proses kompilasi. Aturan pada program Pascal adalah sebagai berikut:

- Program pascal dapat ditulis pada kolom berapa saja dan diakhiri pada kolom berapa saja.
- Antar statement / perintah dipisahkan dengan tanda ; (titik koma)
- Akhir dari sebuah program Pascal ditandai dengan tanda . (titik) setelah perintah **END**. Semua statement / perintah setelah **END**. tidak akan dianggap sebagai perintah.
- Spasi antar pengenal (identifier) diabaikan.
- Baris komentar diletakkan di antara tanda (* dan *) atau { dan }. Baris komentar tidak akan dieksekusi oleh komputer. Baris komentar biasanya dipergunakan untuk memberikan penjelasan-penjelasan guna memperjelas pengertian variabel atau tipe atau perintah dalam sebuah program.

4. Simbol (Symbols)

Pascal mengenal simbol-simbol yang dapat digunakan dalam program, yaitu:

1. huruf : A..Z , a..z
2. digit : 0..9
3. digit heksadesimal : 0..9, A..F, a..f
4. Karakter khusus : + - * / = < > [] . , () : ^ @ { } \$ #
5. Pasangan karakter : <= >= := (* *)

Selain dari karakter-karakter di atas, adalah karakter yang tidak dikenali oleh bahasa Pascal.

5. Kata tercadang (Reserved Words)

Kata tercadang adalah bagian dari bahasa Pascal dan tidak dapat dipakai untuk kegunaan lain dalam program (tidak dapat didefinisikan ulang). Kata tercadang ini tidaklah case sensitive, artinya for akan sama dengan FOR.

Berikut adalah perbandingan kata tercadang dalam Turbo Pascal dan free pascal:

	Free Pascal	Turbo Pascal
A	absolute, and, array, asm	absolute, and, array, asm
B	begin, break	begin, break
C	case, const, constructor, continue	case, const, constructor, continue
D	destructor, div, do, downto, dispose	destructor, div, do, downto
E	else, end, exit	else, end
F	file, for, function, false	file, for, function
G	goto	Goto
I	if, implementation, in, inherited, inline, interface	if, implementation, in, inherited, inline, interface
L	label	label
M	mod	mod
N	nil, not, new	nil, not
O	object, of, on, operator, or	object, of, on, operator, or
P	packed, procedure, program	packed, procedure, program
R	record, repeat	record, repeat
S	self, set, shl, shr, string	self, set, shl, shr, string

T	then, to, type, true	then, to, type
U	unit, until, uses	unit, until, uses
V	var, while, with	var, while, with
X	xor	xor

6. Pengenal (*Identifier*)

Identifiers are names you give to any part of a Turbo Pascal (TP) program, including variables, types, procedures, functions, and so on. TP requires that you declare identifiers before using them in your code (program).

Syarat-syarat penamaan sebuah identifier adalah:

- Dapat sepanjang apapun, namun Turbo Pascal akan mengambil 63 karakter pertama dari nama identifier
- HARUS diawali dengan huruf atau underscore (_)
- Karakter ke dua dan selanjutnya dapat berupa huruf, angka, atau underscore
- Tidak boleh ada 2 identifier yang sama dalam satu program
- Tidak boleh berupa *reserved word*. *Reserved Word* adalah kata yang telah dikenal oleh Pascal yang telah mempunyai kegunaan tertentu.

Contoh penulisan identifier yang benar:

- cobal
- jari_jari
- programcoba_coba
- integer (mengapa identifier ini diperbolehkan?)

Contoh penulisan identifier yang salah:

- coba 1 (mengandung spasi)
- jari-jari (mengandung karakter -)
- 2b (diawali dengan angka)
- to (mengapa tidak diperbolehkan?)

Contoh penggunaan identifier:

```
program cobal;
```

```
var
  a,b:byte;
```

```
type
  c=word;
begin
end.
```

Pada program di atas, terdapat 4 buah identifier, yaitu cobal, a, b dan c. Cobal digunakan sebagai identifier nama program, a dan b digunakan sebagai identifier dari deklarasi var dan c sebagai identifier nama type.

7. Angka (*Numbers*)

Ada 2 macam penulisan angka dalam bilangan bulat yang dikenal dalam Pascal.

- Normal, dalam format desimal (basis 10).
- Heksadesimal (basis 16), penulisan angka dalam format heksadesimal harus didahului oleh tanda \$. Misalnya \$FF berarti 255 desimal.

Free Pascal menambahkan 2 format penulisan lagi, yaitu:

3. Oktal (basis 8), penulisan angka dalam format oktal harus didahului oleh tanda &. Misalnya 15 desimal mempunyai nilai yang sama dengan &17.
4. Biner (basis 2), penulisan angka dalam format biner harus didahului oleh tanda %. Misalnya 15 desimal mempunyai nilai yang sama dengan %1111.

8. Tipe (Type)

There are several ways to categorize data types:

- a. *Some types are predefined (or built-in); the compiler recognizes these automatically, without the need for a declaration. Other types are created by declaration (user-defined types)*
- b. *Types can be classified as simple, string, structured, or pointer.*

Berikut ini adalah taksonomi dari tipe data yang ada di Turbo Pascal

- Sederhana
 - Ordinal
 - Integer / bilangan bulat (dibagi lagi menjadi : Byte, Shortint, Word, Integer, Longint)
 - Character
 - Boolean
 - Enumerated
 - Subrange
 - Real
- String
- Terstruktur
 - Set
 - Array
 - Record
 - File
- Pointer

Free Pascal melakukan beberapa perubahan untuk taksonomi tipe data ini, yaitu:

- Tipe dasar / tipe sederhana (*Based Types*)
 - Tipe ordinal (*ordinal types*)
 - Tipe real (*real types*)
- Tipe karakter (*Character Types*)
 - Karakter (*Char*)
 - Untai (*Strings*)
 - Untai pendek (*Short strings*)
 - Ansistrings
 - Konstanta string (*Constant string*)
 - Pchar – Null terminated string
- Tipe terstruktur (*Structured types*)
 - Array
 - Record
 - Set
 - File
- Pointer
- Deklarasi Forward
- Tipe Prosedur

Diagram Tipe data di Turbo Pascal

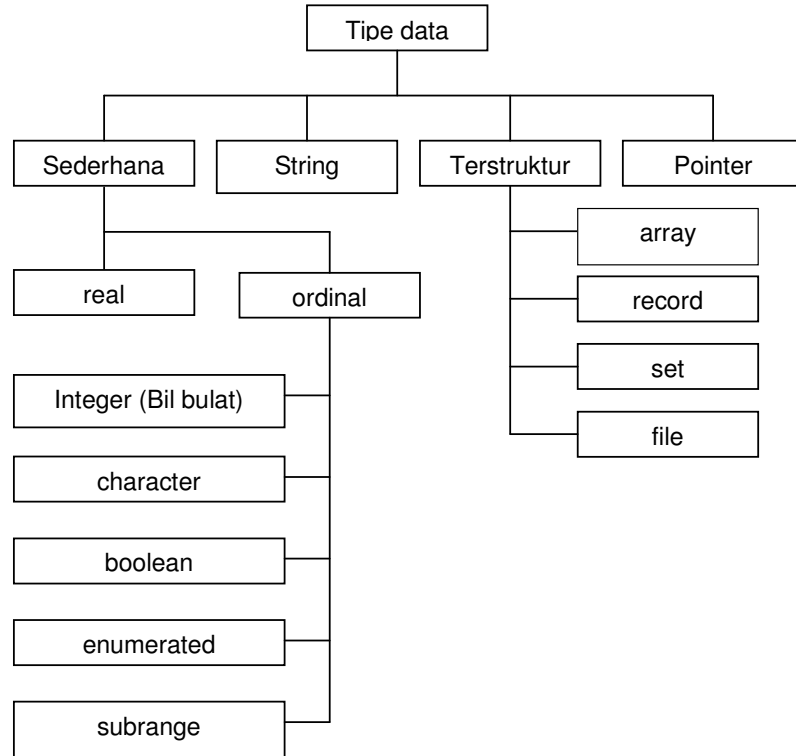
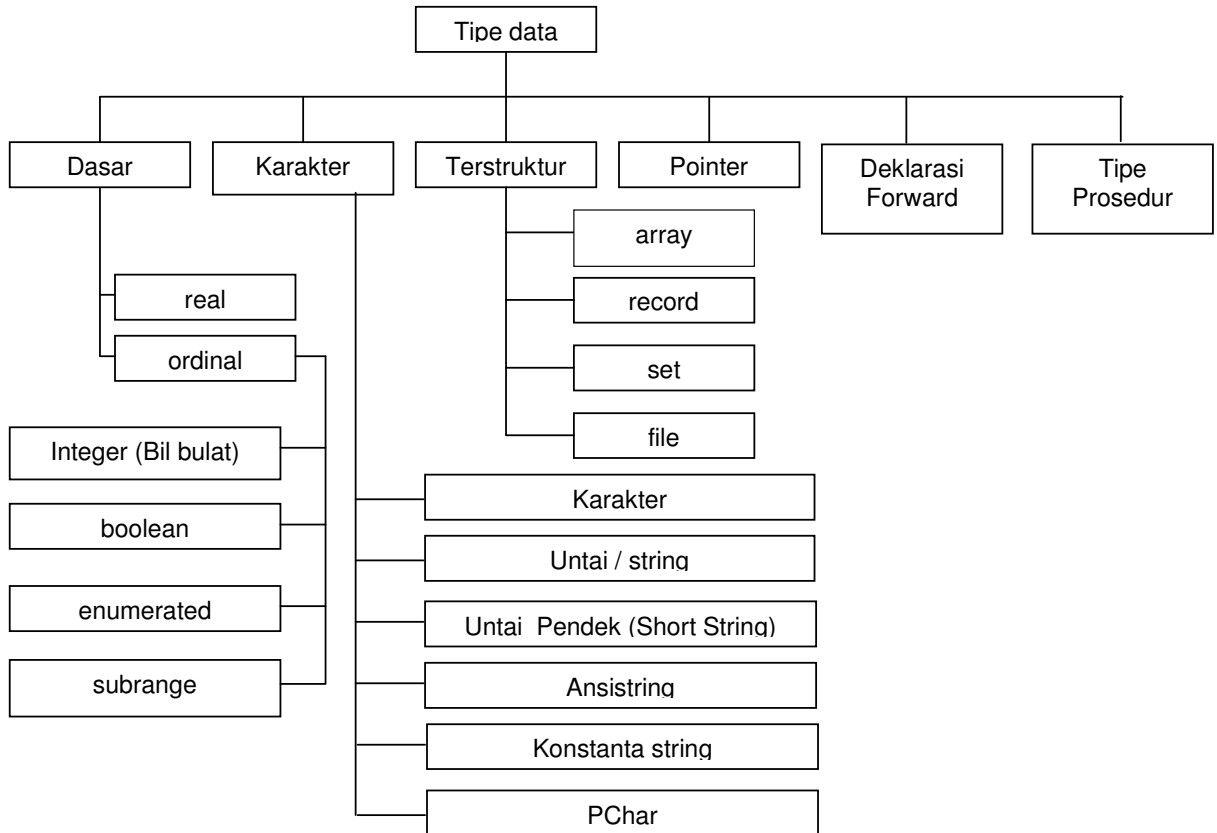


Diagram Tipe data di Free Pascal



Penjelasan Tipe data pada Free Pascal

A. Tipe Dasar / Sederhana (*Based Types or Simple Types*)



Tipe dasar mendefinisikan suatu nilai yang berurutan. Ada 2 kelas dari tipe sederhana ini yaitu:

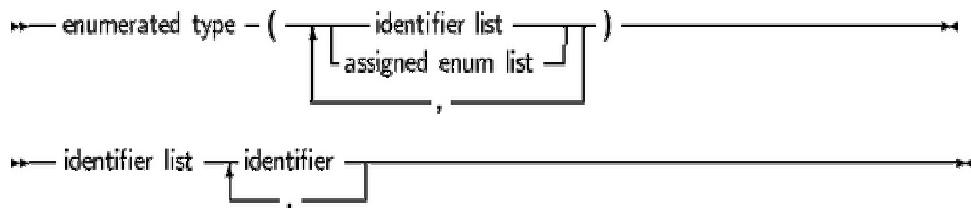
- **Tipe Ordinal (*Ordinal type*)**

Pada tipe ordinal, Free Pascal telah mendefinisikan 14 *predefined type* (apa yang dimaksud *predefined type*?), yaitu : bilangan bulat, boolean, char. (dimana 11 buah tipe data yang lain?). Artinya pembuat program tidak perlu mendefinisikan tipe untuk tipe-tipe data di atas. Sedangkan untuk *user defined type* pada tipe ordinal ada 2 macam, yaitu : tipe enumerasi (*enumerated types*) dan tipe subjangkauan (*subrange types*).

