Seri Buku Persiapan Seleksi Tim Olimpiade Komputer Indonesia

Buku Untuk Guru

Aspek Pedagogi
Pengajaran Pemrograman Pertama

Menggunakan Bahasa Pascal

Disusun Oleh :

Tim Pembina TOKI

Judul Buku : Aspek Pedagogi Pengajaran Pemrograman Pertama (Menggunakan Bahasa Pascal)
Penyusun : Tim Pembina TOKI
Kontributor : Dr. Ir. M.M. Inggriani Liem
Penyunting : TOKI
Disain Cover : TOKI
Diterbitkan oleh : Bagian Proyek Pengembangan Wawasan Keilmuan
Direktorat Pendidikan Menengah Umum,
Direktorat Jenderal Pendidikan Dasar dan Menengah,
Departemen Pendidikan Nasional RI
Cetakan Pertama : 2004

Seri Buku Periapan Seleksi Olimpiade Komputer Indonesia

Buku 1 Untuk Siswa : Referensi Pemrograman Bahasa Pascal (*menggunakan free pascal ver. 1.0.10*)

Buku 2 Untuk Siswa dan Guru : Konsep Dasar Pemrograman (*dilengkapi dengan kisi-kisi seleksi, contoh soal, pembahasan dan solusi*)

Buku 3 Untuk Guru : Aspek Pedagogi Pengajaran Pemrograman Pertama *(Menggunakan Bahasa Pascal*)

Kata Pengantar

Sejak keikutsertaan Indonesia dalam ajang International Olympiad in Informatics pada tahun 1997, prestasi siswa-siswa dalam ajang tersebut cukup membanggakan. Hingga tahun 2004 ini (keikutsertaan yang ke 9) secara total Tim Olimpiade Komputer Indonesia telah mengumpulkan 1 Medali Emas, 6 Medali Perak dan 7 Medali Perunggu.

Sejauh ini Tim Olimpiade Komputer Indonesia yang dikirim ke ajang IOI masih didominasi oleh siswa-siswa di wilayah Jawa, terutama DKI Jakarta, Jawa Barat dan Jawa Timur, hal ini menunjukkan bahwa masih terdapat ketimpangan pengetahuan dan pembinaan di bidang Komputer/Informatika terhadap siswa-siswa di daerah lain. Hal ini dapat terjadi salah satunya adalah karena keterbatasan ketersediaan materi pelajaran computer, khususnya yang mengarah kepada materi-materi yang dilombakan dalam ajang Olimpiade Komputer Indonesia.

Buku ini diterbitkan dalam beberapa seri untuk siswa maupun untuk guru dimaksudkan agar dapat menjadi bahan pelajaran bagi siswa dan panduan pengajaran bagi guru yang memadai untuk mempersiapkan siswa menghadapi rangkaian seleksi Olimpiade Komputer Indonesia. Dengan harapan bahwa kesempatan dan peluang bagi siswa-siswa di seluruh Indonesia menjadi lebih terbuka.

Terima Kasih kepada semua pihak yang telah memberikan kontribusinya sehingga penerbitan buku ini dapat terwujud. Besar harapan kami buku ini dapat bermanfaat bagi semua pihak. Kritik dan saran yang membangun sangat kami harapkan.

Jakarta, Oktober 2004

Tim Olimpiade Komputer Indonesia

Pengantar Penulis

Pengajaran yang pertama merupakan pengajaran yang penting, karena akan merupakan landasan bagi pengajaran selanjutnya. Dalam pengajaran pemrograman pertama, konsep dan elemen pemrograman dalam paradigma tersebut harus sudah dicakup seluruhnya, sehingga murid mempunyai pandangan yang integral.

Pengajaran pemrograman akan sangat abstrak dan sulit ditangkap jika murid hanya dihadapkan pada konsep-konsep tanpa pernah "bermain" dengan komputer dan pemroses bahasanya.

Dalam pengajaran pemrograman prosedural, dipandang perlu untuk "membumikan" konsep-konsep paradigma pemrograman prosedural dalam suatu bahasa yang mampu dieksekusi oleh mesin. Bahasa apapun yang dipilih, elemen pemrograman prosedural harus diterjemahkan dalam bahasa tersebut, dan pengajaran bukan hanya diorientasikan ke sintaks, melainkan ke semantik dari elemen pemrograman tersebut. Pada pengajaran yang berikutnya, bahasa pertama yang dipilih akan merupakan "meta bahasa" yang akan mempercepat proses pengajaran.

Oleh karena itu, walaupun diajarkan bahasa algoritmik (yang tidak mempunyai pemroses bahasa, karena sangat konseptual), sangat disarankan untuk memilih salah satu bahasa yang dipakai oleh murid untuk mencoba mengeksekusi program yang ditulisnya. Ini akan memperingan tugas guru dalam pemeriksaan tugas-tugas kecil dan sederhana.