Tipe Ordinal dapat dioperasikan dengan fungsi / prosedur berikut:

- ORD (fungsi)
- PRED (fungsi)
- SUCC (fungsi)
- HIGH (fungsi)
- LOW (fungsi)
- INC (prosedur)
- DEC (prosedur)

Tipe Enumerasi (*Enumerated*)



Tipe Enumerasi mendefinisikan nilai yang berurutan ke **suatu elemen** dalam suatu daftar identifier. Elemen pertama mempunyai nilai 0, yang kedua mempunyai nilai 1 dan seterusnya.

Format penulisan:

```
type
  identifier=(identifier_1,identifier_2,...,identifier_n)
```

(Berdasarkan definisi tipe enumerasi, mana yang disebut elemen pada format penulisan di atas? Mana yang disebut nilai yang berurutan? Mana yang disebut daftar identifier? Dan di mana letak user defined type dari tipe enumerasi ini)

Contoh tipe enumerasi yang benar:

```
type
  kartu=(club, heart, diamond, spade);
  musik=(jazz, blues, country, pop);
```

Untuk mengetahui urutan suatu nilai dari tipe enumerasi ini dengan menggunakan fungsi `ord`, untuk mengetahui nilai sebelumnya menggunakan fungsi `pred` dan untuk mengetahui nilai sesudahnya digunakan fungsi `succ`.

Contoh penggunaan tipe enumerasi dalam program:

```
var
  model:(small,medium,large);
begin
  model:=medium;
  writeln(ord(model)); {berapa urutan dari model?}
  model:=pred(model); {apa nilai dari model?}
  model:=succ(succ(model)); {apa nilai dari model?}
end.
```

Pada Free Pascal tipe enumerasi dapat dibuat dengan model bahasa C, yaitu:

```
Type
  EnumType = (satu, dua, tiga, empatpuluh:= 40,empatsatu);
```

Hasilnya, ordinal dari forty adalah 40, bukan 3 seperti pada tipe enumerasi biasa dan ordinal dari fortyone adalah 41. Namun harap diperhatikan bahwa fungsi `Pred` dan `Succ` tidak dapat digunakan untuk tipe enumerasi seperti di atas.

Compiler akan menganggap salah untuk tipe enumerasi berikut:

```
Type
  EnumType = (satu,dua, tiga, empatpuluh := 40,tigapuluh=30);
```

Contoh penggunaan tipe enumerasi yang benar:

```
1. var MyCard: (Club, Diamond, Heart, Spade);
2. var Card1, Card2: (Club, Diamond, Heart, Spade);
3. type Suit = (Club, Diamond, Heart, Spade);
   var
     Card1: Suit;
     Card2: Suit;
```

Contoh penggunaan tipe enumerasi yang salah:

```
1.
   var Card1: (Club, Diamond, Heart, Spade);
   var Card2: (Club, Diamond, Heart, Spade);
```

(dimana letak kesalahannya?)

Tipe Subjangkauan (*Subrange*)

↳ — **subrange type** — constant — .. — constant — ↳

Tipe subjangkauan mendefinisikan suatu elemen dari nilai terkecil sampai nilai terbesar.

Format penulisan

```
type
  identifier=nilai1..nilai2;
```

`nilai1` dan `nilai2` haruslah merupakan merupakan nilai dari **tipe ordinal** yang sama. Pada suatu tipe subjangkauan, juga dapat digunakan fungsi-fungsi `ord`, `pred` dan `succ`. Fungsi-fungsi tersebut masih tetap mengacu pada tipe enumerasi asalnya.

Contoh dalam program:

```

var
  nilai:'a'..'e';
begin
  nilai:='a';
  writeln(ord(nilai)); {berapa yang dihasilkan, mengapa?}
  writeln(pred(nilai)); {mengapa diperbolehkan?}
end.

```

Tipe Boolean (*Boolean*)

Tipe boolean adalah sebuah tipe yang hanya dapat bernilai `false` atau `true`. Dalam Free Pascal mendukung tipe `Bytebool`, `WordBool` dan `LongBool`.

Karena masih merupakan tipe ordinal, maka nilainya dapat diketahui urutannya dengan menggunakan fungsi `ord`.

```
ORD(FALSE) = 0
```

```
ORD(TRUE) = 1
```

Operasi-operasi yang dapat dilakukan pada tipe boolean adalah (diurutkan berdasarkan hirarki teratas sampai terendah):

- NOT
- AND
- OR

Contoh dalam program:

```

Var
  X:boolean;
  A:byte;
Begin
  A:=0;
  If a>100 then x:=true else x:=true;
End.

var
  Ok:boolean;
  a,b:byte;
Begin
  a:=10;
  b:=20;
  Ok:=a=b;
End.

```

Pada Free Pascal, ekspresi boolean akan dievaluasi sedemikian sehingga pada saat hasilnya telah diketahui maka ekspresi selanjutnya tidak akan dievaluasi. Contoh berikut akan menyebabkan fungsi FUNC (fungsi yang menghasilkan nilai boolean) tidak dijalankan.

```

...
b:=false;
a:=b and FUNC;
...

```

Karena `b` bernilai `false`, maka `a` pasti akan bernilai `false`, sehingga `FUNC` tidak pernah dievaluasi.

Tipe Bilangan Bulat (*Integer*)

Ada 10 tipe bilangan bulat, yaitu:

Tipe	Range	Ukuran (byte)
Byte	0 .. 255	1

Shortint	-128 .. 127	1
Smallint	-32768 .. 32767	2
Word	0 .. 65535	2
Integer	Masuk dalam smallint, longint dan Int64	2, 4, 8
Cardinal	Masuk dalam word, longword, dan qword	2, 4, 8
Longint	-2147483648 .. 2147483647	4
Longword	0 .. 4294967295	4
Int64	-9223372036854775808 .. 9223372036854775807	8
Qword	0 .. 18446744073709551615	8

Tipe Integer secara standard akan diarahkan pada tipe smallint, sedangkan tipe cardinal selalu diarahkan pada tipe longword.

Tipe bilangan bulat ini dapat disebut sebagai *predefined type*, atau dapat juga dimasukkan dalam tipe subjangkauan.

Operasi-operasi yang dapat dilakukan pada tipe bilangan bulat adalah (diurutkan berdasarkan hirarki teratas sampai terendah):

- @, NOT
- *, /, div, mod, and, shl, shr
- +, -, or, xor
- =, <>, <, >, <=, >=, in

Operator pada bilangan bulat

Opera tor	Proses	Tipe operand	Tipe hasil
+	Penjumlahan	Bilangan bulat	Bilangan bulat
-	Pengurangan	Bilangan bulat	Bilangan bulat
*	Perkalian	Bilangan bulat	Bilangan bulat
/	Pembagian	Bilangan bulat	Real
DIV	Pembagian integer	Bilangan bulat	Bilangan bulat
MOD	Sisa Bagi	Bilangan bulat	Bilangan bulat

Evaluasilah ekspresi di bawah ini, tentukan apakah ekspresi tersebut benar atau salah, jika benar berapa nilai akhirnya, jika salah tentukan alasannya!

1. $3 + 5 * 2$
2. $6 \text{ or } 2 \text{ and } 5 \text{ div } 2$
3. $4 \text{ mod not } 6$
4. $5 \geq 6$
5. $10 \text{ and } 3 = 0$
6. $(10 \text{ and } 5 = 0) \text{ or } (4 * 1 = 4)$

```

var
  x:boolean;
  a,b:byte;
7.
begin
  a:=10;
  x:=true;
  if (a>=0) and x then write('a');
end.

```

```

8.
begin
  a:=4;

```

```

b:=a and 4;
x:=true;
if b and x then write('a');
end.

```

```

9.
begin
  if b=4 and a=1 then write('a');
end.

```

- **Tipe Real**

Sebuah tipe real adalah anggota bilangan real, yang dinyatakan dalam notasi *floating point*. Ada 6 model tipe real, masing-masing mempunyai jangkauan, ketepatan angka dan ukuran.

Type	Jangkauan	Digit	Ukuran
Real	$2,9 \cdot 10^{-39} .. 1,7 \cdot 10^{38}$	11-12	6 byte
Single	$1,5 \cdot 10^{-45} .. 3,4 \cdot 10^{38}$	7-8	4 byte
Double	$5,0 \cdot 10^{-324} .. 1,7 \cdot 10^{308}$	15-16	8 byte
Extended	$3,4 \cdot 10^{-4932} .. 1,1 \cdot 10^{4932}$	19-20	10 byte
Comp	$-2 \cdot 10^{63} + 1 .. 2 \cdot 10^{63} - 1$	19-20	8 byte

Ada 4 macam operator pada tipe real, yaitu:

Operator	Proses	Tipe operand	Tipe hasil
+	Penjumlahan	Real	Real
-	Pengurangan	Real	Real
*	Perkalian	Real	Real
/	Pembagian	Real	Real

B. Tipe Karakter (*Character Type*)

Tipe Karakter (*Character*)

Merupakan suatu tipe data untuk menyimpan karakter ASCII. Ditulis dengan menggunakan tanda petik atau diawali dengan #. Karena masih masuk dalam tipe ordinal maka dapat dioperasikan dengan menggunakan fungsi ord (untuk mengetahui urutannya), fungsi succ, fungsi pred, prosedur inc, prosedur dec. Tipe ini juga dapat menerima fungsi CHR (mengubah suatu bilangan bulat menjadi karakter yang sesuai dengan ASCII)

Contoh:

```

Var
  A:char;
Begin
  A:='A';
  A:=CHR(65);
  A:=#1;
End.

```

Tipe Untai (*String*)



A string represents a sequence of characters.

Format penulisan:

```
type
  identifier=string[panjang];
```

Bila panjang tidak didefinisikan, maka Pascal menganggap panjangnya adalah 255 karakter (panjang maksimum).

Contoh penggunaan dalam program

```
type
  str80=string[80]; {bisa dideklarasikan dalam tipe terlebih
dulu}
var
  a:str80;
  b:string[10]; {atau langsung dalam variabel}

const
  Heading: string[7] = 'Section';
  NewLine: string[2] = #13#10;
  TrueStr: string[5] = 'Yes';
  FalseStr: string[5] = 'No';
```

The standard function `Length` returns the number of characters in a string. Comparison of strings is defined by the ordering of the characters in corresponding positions. For example, "AB" is greater than "A"; that is, 'AB' > 'A' returns True. Zero-length strings hold the lowest values.

You can index a string variable just as you would an array. If `S` is a string variable and `i` an integer expression, `S[i]` represents the `i`-th character. The statement `MyString[2] := 'A'`; assigns the value A to the second character of `MyString`. The following code uses the standard `UpCase` function to convert `MyString` to uppercase.

```
var I: Integer;

begin
  I := Length(MyString);
  while I > 0 do
  begin
    MyString[I] := UpCase(MyString[I]);
    I := I - 1;
  end;
end;
```

Be careful indexing strings in this way, since overwriting the end of a string can cause access violations.

Tipe Untai Pendek (*Short String*)

Deklarasi String akan mendeklarasikan sebagai `shortstring` apabila:

8. Opsi dimatikan `{H-}`, maka deklarasi string akan selalu `shortstring`
9. Opsi dinyalakan `{H+}` dan diberikan panjang tertentu, maka deklarasi tersebut adalah `shortstring`.

Predefined type untuk `shortstring` adalah:

```
ShortString = String[255];
```

Tipe Anstrings

Format penulisan:


```
type
  identifier=ansistring;
```

Tipe ansistring adalah tipe string dengan panjang tanpa batas. Dalam proses kompilasi, ansistring diperlakukan sebagai pointer. Jika string adalah kosong (''), maka pointer akan menunjuk pada NIL. Jika ada isinya, pointer akan menunjuk suatu alamat pada heap memory.

Ketika tipe ansistring dideklarasikan, Free Pascal hanya mengalokasikan alamat untuk pointernya saja. Pointer ini pasti berisi NIL, sehingga isi mula-mula dari variabel ansistring pasti kosong (tidak mempunyai nilai apa-apa)

Tipe PChar

Format penulisan:

```
type
  identifier=PChar;
```

Pchar adalah variabel berjenis pointer yang mengarah ke sebuah array dengan tipe Char, yang diakhiri dengan karakter 0 (#0). Dengan tipe Pchar, dimungkinkan adanya operasi tambahan pada tipe ini.

Perhatikan cara pemakaian PChar pada program di bawah ini:

```
program one;
  var p : PChar;
  begin
    P := 'This is a null-terminated string.';
    WriteLn (P);
  end.
```

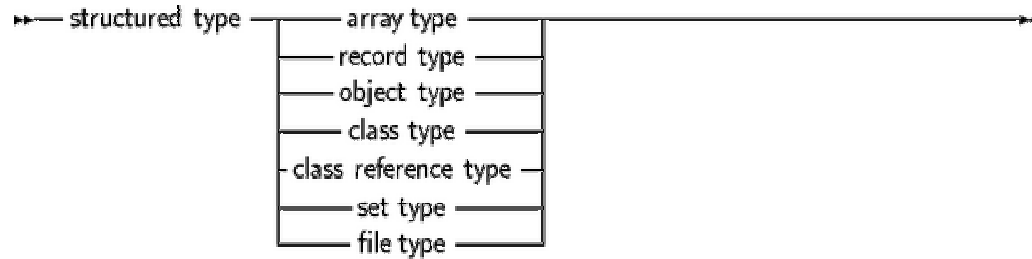
Akan sama dengan:

```
program two;
  const P : PChar = 'This is a null-terminated string.'
  begin
    WriteLn (P);
  end.
```

Mungkin juga nilai sebuah string diberikan pada Pchar, caranya:

```
Program three;
  Var S : String[30];
      P : PChar;
  begin
    S := 'This is a null-terminated string. '#0;
    P := @S[1];
    WriteLn (P);
  end.
```

C. Tipe Terstruktur (*Structured Type*)



A structured type, characterized by its component type(s), holds more than one value.

Free Pascal mendefinisikan beberapa tipe terstruktur ini, yaitu:

- Tipe Array



Tipe array adalah suatu tipe yang dapat menampung beberapa nilai dengan tipe yang sama dalam bentuk satu dimensi atau multidimensi. Tiap nilai dalam array dapat diacu dengan nama array dan indeksinya yang diletakkan dalam kurung.

Format penulisan:

```

type
  identifier=array[tipe_indeks] of tipe_data

```

Tipe_indeks adalah suatu tipe data **ordinal** (apa itu tipe data ordinal? Terdiri dari apa saja tipe data ordinal?)
 Tipe_data mendefinisikan bahwa array tersebut untuk menampung nilai dengan tipe data apa saja.

Contoh dalam program:

```

type
  a=array[boolean] of byte;
  c=5..10;
  b=array[1..10,c] of real; {termasuk tipe data apa c?}
var
  nilai_a:a;
  nilai_b:b;
begin
  nilai_a[true]:=10;
  nilai_b[1,5]:=10.4;
end.

```

Dalam contoh di atas, tipe_indeks apa yang digunakan? Berapa byte-kah yang digunakan program di atas untuk mendeklarasikan variabel dalam tipe tersebut?

- Tipe Himpunan (*Set*)



A set is a collection of values of the same ordinal type. The values have no inherent order, nor is it meaningful for a value to be included twice in a set.

Format penulisan:

```

type

```

```
identifier=set of ordinal_type;
```

Contoh dalam program:

```
type TIntSet = set of 1..250;
var Set1, Set2: TIntSet;
begin
  Set1 := [1, 3, 5, 7, 9];
  Set2 := [2, 4, 6, 8, 10]
End.
```

Operasi-operasi yang dapat diberlakukan pada tipe data set:

Operator	Operation	Operand types	Result type	Example
+	union	set	set	Set1 + Set2
-	difference	set	set	S - T
*	intersection	set	set	S * T
<=	subset	set	Boolean	Q <= MySet
>=	superset	set	Boolean	S1 >= S2
=	equality	set	Boolean	S2 = MySet
<>	inequality	set	Boolean	MySet <> S1
in	membership	ordinal, set	Boolean	A in Set1

Evaluasilah ekspresi di bawah ini. Tentukan nilai akhir untuk ekspresi tersebut.

```
Var
  A,B:set of char
Begin
  A:=[ 'A'..'E' ];
  B:=[ 'D'..'H' ];
End.
```

1. A + B
2. A - B
3. B - A
4. A * B
5. 'A' in B
6. A in B
7. 'A' <= B

- Tipe Record



Untuk Field list, diagram penulisannya:



Untuk fixed fields diagram penulisannya:



A record contains a number of components, or fields, that can be of different types.

Format penulisan:

```
Type identifier = record
  fieldList1: type1;
  ...
  fieldListn: typen;
end;
```

Contoh penulisan sebuah record:

```
Type
Point = Record
  X,Y,Z : Real;
end;
RPoint = Record
  Case Boolean of
    False : (X,Y,Z : Real);
    True : (R,theta,phi : Real);
  end;
BetterRPoint = Record
  Case UsePolar : Boolean of
    False : (X,Y,Z : Real);
    True : (R,theta,phi : Real);
  end;
TDateRec = record
  Year: Integer;
  Month: (Jan, Feb, Mar, Apr, May, Jun,
         Jul, Aug, Sep, Oct, Nov, Dec);
  Day: 1..31;
end;

var Record1, Record2: TDateRec;
```

This variable declaration creates two instances of TDateRec, called Record1 and Record2. You can access the fields of a record by qualifying the field designators with the record's name:

```
Record1.Year := 1904;
Record1.Month := Jun;
Record1.Day := 16;
```

Or use a with statement:

```
with Record1 do
begin
  Year := 1904;
  Month := Jun;
  Day := 16;
end;
```

You can now copy the values of Record1's fields to Record2:

```
Record2 := Record1;
```

- Tipe File



A file is an ordered set of elements of the same type.

Sintaks penulisan untuk tipe data file:

```
type fileTypeNamen = file of type
```

di mana `fileTypeNamen` is sembarang nama identifier dan tipe adalah **fixed-size type**. Contoh:

```
type
  PhoneEntry = record
    FirstName, LastName: string[20];
    PhoneNumber: string[15];
    Listed: Boolean;
  end;
  PhoneList = file of PhoneEntry;
Var
  Filephone: PhoneList;
```

Deklarasi di atas, mendeklarasikan file bertipe record. Deklarasi di atas, dapat juga secara langsung dituliskan:

```
var List1: file of PhoneEntry;
```

Tipe data file dapat juga berupa file tidak bertipe.

Format penulisan:

```
Var
  MyFile : File;
```

D. Tipe Pointer

⇨ pointer type = ^ - type identifier ⇨

A pointer is a variable that denotes a memory address. When a pointer holds the address of another variable, we say that it points to the location of that variable in memory or to the data stored there. In the case of an array or other structured type, a pointer holds the address of the first element in the structure.

Format penulisan:

```
type pointerTypeName = ^type
```

Contoh dalam program:

```
var
  X, Y: Integer; // X and Y are Integer variables
  P: ^Integer; // P points to an Integer
begin
  X := 17; // assign a value to X
  P := @X; // assign the address of X to P
  Y := P^; // dereference P; assign the result to Y
end;
```

E. Tipe prosedural (Procedural Types)

⇨ procedural type — function header — of - object — ; - call modifiers ⇨

Tipe prosedural pada Free Pascal berbeda dengan Turbo Pascal walaupun secara konsep sama. Berikut adalah contoh pemakaian dalam program:

```
Type TOneArg = Procedure (Var X : integer);
      TNoArg = Function : Real;
var proc : TOneArg;
    func : TNoArg;
```

9. Pengubah (*Variable*)

A variable is an identifier whose value can change at runtime. Variables are like containers for data, and, because they are typed, they tell the compiler how to interpret the data they hold. The declaration itself is made up of two parts: the left side is the name of the new variable, and the right side declares what the variable's type is. The two parts are separated by a colon (:).

Cara mendeklarasikan variabel adalah:

```
Var Identifierlist : type;
```

Identifierlist merupakan kumpulan identifier yang dipisahkan dengan tanda koma dan type adalah sembarang tipe data.

Contoh pendeklarasian variabel yang benar:

1.
Var
 a:byte;
2.
Var a,b:byte;
3.
Var
 X, Y, Z: Double;
 I, J, K: Integer;
 Digit: 0..9;
 Okay: Boolean;

Contoh pendeklarasian variabel yang salah:

1.
Var
 a,b;
2.
var
 a:8;

10. Konstanta (*Constant*)

While variables contain values that can change as your application runs, constants contain values that can't change. Constants are given a value at the time they are declared.

Contoh deklarasi konstanta

```
const  
Pi = 3.14159;  
Answer = 342;  
ProductName = 'Delphi';  
MaxData = 1024 * 64 - 16;  
NumChars = Ord('Z') - Ord('A') + 1;  
Message = 'Hello world...';
```

Konstanta seperti di atas disebut sebagai konstanta tak bertipe. Sedangkan konstanta yang bertipe mempunyai sifat seperti variabel.

Declare a typed constant like this:

```
const identifier: type = value
```

- **Mendeklarasikan konstanta bertipe untuk ARRAY:**

```
const Digits: array[0..9] of Char = ('0', '1', '2', '3', '4',  
'5', '6', '7', '8', '9');  
Factorial: array[1..7] of Integer = (1, 2, 6, 24, 120, 720,  
5040);
```

- **Mendeklarasikan konstanta bertipe untuk RECORD**

```
type  
  Point = record  
    X, Y: Real;  
  end;  
  Vector = array[0..1] of Point;  
  Month = (Jan, Feb, Mar, Apr, May, Jun, Jly, Aug, Sep, Oct, Nov,  
Dec);  
  Date = record  
    D: 1..31;  
    M: Month;  
    Y: 1900..1999;  
  end;  
const  
  Origin: Point = (X: 0.0; Y: 0.0);  
  Line: Vector = ((X: -3.1; Y: 1.5), (X: 5.8; Y: 3.0));  
  SomeDay: Date = (D: 2; M: Dec; Y: 1960);
```

- **Mendeklarasikan konstanta bertipe untuk SET**

```
type  
  Digits = set of 0..9;  
  Letters = set of 'A'..'Z';  
const  
  EvenDigits: Digits = [0, 2, 4, 6, 8];  
  Vowels: Letters = ['A', 'E', 'I', 'O', 'U', 'Y'];  
  HexDigits: set of '0'..'z' = ['0'..'9', 'A'..'F', 'a'..'f'];
```

- **Mendeklarasikan konstanta bertipe untuk tipe data sederhana**

```
const  
  Maximum: Integer = 9999;  
  Factor: Real = -0.1;  
  Breakchar: Char = #3;
```

- **Mendeklarasikan konstanta bertipe untuk tipe data string**

```
const  
  Heading: string[7] = 'Section';  
  NewLine: string[2] = #13#10;  
  TrueStr: string[5] = 'Yes';  
  FalseStr: string[5] = 'No';
```

1. Operasi Aritmatika

Operasi Aritmatika adalah operasi untuk melaksanakan suatu proses perhitungan. Hasil dari operasi ini diberikan pada variabel yang dapat menampung nilai numerik.