Seperti halnya ketika belajar berenang murid harus berenang di kolam atau bahkan di sungai, atau pada pelajaran bermain piano murid tidak hanya belajar teori musik melainkan harus memainkan piano, maka dalam pengajaran pemrograman hal yang setara (yaitu "praktek") berlaku pula. Murid selayaknya belajar dan mencoba dengan komputer. Setiap guru pengajar pemrograman selayaknya juga memang pernah memprogram. Akan sangat lebih baik jika pernah menghasilkan program yang memang pernah dipakai di dunia operasional (dalam konsteks sekecil apapun). Tidak semua programmer yang "baik" akan secara otomatis menjadi "guru" yang baik. Oleh karena itu, diperlukan suatu pedagogi pemrograman, yang seharusnya dikuasai oleh para guru sebelum mengajar.

Di Departemen Teknik Informatika ITB, pengajaran pemrograman dianggap sebagai mata ajar yang penting untuk menunjang mata pelajaran yang lain. Oleh karena itu, pengajarannya cukup mendapat perhatian. Bahan ajar mulai dari sederhana sampai kompleks, mencakup berbagai paradigma (prosedural, fungsional, deklaratif, berorientasi objek) telah dan masih terus menerus dikembangkan secara bertahap sejak Departemen Teknik Informatika didirikan pada tahun 1981. Pengajaran pemrograman yang dibedakan dari pengajaran bahasa pemrograman telah menghasilkan beberapa "kit" pengajaran saat ini sudah mencakup berbagai paradigma. Buku kecil ini adalah salah satu diantaranya.

Buku ini merupakan “Buku Panduan Bagi Guru”, sebagai pelengkap dari "Program Kecil dalam Bahasa Pascal", yang dipakai sebagai pengajaran pemrograman pertama dalam bahasa Pascal, yang dapat diselesaikan kira-kira 1 minggu penuh (kuliah dan praktikum). Buku Program Kecil dan Panduan ini seharusnya masih harus dilengkapi oleh masing-masing guru pengajar dengan bank contoh & latihan soal (yang dibuat oleh Guru sendiri), karena pada hakekatnya, pemrograman bertumpu pada pembahasan sejumlah contoh tipikal yang dikembangkan oleh masing-masing pengajar sesuai dengan konteks.

Sebagai penutup, buku edisi khusus untuk TOKI ini mengandung bahan-bahan yang diperlukan dalam pembinaan murid untuk mengikuti lomba (seleksi) TOKI dan IOI.

Bandung, Oktober 2004

Penulis

Daftar Isi

Daftar Program Kecil

DESKRIPSI PROGRAM KECIL PASCAL

KELOMPOK	NAMA	DESKRIPSI
Struktur Program pascal	hello.pas	Menuliskan hello ke layar (Pascal standart)
	hellodos.pas	Menuliskan hello ke layar (Lingkungan DOS)
Assignment	asign.pas	Deklarasi integer, Asignment
	asignl.pas	Asignment
Type dasar	Tdasar.pas	Deklarasi dan penulisan nilai type dasar
	oprator.pas	Asignment dan ekspresi sederhana
String	strmanip.pas	Manipulasi string sederhana
	bacastr.pas	Pembacaan string
Konstanta	konstant.pas	Deklarasi konstanta, ekspresi
	konstan2.pas	Deklarasi konstanta, ekspresi
Type SET	Himpunan.pas	Deklarasi dan pemanfaatan SET dalam Pascal
Type bentukan	type.pas	Deklarasi type bentukan
Input/Output	baca.pas	Pembacaan data
Analisa Kasus	if1.pas	Kasus: satu kasus
	if2.pas	Kasus: dua kasus komplementer
	if3.pas	Kasus: 3 kasus
	kasus.pas	Kasus: banyak kasus
	wujudair.pas	Terjemahan wujud air dari diktat
	max2.pas	Maksismum dua nilai
Subprogram	subprg.pas	Prosedur dan fungsi : pendefinisian dan pemanggilan
	lingkup.pas	Scope & Life time dari variabel
	funcrec.pas	Realisasi Fungsi yang hrs menghasilkan type record
Pengulangan	prifor.pas	Pengulangan FOR
	priw.pas	Pengulangan WHILE

KELOMPOK	NAMA	DESKRIPSI
	pririp.pas	Pengulangan REPEAT
	printer.pas	Pengulangan ITERATE
	kasusrep.pas	Kasus digabung dengan REPEAT
Tabel	tabel.pas	Deklarasi tabel integer, pengisian dan penulisan
	tabstru.pas	Deklarasi tabel dengan elemen type bentukan
	tab2dim.pas	Tabel dua dimensi
	tab3dim.pas	Tabel tiga dimensi
RECORD VARIAN	recvar.pas	Deklarasi, mengacu field record varian
	recvarx.pas	Record varian dengan komponen type bentukan
Pointer	ptint.pas	Pointer ke integer
	ptab.pas	Pointer ke tabel integer
	ptrec.pas	Pointer ke record
	list.pas	Deklarasi list linier sederhana
File Eksternal	Bacatxt.pas	Membaca text file da nmenuliskan isinya ke layar. File diakhiri titik dan EOF
	Rekam.txt	Rekam file teks
	Rwint.pas	Merekam dan menuliskan isi File of integer
	frec.pas	Merekam dan menulis isi File of integer
Unit dalam Turbo Pascal dan FreePascal	upoint.pas mpoint.pas	Unit untuk manipulasi type point, dan tabel yang elemennya point

Pendahuluan

Bahasa Pascal seringkali dipakai sebagai bahasa yang diajarkan pertama kali, karena sederhana, terstruktur dan tidak kompleks. Bahasa Pascal ada banyak versi implementasinya. Bisanya, karena ketersediaannya di lingkungan PC, maka Turbo Pascal banyak dipakai sebagai kompilator untuk praktikum.