Seperti yang telah ditulis pada pembahasan tipe bilangan bulat dan real, operator pada operasi aritmatika adalah:

OPERATOR	OPERASI	TIPE OPERAN	TIPE HASIL
+	Penjumlahan	Tipe bilangan bulat, tipe real	Tipe bilangan bulat, tipe real
-	Pengurangan	Tipe bilangan bulat, tipe real	Tipe bilangan bulat, tipe real
*	Perkalian	Tipe bilangan bulat, tipe real	Tipe bilangan bulat, tipe real
/	Pembagian	Tipe bilangan bulat, tipe real	Tipe real
Div	Pembagian bilangan bulat	Tipe bilangan bulat	Tipe Bilangan bulat
Mod	Sisa bagi	Tipe bilangan bulat	Tipe Bilangan bulat

Operator + juga dipergunakan untuk operasi string dan operasi himpunan (set)
Operator +, - dan * juga dipergunakan untuk operasi himpunan (set)

Yang perlu diperhatikan dalam operator aritmatika adalah:

- Tipe bilangan bulat dapat diberikan kepada tipe real, tapi tidak berlaku sebaliknya.

Contoh:

```
Var
  A:real;
  L:Longint;
begin
  A := L;           {statement yang benar}
  L := A;           {statement yang salah}
End.
```

- Tidak boleh ada dua atau lebih operator yang berurutan dalam sebuah ekspresi aritmatika.

Contoh:

```
Var
  A:real;
  L:Longint;
Begin
  A := A + 1;      {statement yang benar}
  A := L++;        {statement yang salah}
End.
```

2. Operasi Logika

Operasi logika menggunakan sistem bilangan biner untuk perhitungannya. Ada 6 buah operator logika yang dikenal Pascal, yaitu:

OPERATOR	OPERASI	TIPE OPERAN	TIPE HASIL
NOT	Ingkaran bit	Tipe bilangan bulat	Tipe bilangan bulat

AND	Logika AND bit	Tipe bilangan bulat	Tipe bilangan bulat
OR	Logika OR bit	Tipe bilangan bulat	Tipe bilangan bulat
XOR	Logika XOR bit	Tipe bilangan bulat	Tipe bilangan bulat
SHL	Geser bit ke kiri	Tipe bilangan bulat	Tipe Bilangan bulat
SHR	Geser bit ke kanan	Tipe bilangan bulat	Tipe Bilangan bulat

3. Operasi Boolean

Operasi boolean menggunakan operand yang bertipe boolean serta hasil boolean untuk operasinya. Ada 4 operator boolean yang dikenal Pascal, yaitu:

OPERATOR	OPERASI
NOT	Ingkaran bit
AND	Logika AND
OR	Logika OR
XOR	Logika XOR

4. Operasi String

Operasi string pada Pascal hanya mengenal sebuah operator yaitu operator + atau penggabungan antar string.

5. Operasi himpunan

Pada operasi himpunan selalu menggunakan tipe data himpunan (set) untuk operator maupun operannya. Ada 3 macam operator pada operasi himpunan, yaitu:

OPERATOR	OPERASI
+	Union
-	Selisih
*	Interseksi

Hasil dari operasi-operasi himpunan berdasarkan aturan sebagai berikut:

- Nilai ordinal C ada di A+B hanya jika C ada di A atau B
- Nilai ordinal C ada di A-B hanya jika C terdapat di A dan bukan di B
- Nilai ordinal C ada di A*B hanya jika C terdapat baik di A maupun B

6. Operasi Relasi

Operasi ini digunakan untuk melakukan perbandingan dan menghasilkan boolean (true atau false) yang menunjukkan perbandingan tersebut bernilai benar atau salah.

OPERATOR	OPERASI	TIPE OPERAN	TIPE HASIL
=	Sama dengan	Tipe sederhana, pointer, himpunan dan string	Boolean
<>	Tidak sama dengan	Tipe sederhana, pointer, himpunan dan string	Boolean
<	Lebih kecil	Tipe sederhana, pointer, himpunan dan string	Boolean
>	Lebih besar	Tipe sederhana, pointer, himpunan	Boolean

		dan string	
<=	Lebih kecil sama dengan	Tipe sederhana, pointer, himpunan dan string	Boolean
>=	Lebih besar sama dengan	Tipe sederhana, pointer, himpunan dan string	Boolean
<=	Subset dari	Himpunan	Boolean
>=	Superset dari	Himpunan	Boolean
In	Anggota dari	Operan kiri: sembarang tipe ordinal Operan kanan: himpunan	Boolean

Statement adalah perintah yang dikenal oleh Pascal. Dalam bahasa Pascal terdapat 11 statement, artinya SEMUA program dalam Pascal hanya menggunakan kombinasi dari ke-11 statement ini saja.

Statement-statement yang dikenal Pascal:

1. Assignment (pemberian nilai)
2. Compound (penggabungan)
3. IF – THEN – ELSE
4. CASE – OF
5. FOR – TO – DO
6. REPEAT – UNTIL
7. WHILE – DO
8. WITH
9. Procedure Call
10. Goto
11. Inline

Statement GOTO adalah statement yang jarang digunakan karena dianggap sebagai statement yang tidak sesuai dengan konsep pemrograman terstruktur. Sedangkan statement inline adalah statement untuk menjalankan instruksi bahasa mesin dan tidak digunakan dalam IOI.

1. Assignment (Pemberian nilai)

Statement Assignment digunakan untuk memberikan nilai pada sebuah variabel.

Sintaks penulisan:

Variabel := nilai

Contoh 1:

```
var a,b:byte;
begin
  a:=10;
  a:=a*4;
  if a>5 then b:=40;
end.
```

Contoh 2:

```
var a,b:byte;
begin
  a:=0;
  a=a+4;
end.
```

Ada berapa assignment statement dari contoh di atas?

Catatan untuk FreePascal

Assignment	Hasil
A += b	Menambah b ke a, hasil disimpan di a.
A -= b	Mengurangi b dari a, hasil disimpan di a.
A *= b	Mengalikan a dengan b, hasil disimpan di a.
A /= b	Membagi a terhadap b, hasil disimpan di a.

2. Compound Statement

Digunakan untuk menggabungkan beberapa statement.

Dalam teks lain disebutkan:

Compound statements are a group of statements, separated by semicolons, that are surrounded by the keywords Begin and End. The Last statement doesn't need to be followed by a semicolon, although it is allowed.

Sintaks Penulisan:

```
begin
  statement;
  statement;
  ...
  statement
end
```

Contoh:

```
if a>5 then
begin
  a:=a*4;
  b:=a-4
end;
```

- Ada berapa statement assignment pada potongan program di atas?
- Ada berapa statement pada potongan program di atas?
- Mana yang disebut compound statement?

3. IF – THEN – ELSE Statement

Merupakan perintah percabangan yang akan menjalankan statement sesuai dengan kondisi yang ada.

Ada 2 macam sintaks penulisan IF – THEN – ELSE:

```
IF kondisi THEN statement
```

Dan

```
IF kondisi THEN statement ELSE statement
```

Contoh:

```
If a>5 then a:=a+1;
If a-4=2 then begin a:=2;b:=a end;
if (a=3) and (b=2) then
begin
  a:=a-1;
  b:=a
end else b:=3;

if a>5 then a:=4
else begin
  a:=4 mod b;
  b:=b-1
end;
```

- Tentukan kondisi-kondisi pada IF Statement yang ada!
- Apakah compound statement digunakan pada contoh di atas?
- Apakah assignment statement digunakan pada contoh di atas?
- Jika ya, tentukan di mana letak compound dan assignment statement!

4. FOR – TO / DOWNTO – DO Statement

Digunakan untuk mengulang statement

Sintaks penulisan

```
FOR variabel := awal TO akhir DO statement
```

Atau

```
FOR variabel := akhir DOWNTO awal DO statement
```

Catatan:

- Statement akan diulang sebanyak akhir-awal+1
- Selama perulangan, nilai variabel akan bernilai dari awal sampai akhir

Contoh:

```

var          var
a:byte;      a,b:byte;
begin        begin
  for a:=1 to 5 do      for a:=5 downto 2 do
    write(a)           for b:=1 to a do write(a+b)
  end.                 end.

```

- write(a) pada contoh 1, merupakan statement procedure call (pemanggilan prosedur)
- Pada contoh 2, setelah for a:=5 downto 2 do, statement berikutnya adalah statement for – to – do.
- Berapa kali statement write(a+b) dieksekusi?
- Tuliskan output dari contoh 2.

5. CASE OF Statement

Statement case digunakan untuk perintah percabangan dengan banyak kondisi. Terdiri dari ekspresi (atau biasa disebut dengan selector) dan serangkaian statement.

Sintaks penulisan:

```

case expression of
  case: statement;
  ...
  case: statement;
end

```

Atau:

```

case expression of
  case: statement;
  ...
  case: statement;
else
  statement
end

```

Catatan:

Jika pada sebuah kondisi dapat terdiri dari banyak range, maka dapat kondisi-kondisi yang ada dapat dipisahkan dengan koma.

Contoh:

```

case Ch of
  'A'..'Z', 'a'..'z': WriteLn('Letter');
  '0'..'9':           WriteLn('Digit');
  '+', '-', '*', '/': WriteLn('Operator');
else
  WriteLn('Special character');
end;

```

6. Repeat – Until Statement

Perintah yang ada di dalam statement repeat – until akan diulang sehingga kondisi boolean pada until bernilai true.

Sintaks penulisan:

```

repeat
  statement;

```

```
statement;  
...  
statement  
until expression
```

Contoh:

```
repeat Ch := GetChar until Ch <> ' ';  
  
repeat  
  Write('Enter value: ');  
  ReadLn(I);  
until (I >= 0) and (I <= '9');
```

Catatan: GetChar pada contoh pertama merupakan statement procedure call.

7. While – Do Statement

Statement while terdiri dari sebuah ekspresi boolean yang mengontrol eksekusi dari sebuah statement.

Sintaks Penulisan:

```
while Ch = ' ' do Ch := GetChar;  
  
while not Eof(InFile) do  
begin  
  ReadLn(InFile, Line);  
  WriteLn(OutFile, Line);  
  Inc(LineCount);  
end;
```

Catatan:

Contoh kedua adalah statement while yang menggunakan compound statement di dalamnya.

8. Statement With

Statement with digunakan untuk mereferensi field pada sebuah record.

Sintaks Penulisan:

```
with var, var, ... var do statement
```

Contoh:

```
with Date[I] do  
begin  
  month := 1;  
  year := year + 1;  
end;
```

Ekivalen dengan perintah:

```
Date[I].month := 1;  
Date[I].year := Date[I].year + 1;
```

Konsep subprogram atau modul merupakan konsep yang sangat bermanfaat dalam penyusunan sebuah program yang besar. Subprogram atau modul berkembang sejalan dengan teknik pemrograman terstruktur, yaitu membagi masalah yang besar menjadi masalah-masalah kecil yang diselesaikan dalam sebuah subprogram. Jika subprogram telah benar untuk memecahkan masalah-masalah kecil, maka diharapkan dalam penyusunan program yang besar, program dapat lebih disederhanakan sehingga memudahkan memecahkan masalah utama.

Dalam Pascal ada 2 macam subprogram, yaitu prosedur (*procedure*) dan fungsi (*function*).

Bentuk umum penulisan prosedur atau fungsi dalam Pascal adalah:

```
PROGRAM      nama(file1,file2,file3);
CONST       deklarasikonstanta;
VAR         deklarasivariabel;
TYPE        deklarasitype;
LABEL       deklarasilabel;
FUNCTION    deklarasifungsi;
PROCEDURE   deklarasiprosedur;

PROCEDURE lain1;
BEGIN
  statement1;
  statement2;
  ...
END;

FUNCTION lain2:tipedata;
BEGIN
  statement1;
  statement2;
  ...
  lain2:=nilai;
END;

BEGIN
  statement1;
  statement2;
  statement3;
  ...
END.
```

1. Procedure

Prosedur mempunyai struktur yang sama dengan struktur program, yaitu terdiri dari nama prosedur, deklarasi-deklarasi dan bagian utama dari prosedur itu sendiri. Di dalam prosedur tersebut, dimungkinkan ada prosedur lagi yang strukturnya sama. Bentuk prosedur dalam prosedur tersebut disebut dengan prosedur tersarang (*nested procedure*).

Struktur penulisan prosedur adalah sebagai berikut:

```
PROCEDURE namaprocedure(daftarparameterformal);
  {bagian deklarasi}
BEGIN
  Statement1;
  Statement2;
  Statement3;
  ...
END;
```

Daftar parameter formal dan tanda kurung (setelah namaprocedure) bersifat opsional. Artinya bisa digunakan, bisa juga tidak. Parameter formal terdiri dari 2 macam, yaitu:

- Parameter nilai (*value parameter*)
- Parameter perubah (*variable parameter*)

Sebelum membahas kedua parameter tersebut, maka sebelumnya harus diketahui bagaimana mekanisme suatu pemanggilan prosedur. Misalnya diketahui sebuah program sebagai berikut:

```
USES crt;

PROCEDURE tulis(x,y:integer;s:string);
BEGIN
  GOTOXY(x,y);
  WRITE(s);
END;

BEGIN
  Tulis(10,2,'Hello World');
END.
```

Dari program di atas, yang dimaksud dengan **parameter aktual** adalah 10, 2, dan 'Hello World' (pada pemanggilan prosedur tulis). Sedangkan pada deklarasi prosedur tulis, x, y, dan s disebut sebagai **parameter formal**.

Mekanisme pemanggilan tulis tersebut adalah:

Program bekerja dari program utama, yaitu memanggil procedure tulis. Pada waktu terjadi pemanggilan procedure tulis, maka isi parameter aktual diberikan diberikan pada parameter formalnya. Nilai ini mungkin diproses dalam procedure tersebut. Nilai terakhir dari parameter formal tersebut **TIDAK** dikembalikan pada parameter aktualnya. Parameter seperti inilah yang disebut dengan **parameter nilai**. Yang dapat digunakan sebagai parameter nilai adalah: konstanta, variabel dan ekspresi.

```
VAR x,y:integer;

PROCEDURE tukar(var a,b:integer);
  Var c:integer;
BEGIN
  c:=a;
  a:=b;
  b:=c;
END;

BEGIN
  x:=5;
  y:=1;
  tukar(x,y);
END.
```

Parameter aktual dari program di atas adalah x dan y. Sedangkan parameter formalnya adalah a dan b. Dalam parameter formal, terdapat kata var yang menunjukkan bahwa parameter tersebut adalah **parameter perubah**.

Pada waktu procedure yang mempunyai *parameter perubah* dipanggil, parameter aktual (x dan y) pada pemanggilnya akan digantikan parameter formal. Jadi yang diolah pada procedure di atas adalah parameter aktualnya (x dan y). Konsekuensi pemanggilan procedure dengan menggunakan *parameter perubah* adalah setiap perubahan nilai yang terjadi dari parameter aktual (di dalam procedure) akan mengakibatkan turut berubahnya nilai parameter aktual (di dalam pemanggil atau program utama).

Yang dapat dijadikan *parameter perubah* adalah hanya variabel, sedangkan konstanta dan ekspresi TIDAK dapat dijadikan sebagai *parameter perubah*.

2. Function

Sebenarnya secara prinsip, fungsi sama seperti prosedur (sama-sama merupakan subprogram). Perbedaan fungsi dan prosedur adalah fungsi dapat langsung memberikan nilai kembali sedangkan pada procedure harus menggunakan parameter perubah agar dapat mengembalikan suatu nilai.

Bentuk umum suatu fungsi dalam Pascal adalah sebagai berikut:

```
FUNCTION namafungsi(daftarparameterformal):tipe_data;
    {bagian deklarasi}
BEGIN
    Statement1;
    Statement2;
    Statement3;
    ...
    namafungsi:=nilai;
END;
```

Dalam fungsi, semua parameter formal sebaiknya merupakan parameter nilai walaupun diperbolehkan menggunakan parameter perubah.

3. Rekursi

Proses rekursi adalah suatu proses di mana sebuah subprogram dapat memanggil dirinya sendiri. Penggunaan rekursi ini akan sangat membantu pemecahan-pemecahan soal tidak dapat diselesaikan dengan menggunakan teknik iterasi biasa. Karena rekursi merupakan pemanggilan dirinya sendiri (procedure atau fungsi), maka dalam rekursi tersebut haruslah ada yang namanya proses terminasi / point sentinel (kapan berhenti memanggil dirinya sendiri), ini menghindari proses rekursi yang tak terdefiniskan (terus menerus memanggil dirinya sendiri). Proses terminasi ini biasanya berupa kondisi.

Perhatikan program di bawah ini:

```
FUNCTION jumlah(n:integer):integer;
BEGIN
    IF n=1 THEN
        jumlah:=1;
    ELSE
        jumlah:=jumlah(n-1)+n;
    END;

    BEGIN
        writeln(jumlah(3));
    END.
```

Cara kerja rekursi adalah secara terbalik / mundur sehingga terbentur dengan proses terminasi atau point sentinel. Setelah terbentur dengan point sentinel, maka berikutnya proses akan berjalan maju dengan membawa proses perhitungan pada saat proses terbentur dengan point sentinel tadi.

Pada proses di atas, point sentinel adalah pada saat $n=1$. Selama n tidak sama dengan 1, maka function jumlah akan terus menerus dipanggil sampai terbentur dengan point sentinel (yaitu $n=1$).

Jika function jumlah dipanggil dengan:

jumlah(3),

Maka proses yang terjadi adalah:

- Jumlah(3) akan memanggil:
jumlah(2) + 3
- Jumlah(2) akan memanggil:
Jumlah(1) + 2

- Jumlah(1) akan menghasilkan 1 (karena terbentur dengan point sentinel, sehingga tidak memanggil dirinya sendiri lagi).
- Setelah terbentur dengan point sentinel (dengan membawa hasil 1), maka proses akan berjalan maju untuk menyelesaikan tugasnya, yaitu kembali pada:
- Jumlah(1) + 2, karena jumlah(1) membawa atau bernilai 1, maka pada proses ini akan menghasilkan nilai 3. Nilai 3 ini akan dibawa ke pemanggil berikutnya, yaitu jumlah(2).
- Pada jumlah(2) + 3, akan terjadi proses $3 + 3$ (karena jumlah(2) membawa hasil 3), sehingga pemanggilana jumlah(3) akan menghasilkan 6.

SELEKSI AWAL TINGKAT KABUPATEN-KOTAMADYA 2004
(Teori Dasar Pemrograman Pascal)

Soal 51.

```
var i,k: integer;
begin
  i:=5; k:=0;
  k:=trunc(sqrt(i))+1;
  writeln(k);
end.
```

Apa keluaran program di atas ?

- a. 3
- b. 2.24
- c. 2
- d. 0
- e. program tidak dapat dijalankan

Jawab:

- a. 3

Pembahasan:

Fungsi sqrt :

Deklarasi :Function Sqrt (X : Real) : Real;
Keterangan : menghasilkan akar pangkat dua dari x, di mana x harus positif

Fungsi trunc

Deklarasi :Function Trunc (X : Real) : Longint;
Keterangan : menghasilkan bilangan bulat dari X, yang akan selalu lebih kecil atau sama dengan X.

Sqrt(5) akan menghasilkan 2.23
Trunc(2.23) akan menghasilkan 2
Sehingga k:=trunc(sqrt(i))+1; akan menghasilkan 3

Soal 52.

```
const
  Data: array [1..3,1..3] of char=
    (('1','1','2'),('2','2','4'),('4','4','8'));

var i, j : byte;
begin
  for i:= 1 to 3 do
  begin
    for j:=3 downto 1 do
      write(Data[i,j]):
      writeln;
    end;
  end.
end.
```

Apa keluaran program di atas ?

- a. 112
224
448
- b. '1''1''2'
'2''2''4'
'4''4''8'
- c. 211
422
844
- d. '2''1''1'
'4''2''2'
'8''4''4'
- e. 124
124
248

Jawab:

- c. 211
422
844

Soal 53.

```
function ABC(k : integer);
var i,j : integer;
begin
  j:=1;
  for i:=1 to k do
    j:=j*2;
    ABC:=j;
  end;

begin
  writeln(ABC(4));
end.
```

Apa keluaran program di atas ?

- a. 16
- b. 8
- c. 4
- d. 2
- e. program tidak dapat dijalankan

Jawab:

- e. program tidak dapat dijalankan

Pembahasan:

Soal ini merupakan soal jebakan. Perhatikan bahwa deklarasi dari sebuah function adalah:
Function namafunction(parameter):tipe;

Sebuah function harus mempunyai nilai kembali. Namun pada soal tidak terdapat nilai kembali.

Soal 54.

```
var s:string;
begin
  s:='TOKI GO GET GOLD!';
  delete(s,1,length(s)-12);
```

```
writeln(s);  
end.
```

Apa keluaran program di atas ?

- GO GET GOLD!
- GO GET GOLD!
- GET GOLD!
- TOKI GO GET
- TOKI GO GE

Jawab:

- GO GET GOLD!

Pembahasan:

Procedure delete:

Deklarasi : procedure Delete (var S: String; Index: Integer;
Count: Integer);

Keterangan : procedure delete akan menghapus S sebanyak count karakter, dimulai dari posisi Index.

Function length:

Deklarasi: Function Length (S : String) : Integer;

Keterangan: Length menghasilkan panjang dari S, bernilai antara 0 sampai dengan 255. Jika S tidak berisi apa-apa maka akan menghasilkan 0.

Statement delete(s,1,length(s)-12) akan menghapus s dari posisi 1 sebanyak panjang s, yaitu 17-12 = 5. Sehingga yang dihapus adalah karakter 'TOKI ' dan s akan bernilai GO GET GOLD!

Soal 55.

```
type tTable=array[1..6] of integer;  
  
procedure Xsort(A,B:tTable;var C:tTable;i,j,k:integer);  
begin  
  repeat  
    if A[i]<B[j] then  
      begin  
        inc(k);C[k]:=A[i];  
        inc(i);  
      end  
    else  
      begin  
        inc(k);C[k]:=B[j];  
        inc(j);  
      end;  
    until k=N;  
end;
```

Apa nama algoritma sort di atas dimana N adalah jumlah data A ditambah data B.

- bubble sort
- quick sort
- insertion sort
- Merge Sort
- shell sort

Jawab:

- Merge Sort

Soal 56.

```

var
  h,i,j:integer;

begin
  h:=0;
  for j:=1 to 10 do
    inc(h);i:=1;
  j:=0;
  repeat
    inc(i);
    inc(j);
  until i=10;
  if h=j then writeln('GOLD')
  else writeln('SILVER');
end.

```

Apa keluaran program di atas

- a. GOLD
- b. SILVER
- c. GOLD
SILVER
- d. SILVER
GOLD
- e. tidak ada keluaran

Jawab:

b. SILVER

Pembahasan:

```

h:=0;
for j:=1 to 10 do
  inc(h);i:=1;
j:=0;

```

Saat selesai mengeksekusi statement di atas, maka h akan bernilai 10, i bernilai 1 dan j bernilai 0. Selanjutnya program akan mengeksekusi statement:

```

repeat
  inc(i);
  inc(j);
until i=10;

```

Karena nilai i sebelumnya adalah 1, maka statement inc(i) dan inc(j) akan dijalankan sebanyak 9 kali, sehingga setelah keluar dari perulangan repeat until, nilai i akan bernilai 10 dan j bernilai 9.

```

if h=j then writeln('GOLD')
else writeln('SILVER');

```

Statement if di atas akan menjalankan statement sesudah else (karena nilai h tidak sama dengan j).

Soal 57.

```

var r : real;
begin
  r:=147.0;
  writein(r:0:5),
end.

```

Apa tampilan program di atas ?

- a. 00147
- b. 0147.0
- c. 147.000
- d. 147.0000
- e. 147.00000

Jawab:

e. 147.00000

Pembahasan:

Penulisan bilangan real pada statement writeln adalah:

```
writeln(OutputVariable : NumChars [: Decimals ])
```

Penulisan bilangan real pada statement writeln adalah:

NumChars adalah banyaknya space yang disiapkan untuk penulisan, dan Decimals adalah banyaknya angka desimal (angka di belakang koma).

Soal 58.

```
procedure ABC(n,k:integer;s:string);
var
  l:integer;
  x:char;
begin
  if n=k then writeln(s)
  else
    begin
      x:=s[l];s[l]:=s[n];s[n]:=x;
      ABC(n+1,k,s);
    end;
end;

begin
  s:='ABC';
  ABC(1,3,s);
end.
```

Apa keluaran program di atas ?