Sebagai informasi, di Departemen Teknik Informatika ITB, yang dipakai adalah FreePascal yaitu salah satu implementasi kompilator bahasa pascal yang berjalan di banyak sistem operasi di antaranya yaitu DOS, Windows, dan Linux, dan merupakan Free Software. FreePascal hampir "kompatibel" dengan Turbo Pascal.

Modul pelatihan ini membahas tentang pemilihan suatu bahasa pertama sebagai meta bahasa, dan bahasa Pascal & pedagoginya, jika diajarkan sebagai bahasa pertama.

Materi untuk modul ini ada dua bagian :

[1] Liem, Inggriani : “Diktat kuliah : Program Kecil dalam bahasa Pascal”, Departemen Teknik Informatika ITB, edisi terbaru, tahun 2004
[2] Dokumen ini, yaitu catatan pedagogi yang menyertai diktat program kecil tersebut.

Urutan penyajian dalam dokumen ini disesuaikan dengan urutan pengajarannya (walaupun Guru dapat mengubah sesuai dengan kebutuhan)

Sedangkan sebagai referensi utama, buku yang diterbitkan oleh penulis bahasa Pascal merupakan bacaan wajib bagi para pengajar :

[3] Jensen Kathleen and Wirth Niklaus : “Pascal User Manual and Report, second edition”, Springer Verlag, 1975

Sebenarnya dua dokumen tersebut belum lengkap sebagai bahan pegangan bagi pengajar, karena sebaiknya disertai dengan soal-soal latihan yang sesuai topiknya dengan program kecil. Latihan-latihan soal tersebut justru seharusnya dikembangkan oleh pengajar secara "bebas", dalam rangka personalisasi kuliahnya. Soal-soal harus disesuaikan dengan pengetahuan dasar publiknya; bahkan dapat disesuaikan untuk publik tertentu. Misalnya, pengajaran untuk:

? Program Studi Sipil, maka soal-soalnya diarahkan pada program-program yang melakukan komputasi yang diperlukan bagi profesional di bidang sipil.
? Program Studi Elektro, maka soal-soalnya seharusnya berkaitan dengan persoalan yang seharusnya diselesaikan seseorang yang bekerja dalam bidang elektro.
? Program Studi Akuntansi, maka soal-soalnya seharusnya berkaitan dengan pembukuan, accounting.
? Program Studi Matematika, maka soa-soal dapat disesuaikan untuk yang berarah lebih matematis.
? Tim Olimpiade Komputer Indonesia, soal-soalnya yang terarah ke soal-soal yang dilombakan (*problem solving*).
? dsb

Semua "contoh" dengan nama program yang ditulis dalam dokumen ini merupakan nama program kecil dalam diktat [1].

Peran Guru Pengajar

Guru pengajar diharapkan bukan hanya menjadi "pengguna" dari dua modul yang telah dikembangkan ini, melainkan "melengkapi" modul ini dengan mengembangkan contoh-contoh soal yang akan dipakai menjadi bahan latihan untuk dibahas di kelas maupun untuk bahan praktikum.

Konstruktor (Elemen) Pemrograman Prosedural

Semua konstruktor / elemen pemrograman prosedural yang harus dicakup dalam pengajaran pertama, tanpa memandang implementasinya ke dalam bahasa tertentu. Yang dimaksud dengan konstruktor / Elemen tersebut adalah:

1. Program utama : definisi, bentuk, contoh.
2. Type: definisi, type dasar dan type bentukan
3. Konstanta : definisi, pemakaian
4. Variabel (nama informasi) : definisi, pemakaian
   - 4.1. Deklarasi
   - 4.2. Alokasi (hanya untuk pointer)
   - 4.3. Inisialisasi (hanya ada di beberapa bahasa pemrograman)
5. Ekspresi, operator, operan
6. Struktur Data : definisi, deklarasi, manipulasi
   - 6.1. Type bentukan (record)
   - 6.2. Array
   - 6.3. Set / enumerasi
   - 6.4. pointer
   - 6.5. Sub type (jika ada)
   - 6.6. Pendefinisan struktur data (type) baru
7. Instruksi :
   - 7.1. Baca / tulis
   - 7.2. Assignment

- 7.3. Sequence
- 7.4. Analisa kasus (Conditional statement) : if.. then ….; if … then … else..; case…
- 7.5. Pengulangan (Loop) : while, repeat, for

8. Program moduler : definisi, spesifikasi, parameter, parameter passing
   - 8.1. Fungsi
   - 8.2. Prosedur
   - 8.3. Scope & life time dari type, variabel, konstanta dengan adanya fungsi / prosedur

9. File eksternal : definisi, deklarasi, manipulasi
   - 9.1. Text file
   - 9.2. File lain : file of integer / real; file of <type bentukan>