- a. ABC
ACB
BAC
BCA
CAB
CBA
- b. CBA
CAB
BCA
BAC
ACB
ABC
- c. CAB
CBA
BAC
BCA
ABC
ACB
- d. ACB
ABC

- BCA
- BAC
- CBA
- CAB
- e. ABC
- BAC
- CAB
- ACB
- BCA
- CBA

Jawab:

-

Pembahasan:

Kemungkinan program ini maksudnya adalah membuat sebuah procedure permutasi yang akan mempermutasikan string ABC. Namun ada baris program yang kurang sehingga tampilan di layar tidak menghasilkan permutasi.

Soal 59.

```
Begin
  Writeln((10 shr 1) shl 2);
end.
```

Apa yang dihasilkan oleh program diatas...

- a. 18
- b. 19
- c. 20
- d. 21
- e. 22

Jawab:

c. 20

Pembahasan:

Operator SHR adalah operasi pergeseran bit ke kanan dan operasi shl adalah operasi pergeseran bit ke kiri.

$10 \text{ shr } 1 = 5 \rightarrow (1010 \text{ shr } 1 = 0101 = 5)$
 $5 \text{ shl } 2 = 20 (0101 \text{ shl } 2 = 10100 = 20)$

Soal 60.

```
Type
  PByte = ^Byte;
Var
  A: Word;
Begin
  A:=$FFFF,
  Writeln(PByte(A)^);
end.
```

Apa yang dihasilkan oleh program diatas...

- a. 32
- b. 63
- c. 127
- d. 255
- e. \$FFFF

Jawab:

d. 255

Pembahasan:

Statement penulisan `Writeln(PByte(A)^)`; adalah statement untuk menuliskan isi dari variabel A namun sebatas ukurang dari PByte, yaitu Byte. Nilai \$FFFF berukuran 16 bit (atau 2 byte), sehingga yang jika ditulis sebesar ukuran Byte, maka yang muncul adalah 255 (atau \$FF)

Soal 61.

```
Function Sum(const A,B:Integer): Integer;
Begin
    Sum:=A+B;
end;

Begin
    Writeln(Sum(5,10));
end.
```

Apa yang dihasilkan oleh program diatas.

- a. 5
- b. 10
- c. 15
- d. 20
- e. Tidak bisa di compile.

Jawab:

c. 15

Pembahasan:

Parameter const a,b:integer merupakan parameter konstanta (constant parameter) yang sama seperti parameter biasa.

Soal 62.

```
var
    A,B:string;
    C:string[10];

begin
    A:='TOKI MEMANG';
    B:='HEBAT';
    C:=A+B;
    if (Pos(B)>0) then
    Begin
        Writeln('A');
    end else
        Writeln('B');
end.
```

Apa yang terjadi jika program di atas di jalankan...

- a. Huruf 'A' tercetak
- b. Huruf 'B' tercetak
- c. Tidak dapat dipastikan
- d. Terjadi error
- e. Tidak bisa di compile

Jawab:

e. Tidak bisa di compile

Pembahasan:

Kesalahan pertama yang akan ditemui program adalah pada function pos.

Deklarasi: `Function Pos (Substr : String; S : String) : Integer;`

Keterangan: function pos akan menghasilkan urutan atau posisi substr di S. Jika tidak ditemukan, maka akan menghasilkan 0.

Pada program function pos hanya terdiri dari 1 parameter saja sehingga program tidak akan dapat dijalankan.

Soal 63.

```
Var
  i, j: Shortint;
Begin
  for i:=1 to 200 do
    Inc(j);
  end.
```

Program diatas akan mengbasilkan:

- a. Nilai j = 200;
- b. Nilai j = 127;
- c. Tidak dapat dipastikan
- d. Terjadi error (code 76).
- e. Program tidak dapat berhenti

Jawab:

d. Terjadi error (code 76).

Pembahasan:

Shortint adalah sebuah tipe data yang mempunyai jangkauan -128..127. Jika dipaksa diberikan nilai hingga 200, maka yang muncul adalah Error 76: Constant out of range.

Soal 64.

```
Begin
  Assign(Output, 'ABC.TXT');
  Rewrite(Outptut);
  Write('GO GET GOLD');
  Close(Output);
end.
```

Program diatas akan menghasilkan:...

- a. File 'ABC' dengan isi 'GO GET GOLD'
- b. Run-time error.
- c. Tidak dapat dicompile karena variable 'Output' tidak ada
- d. Tidak menghasilkan apa-apa
- e. Tidak ada yang benar

Jawab:

a. File 'ABC' dengan isi 'GO GET GOLD'

Pembahasan:

Input dan Output adalah variabel standard yang telah ada pada unit sistem, sehingga tidak perlu dideklarasikan lagi.

Soal 65.

```
function A:boolean;
begin
```

```

    write('A');
    A:=true;
end;

function B:boolean;
begin
    write('B');
    B:=False;
end;

begin
    if (A and B) then
        Write('C');
        Writeln('D')
    end.

```

Program di atas akan menghasilkan:

- a. Huruf ABC
- b. Huruf BAC
- c. Huruf ABC
- d. Huruf ABD
- e. Huruf ABCD

Jawab:

d. Huruf ABD

Soal 66.

```

procedure Sum(Var a,b:integer; c:Integer);
Begin
    a:=1;
    b:=2;
    c:=a+b;
end;

Var
    i,j,k: Integer;
Begin
    i:=10;
    j:=20;
    k:=30;
    Sum(i, j, k);
end.

```

Isi variabel i,j,k berturut-turut

- a. 10, 20, 30
- b. 1, 2, 3
- c. 10, 2, 30
- d. 1, 2, 30
- e. 10, 20, 3

Jawab:

d. 1, 2, 30

Pembahasan:

Dalam parameter di procedure sum terdapat kata var a,b:integer. Var di sini berarti akan membuat nilai a dan b akan ikut berubah setelah pemanggilan procedure tersebut. Sedangkan variabel c nilainya tetap.

Soal 67.

```

Var A: Real;
Begin
  A:=1000.52345
  Writeln(Round(A));
  Writeln(Trunc(B));
  Writeln( Frac(C));
end.

```

Output program diatas berturut-turut:

- 1000, 1001, 52345
- 1001, 1001, 52345
- 1001, 1000, 0.52345
- 1000, 1001, 52345
- 1000.52345, 1000.52345, 1000

Jawab:

-

Pembahasan:

Fungsi round

Deklarasi :Function Round (X : Real) : Longint;
 Keterangan : membulatkan bilangan X, yang mungkin lebih besar atau lebih kecil dari X.

Fungsi trunc

Deklarasi :Function Trunc (X : Real) : Longint;
 Keterangan : menghasilkan bilangan bulat dari X, yang akan selalu lebih kecil atau sama dengan X.

Fungsi frac

Deklarasi :Function Frac (X : Real) : Real;
 Keterangan : mengambil bilangan selain bilangan bulat dari X (angka di belakang koma)

Dalam program terdapat statement .

```

Writeln(Round(A));
Writeln(Trunc(B));
Writeln( Frac(C));

```

Di mana B dan C tidak dikenali.

Soal 68.

```

function A(const a:Longint):Boolean;
var
  x,y:Longint;
Begin
  x:=Trunc(Sqrt(a))+1;
  y:=2;
  while x<>y do
    if (a mod y)>0 then Inc(y) else break;
  A:=x=y;
end;

```

Begin

```

if A(1) then Write('A');
if A(2) then Write('B');
if A(3) then Write('C');
if A(4) then Write('D');
if A(5) then Write('E');

```

end.

Program diatas akan menghasilkan:...

- a. Huruf ABCDE
- b. Huruf AB
- c. Huruf ABC
- d. Huruf ADE
- e. Huruf BCE

Jawab:

-

Pembahasan:

Dalam program di atas, terdapat 2 identifer yang sama yaitu A, yang digunakan sebagai nama function dan parameter. Program di atas, tidak dapat dijalankan.

Soal 69.

```
begin
  writeln(round(frac(3.7)));
end.
```

Apa keluaran program di atas ?

- a. 0
- b. 1
- c. 2
- d. 3
- e. 4

Jawab:

b. 1

Pembahasan:

Fungsi frac

Deklarasi :Function Frac (X : Real) : Real;

Keterangan : mengambil bilangan selain bilangan bulat dari X (angka di belakang koma)

Fungsi round

Deklarasi :Function Round (X : Real) : Longint;

Keterangan : membulatkan bilangan X, yang mungkin lebih besar atau lebih kecil dari X.

Frac(3.7) akan menghasilkan 0.7

Round(0.7) akan menghasilkan 1

Soal 70.

```
var i, j:byte;
begin
  i:=100; j:=200;
  writeln(i*j)
end.
```

Apa tampilan program di atas?

- a. 12
- b. 22
- c. 32
- d. 20000
- e. 400000

Jawab:

d. 20000

Soal 71.

```

procedure P(x:integer);
var j:integer;
begin
  if (x=1) then writeln(k) else
  begin
    k:=k*x;
    P(x-1);
  end;
end;

begin
  k:=1;
  P(5);
  P(3);
end.

```

Output dari program di atas adalah:

- a. 120
720
- b. 120 6
- c. 120
6
- d. 120 720
- e. Semua jawaban di atas salah

Jawab:

e. Semua jawaban di atas salah

Pembahasan:

Error pertama kali yang akan ditemui oleh compiler adalah bahwa k tidak diketahui karena belum dideklarasikan.

Soal 72.

```

var a,b:char;
begin
  for a:='A' to 'E' do
  begin
    for b:='A' to a do
    begin
      if (a=b) then writeln(b) else write(b);
    end;
  end;
end.

```

Output dari program di atas adalah

- a. ABCDE
ABCD
ABC
AB
A
- b. A
AB
ABC
ABCD
ABCDE
- c. ABCDE
ABCDE
ABCDE
ABCDE

- ABCDE
- d. EDCBA
 - EDCB
 - EDC
 - ED
 - E
- e. A
 - BA
 - CBA
 - DCBA
 - EDCBA

Jawab:

- b. A
 - AB
 - ABC
 - ABCD
 - ABCDE

Soal 73.

```

type data=set of byte;
var setint:data;
    i:integer;
begin
    setint:=[1];
    setint:=setint+[3];
    setint:=[5];
    for i:=1 to 5 do
    begin
        if (i in setint) then continue else setint:=[i];
    end;
end.

```

Output dari program di atas adalah:

- a. [1,2,3,4,5]
- b. [1,3,5]
- c. [5]
- d. [1,3]
- e. []

Jawab:

- c. [5]

Pembahasan:

Statement di bawah ini

```

setint:=[1];
setint:=setint+[3];
setint:=[5];

```

Akan membuat setint berisi [5] saja. Pada statement berikutnya:

```

for i:=1 to 5 do
begin
    if (i in setint) then continue else setint:=[i];
end;

```

Akan membuat setint berisi nilai terakhir dari i yaitu 5.

Soal 74.

```

var i:integer;

```

```

function f(x:integer):integer;
begin
  f:=x*x+1;
end;

function g(x:integer):integer;
begin
  g:=(x-3)*x;
end;

begin
  for i:=1 to 5 do
  begin
    write(f(g(i)), ' ');
  end;
end.

```

Output dari program di atas adalah

- a. 5 4 3 2 1
- b. 1 2 3 4 5
- c. 101 17 1 5 5
- d. 5 5 1 7 17 101
- e. Semua jawaban di atas salah.

Jawab:

- e. Semua jawaban di atas salah.

Pembahasan:

$g(1) = -2$ dan $f(-2) = 5$
 $g(2) = -2$ dan $f(-2) = 5$
 $g(3) = 0$ dan $f(0) = 1$
 $g(4) = 4$ dan $f(4) = 17$
 $g(5) = 10$ dan $f(10) = 101$

Soal 75.

```

var i, j : integer;
begin
  j := 1;
  for i := 1 to 5 do
  begin
    writeln(i, ' ', j);
    j := i-1;
  end;
end.