10. Rekurens : definisi, implementasi
    - 10.1. Program, prosedur, fungsi rekursif
    - 10.2. Struktur Data rekursif

Pengajaran Pertama

Pada pengajaran pemrograman yang diberikan pertamakali, harus diberikan pemrograman yang paling mendasar. Pemrograman ini akan menjadi dasar dari pemrograman selanjutnya. Karena itu, pada pengajaran Pemrograman pertama sangat perlu diperhatikan::

? pembentukan pola berpikir sistematis, lebih baik (jika ada) sesuai "standard"
? harus mencakup hal yang **esensial**, walaupun sedikit**. Bahan yang diberikan ke murid harus sesedikit mungkin** dan hanya menyangkut yang esensial ini. **Latihan harus banyak**, dibuat oleh murid sendiri dan tidak diberikan sebagai bahan karena justru akan mengaburkan hal yang esensial.
? pengertian akan spesifikasi versus koding. Pelajaran ini hanya mencakup tahap koding, dan sama sekali tidak mencakup desain / pembuatan spesifikasi program.
? walaupun tidak dikatakan secara eksplisit, bahasa pertama ini akan menjadi “meta bahasa” yang akan dipakai pada pengajaran bahasa pemrograman yang berikutnya
? aspek eksekusi (hasil program, trace nilai) sebaiknya tidak pernah diberikan di kelas secara rinci, kecuali jika kuliah memakai komputer. Aspek ini sebaiknya dipisahkan dan diberikan di lab. Bahkan sangat disarankan untuk dilakukan secara mandiri oleh murid.

Biasanya, bahasa Pascal dipakai sebagai bahasa pertama dalam pengajaran pemrograman. Tulisan berikut ini adalah catatan jika Pascal diberikan sebagai bahasa pertama.

Bahan Pascal standard yang tidak dicakup dalam modul dasar:

? record variant
? pointer

Sebagai contoh, untuk pengajaran awal yang "ringan", Bahan ini akan diajarkan ketika mempelajari struktur data, dengan bahasa pemrogaman kedua yang diajarkan, dengan tetap memakai padanan program kecil di Pascal.

Jika memakai Turbo Pascal, Bahan Pascal yang **spesifik Turbo Pascal** yang harus dicakup:

? string dan type khusus lain
? library grafik dalam Turbo Pascal
? implementasi dalam unit

Bahan ini dibahas tetapi jangan terlalu banyak memakan waktu. sebaiknya Grafik dibahas bersamaan dengan unit. Pakailah grafik sebagai sebuah unit program.

String sebenarnya tidak masuk dalam tipe dasar Pascal standar, namun secara de fakto didukung oleh semua kompiler dan sangat diperlukan dalam semua hal, jadi tipe ini harus diajarkan.

Prinsip pengajaran :

? Lewat CONTOH, tidak lewat sintaks formal.
? Contoh harus mewakili persoalan dan merupakan contoh yang baik
? Jika ada satu seri contoh, maka penyampaiannya harus dimulai dari yang sederhana dan berkembang menuju ke yang kompleks.
? Contoh tidak boleh membingungkan, dan harus dikuasai benar oleh guru. Walaupun contoh kecil, guru harus memahami "makna" program yang dipakai sebagai contoh, dan aspek pedagogi apa yang ada dibalik contoh tersebut.
? Variasi contoh diberikan sebagai latihan, bukan sebagai bahan ajar utama.

Aspek yang ditekankan pada program yang dihasilkan murid:

? sedapat mungkin mengikuti sintaks Pascal standard (bukan Turbo Pascal)
? mengikuti pola pengetikan standard yang diberikan di kelas
? aspek ***readability***, ***maintainabilty*** dan ***robustness*** program sangat penting.
? aspek moduler penting: sepotong kecil program (atau bahkan prosedur, fungsi) harus merupakan penyelesaian dari suatu persoalan secara atomik.

Semua contoh program harus diperbaiki **cara penulisannya sesuai "standard" sebagai berikut (sekaligus menjadi latihan program reading)**:

? Penulisan komentar per baris
? Komentar memakai sintaks pascal standar : (* *)
? penulisan ekspresi harus dengan tanda kurung

? penulisan antara satu operan dengan operan lain diberi "space" untuk menambah *readability* program. Contoh : x:=x+1; harus dikoreksi menjadi x := x + 1;

? Kalimat Pascal selalu diakhiri dengan ";". Tidak ada pengecualian, walaupun Pascal memperbolehkan kalimat terakhir sebelum "end" tidak diakhiri dengan ";"

? **Aksi** adalah sederetan kalimat Pascal, dan harus dituliskan di antara "**begin**" **….** "**end;**". Semua aksi harus dituliskan di antara "**begin**" **….** "**end;**" walaupun dalam bahasa Pascal, jika blok mengandung satu kalimat saja tidak perlu menuliskan diantara "begin" …. "end;". Jelaskanlah pada murid bahwa ini aspek standard dan membuat program mudah diubah (aspek ***maintainability***)

Batasan umum :

? Semua contoh yang diberikan sebaiknya mewakili situasi dunia nyata dengan benar (nantinya diperlukan dalam pemrograman kasus nyata). Jadi contoh harus dipilih dengan hati-hati, tidak boleh asal pilih (sembarangan).

? **pengajaran minimal mencakup**: **pengertian akan sintaks dengan pola penulisan standard yang disepakati pada konteks pengajaran** (cara penulisan yang benar menurut definisi bahasa dan juga ***"good practice in programming"***) dan **mekanisme eksekusi** dari sintaks yang benar. Murid jangan dibingungkan untuk mengerti semnatik (apalagi dengan berbagai variasinya yang kadang sangat subtil)

? **Tidak perlu semua fitur bahasa Pascal diajarkan**. Ini terlalu banyak untuk pemula. Sebaiknya yang diajarkan hanya seperlunya yang nantinya akan diperlukan untuk merealisasikan konsep algoritmik. Jadi murid harus diajarkan pola dasar secukupnya, dan dilatih untuk membuat sendiri beberapa latihan dengan memakai pola yang ada. Memberikan terlalu banyak variasi akan membingungkan murid.

? **Hal yang menyangkut spesifikasi/design** tidak dicakup dalam kuliah Pascal yang dasar ini.

? **Contoh yang jelek/salah** hanya boleh diberikan sekali saja. Selanjutnya contoh / prinsip yang jelek dibuang, dan kita harus memakai contoh yang bagus. Contoh: pembacaan nilai N (pembatas) negatif yang dapat mengakibatkan program "abort" di beberapa pengulangan. Selanjutnya, setelah pengulangan dikenal, maka nilai yang dibaca harus diulang sampai benar dengan skema yang baik (dalam hal ini *repeat … until* benar), tapi hal ini tidak dibutuhkan jika batasan input soal sudah diberikan sebagai spefisikasi.

? **Semua kasus kosong, setelah konsep itu dikenal (lewat contoh!)** sebaiknya ditangani. Jadi misalnya EOF pada file: pakailah skema WHILE atau baca pertama, kemudian test dan REPEAT

? ***Exercise* gabungan, melakukan komposisi dari dua atau lebih program kecil** (semacam menggabung prosedur dalam pemrosesan file), harus diajarkan lewat contoh dengan sangat hati-hati. Kadang-kadang,

penggabungan dua konsep tanpa memperhatikan totalitas, akan menghancurkan konsep gabungan secara total.

Arahan untuk pengajaran pemrograman kedua dan selanjutnya

Pengajaran pemrograman yang selanjutnya (intermediate), selain memakai instruksi dan konstruktor dasar yang sudah diajarkan pada pelajaran pemrograman pertama, maka aspek struktur data mulai diajarkan, sambil memperkenalkan bahasa lain. Pengajaran bahasa yang kedua dianjurkan untuk memakai Program kecil yang "sama" dengan yang pernah dipelajari pada bahasa sebelumnya, sehingga pengajaran berkesinambungan (pendalaman konsep), dan sekaligus mendapatkan padanan dalam bahasa yang baru. Setelah instruksi yang merupakan padanan diberikan, selanjutnya murid harus belajar memakai hal-hal spesifik dan "style" pemrograman sesuai dengan bahasa yang baru. Ini diajarkan secara bertahap, melalui contoh-contoh. Pelajaran pemrograman yang kedua harus mengandung "studi kasus", yaitu contoh-contoh pemakaian konsep yang diajarkan, dan bukan sekedar konstruktor/elemen dasar.

Di Departemen Teknik Informatika ITB, telah tersedia satu seri program kecil yang dipakai untuk maksud ini.

Setelah murid mengalami pemrograman kedua (intermediate), maka murid dapat memasuki tingkatan yang lebih tinggi, skala persoalan yang lebih kompleks dan besar, dengan tetap memakai bahasa atau bahkan paradigma yang sama, atau mungkin juga di-diversifikasi. Biasanya sudah spesifik terhadap bidang tertentu dan memakai bahasa yang banyak dipakai dalam bidang tersebut. Teknik pemrograman juga dapat bertambah dengan menggunakan fitur yang advanced. Dapat pula dirancang, jika beranggapan bahwa aspek prosedural dicakup dan dipakai dalam pemrograman berorientasi objek, maka dalam pemrograman berorientasi objek (OOP) hanya diajarkan konsep-konsep yang erat kaitannya dengan OOP dan bagian prosedural sudah tidak diajarkan lagi.

Program Pascal Sederhana

Program sangat sederhana untuk memprint "hello" ke layar merupakan program wajib untuk memulai memprogram dalam bahasa apapun. Contoh wajib :

? hello.pas

? hellodos.pas : untuk menunjukkan spesifiknya Turbo Pascal (jangan bahas terlalu banyak tentang Uses CRT). Free Pascal memiliki unit CRT yang kompatibel dengan Turbo Pascal, sehingga unit ini dapat dicoba di Free Pascal, namun dalam seleksi TOKI maupun lomba IOI *unit (baik unit bawaan kompilator, maupun buatan sendiri) tidak dipakai*.

Butir penting:

? Awali dengan review apa itu program, pemrograman dan bahasa pemrograman secara ringkas.

? Jelaskanlah kepada murid, mengapa diantara banyak "bahasa pemrograman" yang mungkin pernah mereka dengar, di kuliah ini dipilih Pascal sebagai bahasa pertama. Kepercayaan murid terhadap sesuatu yang dipelajari akan berguna, akan menambah motivasi mereka.