```

Output dari program di atas adalah

- | | | | | |
|--------|--------|--------|--------|--------|
| a. 1 1 | b. 1 1 | c. 1 0 | d. 1 5 | e. 1 1 |
| 2 1 | 2 2 | 2 1 | 2 4 | 2 0 |
| 1 3 | 3 3 | 3 2 | 3 3 | 3 1 |
| 4 1 | 4 4 | 4 3 | 4 2 | 4 2 |
| 1 5 | 5 5 | 5 4 | 5 1 | 5 3 |

Jawab:

- d. 1 5
- 2 4
- 3 3
- 4 2
- 5 1

Soal 76.

```
var i, j, k: integer;
begin
  j:=1;
  for k:=1 to 3 do
  begin
    i:=k;
    i:=i+j;
    j:=i-1;
    i:=i-j;
    writeln(i, ' ', j);
  end;
end.
```

Output dari program di atas adalah

- | | | | | |
|--------|--------|--------|--------|--------|
| a. 1 1 | b. 3 1 | c. 1 3 | d. 1 1 | e. 1 1 |
| 2 2 | 2 1 | 1 2 | 1 2 | 1 2 |
| 3 3 | 1 1 | 2 3 | 2 3 | 1 3 |

Jawab:

- d. 1 1
1 2
2 3

Program untuk Soal No.77 dan 78

```
1. var s : string;
2.     i: integer;
3.     begin
4.         for i := 1 to length(s) do
5.             begin
6.                 s[i]:= s[length(s)-i+1];
7.             end;
8.     end.
```

Soal 77.

Output dari program di atas jika s='SC!P!O' adalah

- a. s=' SC!!CS'
- b. s=' SSSSSS'
- c. s=' O!PP!O'
- d. s=' CCCCCC'
- e. s=' !O!O!O'

Jawab:

- c. s=' O!PP!O'

Soal 78.

Output dari program di atas jika s='HaShMaT' dan baris ke-4 diubah menjadi "for i:=length(s) downto 1" adalah

- a. s=' TaMhMaT'
- b. s=' HaShSaH'
- c. s=' HHHHHHHH'
- d. s=' aaaaaaa'
- e. s=' aHaHaHa'

Jawab:

- b. s=' HaShSaH'

Soal 79.

```

type data=set of char;
var setchar:data;
    s:string;
    i:integer;
begin
    setchar:=[];
    readln(s);
    for i:=1 to length(s) do
    begin
        if not(s[i] in setchar) then
        begin
            setchar:=setchar+[s[i]];
            write(s[i]);
        end;
    end;
    writeln;
end.

```

Output dari program di atas jika input 'To be or Not To be that is the question' adalah

- 'To berNthaisqun.'
- 'To berNhaisqu`'
- 'to@bernhaisquN'
- 'T N.'
- 'OBERTHAISQUN'

Jawab:

- 'To berNthaisqun.'

Pembahasan:

Yang perlu diperhatikan adalah bahwa tidak ada anggota yang sama dalam sebuah set (himpunan).

Soal 80.

```

var i,j:integer;
begin
    for i:=1 to 10 do
    begin
        if (i mod 2=0) then for j:=1 to (i div 2) do
        begin
            write(j*2);
            if (j mod 2=0) then write(' ');
        end;
        writeln;
    end;
end.

```

Output dari program di atas adalah

- 2
2 4
2 4 6
2 4 6 8
2 4 6 8 10
- 2
24
24 6
24 68
24 68 10
- 2
24

- 246
- 2468
- 246810
- d. 2
- 24
- 246
- 2468
- 246810
- e. 2
- 2 4
- 2 46
- 2 46 8
- 2 46 8 10

Jawab:

- b. 2
- 24
- 24 6
- 24 68
- 24 68 10

Soal 81.

Perhatikan source code di bawah ini:

```
Begin
  Writeln(exp(5*ln(2)))
End.
```

Output yang tercetak di layar setelah eksekusi adalah:

- a. 52
- b. 5.2
- c. 0.52
- d. Terjadi compile error
- e. Tidak ada jawaban yang benar

Jawab:

- e. Tidak ada jawaban yang benar

Soal 82.

Perhatikan source code di bawah ini:

```
const
  Matrix: array [1..3] of char
    = ('A', '*', 'B');
begin
  writeln (Matrix [1]);
  write (Matrix[2]);
  writeln (Matrix[3] + Matrix [1]);
end.
```

Output yang tercetak di layar setelah eksekusi adalah:

- a. A
- *
- BA
- b. A
- *BA
- c. A
- *
- B+A
- d. terjadi compiler error
- e. tidak ada jawaban yang benar

Jawab:

- b. A
*BA

Soal 83.

```
type
  Matrix=array[1..100] of byte;

procedure Sort(var A:Matrix);
var
  i,j,k:byte;
  B:array[0..255] of byte;
begin
  fillchar(B,256,0);
  for i:=1 to 100 do begin
    inc(B[A[i]]);
  end;
  k:=1;
  for i:=0 to 255 do begin
    for j:=1 to B[i] do begin
      A[k]:=i;
      inc(k);
    end;
  end;
end;
```

Contoh prosedur di atas merupakan salah satu teknik sorting yang disebut:

- Insertion Sort
- Bubble Sort
- Merge Sort
- Quick Sort
- tidak ada jawaban yang benar

Jawab:

- e. tidak ada jawaban yang benar

Soal 84.

Perhatikan source code di bawah inii

```
var
  Mail: word;
begin
  Mail:=' TOKI';
  writein (' TOKI');
  writeln (Mail);
end.
```

Output yang tercetak di layar setelah eksekusi adalah

- TOKI
- TOKITOKI
- TOKITOKITOKI
- terjadi compiler error
- tidak ada jawaban yang benar

Jawab:

- d. terjadi compiler error

Pembahasan:

Tipe data untuk variabel Mail adalah berjenis numerik (Word) sehingga tidak dapat diberikan nilai yang berjenis teks atau string

Soal dan Pembahasan Pemrograman Pascal (Soal Sederhana)

Soal-soal di bawah ini termasuk dalam kategori mudah. Jika soal-soal ini dapat diselesaikan dengan benar, maka dapat segera berlatih TOKI Handbook Jilid II.

Pasukan (PASUKAN.PAS)

Dua kerajaan, kerajaan A dan B hendak berperang. Masing-masing mempunyai kekuatan pasukan berkuda (0..4000), pasukan panah (0..10.000) dan prajurit (0..500.000). Kekuatan pasukan berkuda adalah 3, kekuatan pasukan panah adalah 2 dan prajurit 1.

Misalkan kerajaan A mempunyai 10 pasukan berkuda, 40 pasukan panah dan 100 prajurit, maka total kekuatan pasukan kerajaan A adalah $10 \times 3 + 40 \times 2 + 100 \times 1 = 210$ point.

Buat sebuah program untuk menentukan kemungkinan pemenang dari peperangan yang terjadi apabila diketahui jumlah dari pasukan berkuda, panah dan prajurit dari masing-masing kerajaan.

Format masukan:

Kekuatan pasukan dibaca dari file PASUKAN.IN yang terdiri dari 2 baris, yaitu: baris pertama berisi kekuatan kerajaan A yang terdiri dari 3 angka yang menunjukkan jumlah pasukan berkuda yang dimiliki, jumlah pasukan panah yang dimiliki dan jumlah prajurit. Baris kedua berisi kekuatan kerajaan B yang juga terdiri dari 3 angka yang menunjukkan jumlah pasukan berkuda, panah dan prajurit

Format keluaran:

Tuliskan siapakah (A atau B) yang mungkin menjadi pemenang dalam peperangan tersebut dengan melihat kekuatan yang ada. Apabila total kekuatan sama, maka tampilkan SERI.

Contoh:

```
PASUKAN.IN  
400 2000 10000  
500 1000 15000
```

PASUKAN.OUT

B

Palindrome String (PAL1.PAS)

Palindrome string adalah sebuah string yang sebuah string yang apabila dibaca dari kiri kanan atau kanan kiri tetap sama. Misalnya KAKAK.

Buat sebuah program untuk menentukan apakah suatu string merupakan palindrome string atau bukan.

Format masukan:

Dibaca dari PAL1.IN yang berisi sebuah string

Format keluaran:

Ditulis ke PAL1.OUT apakah string pada PAL1.IN merupakan palindrome string atau bukan. Jika merupakan palindrome, tulis PALINDROME di PAL1.OUT, jika bukan tuliskan BUKAN PALINDROME.

Contoh:

```
PAL1.IN  
KASUR INI RUSAK
```

```
PAL1.OUT  
PALINDROME
```

Selisih Jam (JAM.PAS)

Sebuah kantor hendak menghitung berapa lama seorang pekerja menyelesaikan sebuah pekerjaan. Untuk itu, perusahaan tersebut memberlakukan check clock yang akan mencatat waktu awal pekerjaan dilakukan dan waktu akhir pekerjaan.

Misalnya: pekerjaan dari 10:30 – 12:45, berarti lama pengerjaan adalah 2 jam 15 menit. Buat sebuah program untuk menghitung lama pekerjaan yang dilakukan oleh seorang pekerja.

Format masukan:

Dibaca dari file JAM.IN yang terdiri dari 2 baris.

Baris pertama terdiri dari 2 bilangan yang berupa jam awal dan menit awal kerja

Baris kedua terdiri dari 2 bilangan yang menunjukkan jam akhir dan menit akhir kerja

Format keluaran:

Keluaran ditulis pada JAM.OUT yang terdiri dari satu baris yang menunjukkan lama pekerjaan yang dilakukan.

Contoh:
JAM.IN
11 30
2 35

JAM.OUT
3 jam 5 menit

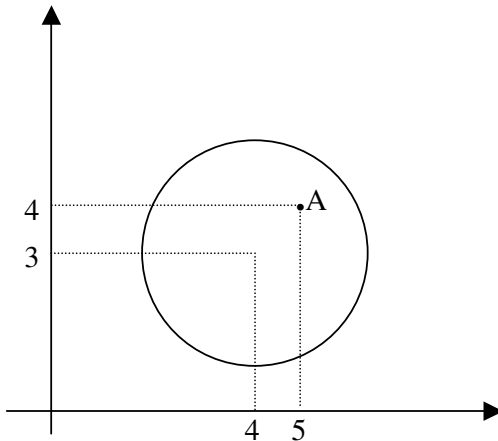
Posisi Titik pada Lingkaran (**CIRCLE.PAS**)

Buat sebuah program untuk menentukan apakah sebuah titik (x,y) berada di dalam, pada atau di luar lingkaran.

Koordinat kartesius: $-1000 \leq x \leq 1000$, $-1000 \leq y \leq 1000$

Contoh:

Titik A mempunyai koordinat (5,4) dan sebuah lingkaran berjari-jari 2 pada koordinat (4,3). Titik A terletak di dalam lingkaran.



Format masukan:

Masukan dibaca dari CIRCLE.IN yang berisi 2 baris, yaitu: Baris pertama berisi 3 angka yang masing-masing adalah titik pusat lingkaran (x dan y) diikuti dengan jari-jari. Baris kedua berisi 2 angka, yaitu koordinat titik A

Format keluaran:

File CIRCLE.OUT berisi apakah titik A berada di luar, pada atau di dalam lingkaran. Jika berada di luar lingkaran, tuliskan DI LUAR. Jika tepat bersinggungan dengan lingkaran, tuliskan BERSINGGUNGAN. Jika berada di dalam lingkaran tuliskan DI DALAM

Contoh:

CIRCLE.IN
4 3 2
5 4

CIRCLE.OUT
DI DALAM

Keterangan: Jika dibutuhkan bilangan pecahan, maka tipe bilangan bulat (seperti byte, integer, word, shortint,

longint) tidak lagi dapat digunakan. Untuk bilangan pecahan, gunakan tipe data real.

Mencari akar, gunakan fungsi SQRT, misalnya $a := \text{SQRT}(n)$, maka a adalah akar kuadrat dari n, dimana a harus bertipe real. Pelajari lebih lanjut tentang bilangan real di buku!

Palindrome Angka (**PAL2.PAS**)

Sama halnya seperti palindrome string, palindrome angka adalah angka yang mempunyai nilai yang sama jika dibaca dari kiri ke kanan atau kanan ke kiri. Misalnya 5, 22, 737, 252, 161 dan sebagainya.

Buat sebuah program untuk menghasilkan semua palindrome angka dari A sampai dengan B di mana $1 \leq A \leq 1000$ dan $1 \leq B \leq 1000$.

Format masukan:

Masukan dibaca dari file PAL2.IN yang berisi 2 baris, yaitu A dan B.