? Sebuah program adalah sebuah teks yang mempunyai pola tertentu. Pola paling sederhana diberikan dalam program hello.pas

? Jelaskan pemroses bahasa yang akan memproses program : interpreter dan kompiler. Yang akan dipakai di kelas adalah kompiler.

? Jelaskanlah prinsip pengolahan program dengan kompiler, dihubungkan dengan perintah Turbo Pascal.

? Jelaskan apa yang dimaksud Pascal standard dan versi kompilator Pascal, misalnya Turbo Pascal yang akan dipakai.

? Jelaskanlah bahwa karena program pascal adalah sebuah teks, maka harus diproses dengan editor teks. Hubungkan dengan perinta h-perintah pada editor.

? Akhiri dengan "aturan permainan" dalam pemrograman yang dipentingkan di kelas: patuh akan standar! jelaskanlah bahwa ini penting di dunia informatika, karena pada akhirnya berprofesi di dunia ini hanyalah mengikuti standar. Problema dari standar adalah*: there are so many of them*. Akibatnya: kita akan pakai salah satu yang disepakati akan dipakai di kelas.

Variabel Ber-type Dasar

"Variabel", adalah istilah yang diadopsi dari variabel dalam persamaan matematika, yang dalam pemrograman berarti: suatu **nama** yang **nilai**nya dapat diubah-ubah (dimanipulasi) oleh program. Jadi variabel adalah nama informasi, dan dengan mengacu ke nama, sebenarnya yang diacu adalah nilainya.

Dalam pemrograman yang ketat – type –, sebuah variabel harus ber-type tertentu, sehingga nilainya harus ber-type sesuai dengan deklarasi variabel.

Type dalam pemrograman ada yang merupakan type dasar (tidak perlu dideklarasi lagi), dan type yang didefinisikan pemrogram.

Kuliah pemrograman selalu dimulai dengan pemakaian variabel ber-type sederhana (primitif), dan baru diikuti dengan type lain Contoh wajib :

? asign.pas
? asign1.pas
? tdasar.pas

Mencakup : deklarasi type sederhana (integer, real, char, boolean), operator yang berlaku dan contoh pemakaiannya

Butir penting:

? Awali dengan melihat kembali hello.pas
? Perhatikan bagaimana sekarang kita akan menambahkan “KAMUS”, yaitu nama variabel yang dipakai lewat kata VAR
? Semua nama yang berada di antara VAR dan BEGIN disebut sebagai deklarasi variabel.
? Jelaskan beda antara deklarasi nama variabel dan nilainya.
? Nilai variabel “belum” terdefinisi ketika dideklarasi. Jika ingin dimanipulasi, harus diberi nilai. Bagaimana caranya?
? Salah satu cara memberi nilai adalah dengan assignment.
? Kita memasuki topik assignment

Assignment

- ? Assignment adalah **pemberian nilai** ke suatu **nama** variabel
- ? Sintaks assignment : pengertian ruas kiri (nama penyimpan nilai) dan ruas kanan (ekspresi)
- ? Mekanisme eksekusi assignment
- ? Standard di kuliah : assignment harus ketat type. Type ekspresi ruas kanan harus setype dengan penyimpan di ruas kiri
- ? Ajarkan deklarasi type sederhana dan bagaimana mengisi nilai dengan assignment.
- ? Jelaskan mengapa assignment dalam pascal mempunyai notasi **:=** . Jelaskanlah arti dari **"ruas kiri"** dan **"ruas kanan"** dari suatu kalimat assignment semua ruas harus setype.
- ? Nilai tanpa rumus ini akan menjadi introduksi yang baik dalam menjelaskan ekspresi
- ? Nilai yang disimpan tentunya ingin dilihat dunia luar: jelaskan instruksi writeln yang ada setelah assignment

Type String

Contoh baca / tulis string : manipstr.pas

Batasan :

? Karena sering dipakai, maka string harus dibahas diawal.
? Bahaslah hanya sekilas saja, tanpa mengatakan bahwa string adalah array.
? String pada awal dibahas sebagai sebuah “type primitif”, dan dibatasi hanya untuk baca / tulis.
? Pada saat membahas tabel / array, baru string dapat dimanipulasi sebagai array of char

Input/Output Type Dasar

Contoh wajib:

? baca.pas

Butir penting:

? Awali dengan review apa yang dilakukan assignment dan program sederhana dengan rumus yang nilai awalnya diisi dengan assignment

? Program yang nilai variabelnya hanya ditentukan dengan assigment selalu "berbuat" yang sama. Jika diinginkan *behaviour* yang berbeda, harus diubah *source code* nya, dikompilasi ulang, ditest. Betapa melelahkan

? Jika program ingin berkelakuan lain, maka harus ada cara untuk memasukkan "data" dari luar.