Format keluaran:

Keluaran ditulis pada PAL2.OUT yang berisi semua palindrome angka dari A sampai dengan B.

Contoh:

PAL2.IN
50
120

PAL2.OUT
55
66
77
88
99
101
111

Menghitung Deret (**DERET.PAS**)

Buat sebuah program untuk menghitung hasil akhir dari rumus berikut:

$$Y = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} \dots - \frac{1}{1999} + \frac{1}{2001}$$

Format masukan: -

Format keluaran:

Tulis hasil Y hingga ketelitian 5 angka dibelakang koma pada file DERET.OUT

Keterangan:

Tipe Bilangan bulat (Byte, Shortint, Longint, Integer dan Word) tidak dapat digunakan untuk menghitung pecahan. Gunakan tipe data REAL untuk menghitung pecahan.
Pelajari lebih dulu tentang cara menulis ke standard output untuk variabel bertipe real.

Plat Nomor Kendaraan (PLAT.PAS)

Tingkat kepadatan di kota Jakarta sudah sangat tinggi sehingga setiap hari terjadi kemacetan pada jalan-jalan protokol. Untuk menghindari kemacetan, pihak pemda memberlakukan sebuah aturan baru, yaitu dengan menyeleksi plat nomor kendaraan yang boleh masuk pada sebuah jalan protokol. Syarat untuk dapat masuk pada jalan protokol adalah yang jumlah digit plat nomornya adalah nomor ganjil. Digit suatu plat nomor dapat berkisar 1..4 digit. Misalnya: plat nomor N 4237 CT, jumlah digitnya adalah $4 + 2 + 3 + 7 = 16 = 1 + 6 = 7$. Karena jumlah digitnya adalah 7, maka mobil tersebut diperbolehkan masuk.

Buat sebuah program untuk menentukan apakah sebuah mobil diperbolehkan masuk di jalan protokol atau tidak.

Format masukan:

Dibaca dari file PLAT.IN yang terdiri dari 1 baris yang berisi digit dari plat nomor kendaraan.

Format keluaran:

Keluaran ditulis pada PLAT.OUT yang berisi MASUK atau KELUAR sesuai dengan syarat yang ada:

Misal:

PLAT.IN	PLAT.OUT
247	KELUAR

*) Penjelasan keluaran: $2 + 4 + 7 = 13 = 1 + 3 = 4 \rightarrow$ genap, tidak boleh masuk

Persamaan Linier 2 variabel (LINIER.PAS)

Untuk menyelesaikan 2 persamaan linier, dapat dilakukan dengan 3 cara, yaitu cara substitusi, eliminasi dan matriks.

Misalnya:

$$3x + 4y = 18$$

$$x - 3y = -7$$

Penyelesaian dengan cara eliminasi:

$$\begin{array}{r|l} 3x + 4y = 18 & | \times 1 | \quad 3x + 4y = 18 \\ x - 3y = -7 & | \times 3 | \quad 3x - 9y = -21 \\ \hline & 13y = 39 \quad \rightarrow y = 3 \end{array}$$

$$3x + 4 \cdot 3 = 18$$

$$x = (18 - 12) / 3$$

$$x = 2$$

Buat sebuah program untuk menghitung nilai x dan y dari 2 buah persamaan linier.

Format masukan:

Dibaca dari file LINIER.IN yang terdiri dari 2 baris.

Baris pertama berisi konstanta persamaan linier pertama, yaitu a, b dan c dari sebuah persamaan $ax + by = c$

Baris kedua berisi konstanta persamaan linier kedua, yaitu d, e dan f dari sebuah persamaan $dx + ey = f$

Format keluaran:

Keluaran ditulis pada file LINIER.OUT yang terdiri dari 2 baris, yaitu nilai x dan y

Contoh:

LINIER.IN	LINIER.OUT
1 -3 7	x = 1
4 1 6	y = -2

Kuadrat (KUADRAT.PAS)

Buat sebuah program untuk menampilkan semua bilangan dari 10 sampai dengan 100 yang memenuhi syarat:

$$55^2 = 3025$$

$$30 + 25 = 55$$

Format masukan: -

Format keluaran:

Semua bilangan yang memenuhi syarat di atas, ditulis pada KUADRAT.OUT

Minimum dan Maksimum (MINMAKS.PAS)

Dari N buah data ($1 \leq n \leq 500$), buatlah program untuk menentukan nilai maksimal dan minimal dari data tersebut.

Format masukan:

Data dibaca dari MINMAKS.IN.

Baris pertama berisi N, yang menunjukkan banyak data.

Baris ke-2 sampai ke N+1 adalah data yang hendak dicari nilai maksimal dan minimal.

Format keluaran:

Keluaran berupa file MINMAKS.OUT yang berisi nilai maksimal dan minimal dari data pada MINMAKS.IN. Nilai maksimal dan minimal dipisahkan menjadi 2 baris.

Contoh:

MINMAKS.IN	MINMAKS.OUT
10	71
4	4
33	
71	
8	
10	
60	
25	
12	

Catatan: Pelajari cara mencari nilai minimal dan maksimal pada buku BASIC kelas I.

PENGURUTAN DATA (SORT.PAS)

Buat sebuah program untuk mengurutkan sekumpulan data dari besar ke kecil.

Format masukan:

Data dibaca dari SORT.IN yang berisi:

Baris pertama : N, banyak data ($1 \leq N \leq 1000$)

Baris kedua - N+1 : data yang hendak diurutkan.

Format keluaran:

Hasil pengurutan ditulis pada file SORT.OUT yang berisi data yang telah terurut dari besar ke kecil.

Contoh:

SORT.IN	SORT.OUT
5	60
10	40
40	30
20	20
60	10
30	

MEDIAN (MEDIAN.PAS)

(Pelajari tentang Array terlebih dulu!)

Median adalah nilai tengah dari sekumpulan data yang terurut. Buat sebuah program untuk mencari nilai median dari N data ($1 \leq N \leq 1000$) yang tidak terurut.

Contoh, untuk data: 5 8 3 -2 -10, mediannya adalah:

-10 -2 3 5 8, karena jumlah data ganjil, maka median adalah 3

Namun, jika jumlah data genap, median adalah rata-rata dari 2 data yang berada di tengah. Misalnya:

-10 -2 3 4, maka mediannya adalah $(-2+3)/2 = -0.5$

Format masukan:

Data dibaca dari file MEDIAN.IN yang berisi:

Baris pertama: N, banyak data ($1 \leq N \leq 1000$)

Baris kedua-N+1: data yang hendak dicari mediannya.

Format keluaran:

Output ditulis pada file MEDIAN.OUT yang berisi 1 baris, yaitu median dari data tersebut.

Contoh:

MEDIAN.IN	MEDIAN.OUT
5	3
5	

8
3
-2
-10

PEMBAHASAN / SOLUSI

Source code di bawah ini merupakan solusi dari tiap-tiap soal. Pembahasan diletakkan dalam baris-baris komentar program.

```
{ Nama Program : PASUKAN.PAS }

var
  fi,fo:text;
  kuda_a, panah_a, prajurit_a, total_power_a : longint;
  kuda_b, panah_b, prajurit_b, total_power_b : longint;
begin
  assign(fi,'pasukan.in');
  reset(fi);

  assign(fo,'pasukan.out');
  rewrite(fo);

  readln(fi,kuda_a,panah_a,prajurit_a);
  readln(fi,kuda_b,panah_b,prajurit_b);

  total_power_a:=kuda_a*3+panah_a*2+prajurit_a; {hitung kekuatan A}
  total_power_b:=kuda_b*3+panah_b*2+prajurit_b; {hitung kekuatan B}

  {tulis ke file output}
  if total_power_a>total_power_b then write(fo,'A') else
    if total_power_a=total_power_b then write(fo,'SERI') else write(fo,'B');

  close(fo); {close file output}
end.
```

```
{ Nama Program : PAL1.PAS }

var
  fi,fo:text;
  i:byte;
  s,s2:string;
begin
  assign(fi,'pall.in');
  reset(fi);

  assign(fo,'pall.out');
  rewrite(fo);

  readln(fi,s);

  s2:=''; {s2 buat menampung s yang dibalik}
  for i:=length(s) downto 1 do
    s2:=s2+s[i];

  if s=s2 then
    write(fo,'PALINDROME')
  else
    write(fo,'BUKAN PALINDROME');

  close(fo); {close file output}
end.
```

```
{ Nama Program : JAM.PAS }

var
  fi,fo:text;
  start_hour, end_hour,
```

```

    start_minute, end_minute,
    total_minute, total_hour : byte;
begin
    assign(fi,'jam.in');
    reset(fi);

    assign(fo,'jam.out');
    rewrite(fo);

    readln(fi,start_hour,start_minute);
    readln(fi,end_hour,end_minute);

    {hitung selisih menit terlebih dulu}

    if start_minute>end_minute then
    begin
        inc(end_minute,60); {end_minute + 60}
        dec(end_hour); {karena menit akhir ditambah
            60, maka, jamnya harus dikurangi}
    end;

    total_minute:=end_minute-start_minute;

    {ada kemungkinan jam akhir melewati jam 12, sehingga end_hour-nya
    lebih kecil. Kalau end_hour lebih kecil, tambah dengan 12}
    if start_hour>end_hour then inc(end_hour,12);

    {menghitung selisih jam}
    total_hour:=end_hour-start_hour;

    write(fo,total_hour,' jam ',total_minute,' menit');

    close(fo); {close file output}
end.

```

```

{ Nama Program : PAL2.PAS }

var
    fi,fo:text;
    i:word; {word: 0..65.535}
    awal,akhir:word;
begin
    assign(fi,'pal2.in');
    reset(fi);

    assign(fo,'pal2.out');
    rewrite(fo);

    readln(fi,awal);
    readln(fi,akhir);

    for i:=awal to akhir do
    begin
        {jika 1 digit, maka pasti palindrome}
        if i<=9 then writeln(fo,i);

        {cek palindrome puluhan}
        if i<=99 then
            if i mod 10 = i div 10 then
                writeln(fo,i);

        {cek palindrome ratusan}
        nilai ratusan merupakan palindrome apabila
        digit pertama dengan digit ketiga sama.
        Misalnya 646, digit pertama didapat dengan
        membagi 646 dengan 100 = 6 (karena bilangan
        bulat, maka hasilnya pasti bulat), sedangkan
        digit satuan (digit ketiga) didapat dengan
        operasi mod 10. 646 mod 10 = 6.)
        if i>=100 then
            if i mod 10 = i div 100 then writeln(fo,i);
    end;
end;

```

```
close(fo); {close file output}
end.
```

```
{ Nama Program : DERET.PAS }

var
  r:real;
  i:integer;
  plus:boolean;
begin
  r:=1;
  i:=1;
  plus:=true;
  while i<2001 do
  begin
    if plus then r:=r+1/i else r:=r-1/i;
    inc(i,2);
    plus:=not plus;
  end;
  assign(output,'deret.out');
  rewrite(output);
  write(r:0:5);
  close(output);
end.
```

```
{ Nama Program : PLAT.PAS }

var
  fi,fo:text;
  n:word;
  ribuan, ratusan, puluhan, satuan:byte;
  total:byte;
begin

  assign(fi,'plat.in');
  reset(fi);

  assign(fo,'plat.out');
  rewrite(fo);

  readln(fi,n);

  {cara untuk mendapatkan tiap digit}
  ribuan:=n div 1000;      {digit ribuan}
  ratusan:=(n div 100) mod 10; {digit ratusan}
  puluhan:=(n div 10) mod 10; {digit puluhan}
  satuan:=n mod 10;      {digit satuan}

  {maksimum untuk total adalah apabila plat
  nomer 9999 = 36 = 3+6 = 9}
  total:=ribuan+ratusan+puluhan+satuan;

  {ada kemungkinan setelah setiap digitnya ditambah,
  masih terdiri dari 2 digit lagi, misalnya 2+4+3+7
  =16. Maka tiap digitnya harus ditambahkan lagi, 1+6=7}
  if total>=10 then
  begin
    puluhan:=total div 10;
    satuan:=total mod 10;
    total:=puluhan+satuan;
  end;

  if total mod 2=0 then write(fo,'KELUAR') else write(fo,'MASUK');

  close(fo); {close file output}
end.
```