? Inilah pentingnya instruksi pembacaan, dalam Pascal Read

? Jelaskan lewat contoh, sebuah program yang membaca nilai variabel bertype dasar dan kemudian menuliskannya

? Untuk Pascal, harus dijelaskan perbedaan read dengan readln

? Jelaskan mengapa suatu pembacaan perlu didahului dengan **write** nama variabel yang dibaca lewat sebuah contoh program lengkap yang mengadung minimal potongan program sebagai berikut. Tanpa instruksi **write** sebelum **read**, maka pengguna harus membaca source code untuk dapat memasukkan data dengan benar. Jika urutan kacau, maka program menjadi kacau.

```
program PK1;
(* membaca nilai a,b,c *)
(* koefisien Persamaan Kuadrat dan x*)
 (* dan menuliskan hasilnya: y=ax2 + bx + c *)
VAR
  a,b,c : real; (* koef PK*)
begin
  read(a,b,c);
  y := (a*x*x2) * b*x + c;
  writeln(y);
end. (*PK1*)
```

Dibandingkan dengan :

```
program PK2;
(* membaca nilai a,b,c *)
(* koefisien Persamaan Kuadrat dan x*)
(* dan menuliskan hasilnya: y=ax2 + bx + c *)
VAR
  a,b,c : real; (* koef PK*)
begin
  write (‘Masukkan nilai a  =”); readln(a);
  write (‘Masukkan nilai b  =”); readln(b);
  write (‘Masukkan nilai b  =”); readln(b);
  y := a*(x*x) * b*x + c;
  writeln(‘Nilai y=a*x*x+b*x+ c adalah  ‘,y);
end. (*PK2*)
```

Jika tidak ada informasi mengenai input data yang harus diberikan seperti pada PK2, pemakai program akan sangat kesulitan mensuplai data sesuai urutan dalam program. Jika rumusnya "komutatif" seperti a*b sama dengan b*a, maka tidak fatal. Tapi bayangkan apa yang terjadi jika rumusnya adalah a / b, dan nilai a dan b yang dibaca terbalik urutan pemberiannya. Tanpa adanya write pesan nilai yang akan diinput, maka pemakai harus membaca program untuk dapat memasukkan data sesuai urutan yang tepat.

? Untuk selanjutnya, jika dalam spesifikasi program hanya dituliskan seperi contoh PK1, maka pemrogram harus menuliskan dengan standrad penulisan data masukan seperti pada program PK2

? Pembacaan nilai adalah cara lain untuk mengisi suatu nilai variabel.

? Soal-soal seleksi TOKI dan lomba IOI tidak pernah menggunakan input dari pengguna, namun hal ini tetap diperlukan oleh programmer dalam pengujian program.

Batasan :

Pada pengajaran awal, cara pengisian nilai variabel akan ditentukan (diberikan spesifikasinya) : apakah melalui assignment atau pembacaan. Murid tidak diharapkan untuk mampu mendesain hal ini

Ekspresi Type Dasar (Numerik, Boolean)

Contoh wajib :

? oprator.pas

Butir bahasan :

? Review bahan yang lalu: assignment nilai tanpa rumus tidak terlalu banyak gunanya. Sekarang kita akan belajar menulis "rumus".

? Rumus : ekspresi. jelaskan arti operator terhadap type tertentu (misalnya kita tidak bisa mengalikan nilai bertype char)

? Sintaks dari ekspresi sederhana (lewat contoh!):

- ✍ Sebuah nama adalah sebuah eksprersi
- ✍ Komponen operan dan operator untuk sebuah ekspresi biner dengan dua operan
- ✍ Bahwa operan boleh sebuah ekspresi (ekspresi "kompleks"). Disini muncul masalah prioritas. Perhatikan aturan penulisan berikut.

? Ekspresi harus dituliskan dengan tanda kurung untuk setiap operan. Ini mengurangi beban pemrogram terhadap independensi pemroses bahasa. Semua spesifikasi/rumus yang diberikan harus diberikan dengan tanda kurung sehingga pemrogram juga tidak bingung.

Konstanta

Konstanta adalah nilai yang tidak dikehendaki untuk diubah (sehingga tidak boleh diubah) dalam program. Konstanta (literal) seringkali dalam sebuah program diberi nama, sehingga dengan menyebutkan nama maka yang dimanipulasi adalah nilainya. Deklarasi sebuah nama sebagai konstanta akan "memproteksi" nilai yang dikandung.

Contoh wajib :

? konstant.pas
? literal.pas

Butir penting :

? Tunjukkan lewat contoh perhitungan luas, bahwa nilai PI dalam konsep matematika adalah nilai yang selalu = 3.1415….
? Nilai yang selalu tetap disebut sebagai konstanta .
? Nama yang dideklarasi sebagai konstanta akan dilindungi oleh pemroses bahaa, sehingga jika terjadi perubahan (akibat assignment atau pembacaan), pemroses bahasa dapat mensinyalkan.
? Konstanta dipakai untuk :
  ✍ Melindungi harga
  ✍ Menghindari pemakaian literal (penulisan langsung suatu nilai) yang berkali-kali, yang akan mengurangi robustness program
? Catatan khusus : Hati-hati dengan option Turbo Pascal! Jika tidak diset “strict”, notion const tidak ada artinya (default Free Pascal adalah "*strict*" terhadap const)

Batasan :

Pada pengajaran pemrograman pertama, semua konstanta diberikan spesifikasinya. Murid tidak diharapkan untuk mampu mendesain apakah suatu nama harus dijadikan konstanta. Namun, jika sebuah program mengandung “banyak” literal, murid boleh “protes”, usul untuk perbaikan spesifikasi.

Type Bentukan, Type Komposisi

Type bentukan adalah type yang terdiri dari komposisi type (boleh dasar, dan boleh type bentukan). Pemrogram mendefinisikan type bentukan dengan memberi nama type. Maka, pendefinisian type adalah :

- nama type
- konstanta
- operator terhadap type tersebut
- domain atau nilai yang terkandung dalam suatu type

Contoh wajib :

? deklarasi dan contoh aplikasi untuk : type point, JAM , murid
Jika perlu, pisahkan contoh yang ada menjadi dua program supaya murid tidak bingung
Untuk ype JAM, belum ada manipulasinya. Silahkan dibuat sendiri, sekaligus memperkenalkan "*range*" dalam bahasa Pascal.

Butir penting :

? Awali dengan review apa yang dilakukan oleh deklarasi dan pemakaian type dasar
? berikan sedikit ilustrasi pentingnya membentuk perngertian baru, dan tidak hanya ekedar type dasar, lewat contoh bericara mengenai "titik"
? Fasilitas untuk membentuk titik: dengan membuat **nama type** baru
? Tunjukkan lewat contoh : ada kata kunci TYPE sebelum VAR
? Batasi bagian awal ini hanya untuk type bentukan yang memakai keyword RECORD dalam bahasa Pascal.
? Bahas lewat contoh tahapan memakai type bentukan:
  ✍ Deklarasi type titik dengan absis dan ordinat integer

- Deklarasi variabel bertype titik
- Bagaimana mengacu komponen type bentukan: secara sederhana operator "**.**" atau dengan WITH.
- Bagaimana memanipulasi type bentukan (karena operasi hanya dapat dilakukan terhadap komponen)

Batasan :

Pada pengajaran pertama, semua definisi type bentukan dan operasinya telah diberikan spesifikasinya. Murid tidak diharapkan untuk mampu mendesain suatu type bentukan. Karena itu penting bagi pengajar untuk tidak asal-asalan membuat type bentukan. Ini akan menjadi dasar bagi kuliah ADT (Abstract Data Type) pada pengajaran selanjutnya.

Analisa Kasus (Kalimat Kondisional)

Analisa kasus (kondisional) memungkinkan sebuah program mempunyai "jalur eksekusi" yang berbeda tergantung kepada nilai variabel (berikan contoh sederhana).

Contoh wajib tahap pertama:

? if1.pas
? if2.pas
? if3.pas
? kasus.pas

Perhatikan bahwa ketiga contoh tsb mewakili satu persoalan yang sama, dengan kasus yang semakin banyak, dan ekspresi kondisi sangat sederhana.

Contoh wajib tahap kedua:

? wujudair.pas.

Contoh ini mewakili ekspresi kondisi cukup kompleks

Contoh wajib tahap ketiga:

? kuadran.pas.

Contoh ini sangat kompleks dan mewakili ekspresi kondisi terhadap type bentukan, sekaligus review terhadap type point. Selain itu, sekaligus menunjukkan analisa kasus "untuk semua kasus yang mungkin". Harus dibahas sebagai contoh terakhir.

Batasan :

Pada pengajaran pertama, semua analisa kasus harus diberikan "sketsa"nya, artinya sudah ditentukan menjadi analisa kasus yang mana. Murid tidak diharapkan untuk dapat mendesain bentuk analisa kasus yang dipakai. Namun demikian, murid harus mampu untuk mengidentifikasi "kasus bocor", yaitu adanya kasus yang belum tercakup dalam suatu analisa kasus. Jika ada analisa kasus "bocor", murid harus "protes" terhadap spesifikasi sebelum mengubah spesifikasi yang diberikan, dengan memberikan usulan perubahan. Murid tidak boleh mengubah spesifikasi tanpa mengajukan usulan perubahan.

Butir penting bahasan :

1. Kalimat kondisional yang diajarkan lewat contoh berpola sebagai berikut

```
if (kondisi) then
begin
  Aksi;
end;
```

```
if (kondisi) then
begin
   Aksi1;
else  (* not kondisi*)
begin
   Aksi2;
end;
```

```
if (kondisi_1) then
begin
   Aksi1;
else if (kondisi_2) then
begin
   Aksi2;
else if (kondisi_3) then
begin
   Aksi3;
end else  (* not kondisi*)
begin
   Aksi_n;
end;
```

```
case (expression) of
    label1:  begin
                aksi1;
             end;
    label2:  begin
                aksi2;
             end;

    label3:  begin
                aksi3;
             end;
end; (* case *)
```

2. Memeriksa analisa kasus yang benar : apakah semua kondisi sudah diprediksi (dicakup), dan adakah kasus yang sama (interseksi kasus).

Pengulangan

Kemampuan komputer yang sangat ampuh adalah mengulang. Berkat instruksi pengulangan, satu kalimat dalam program dapat dieksekusi berkali-kali, bahkan selama-lamanya (?). Ada banyak bentuk pengulangan yang disediakan dalam suatu bahasa pemrograman.

? Contoh wajib persoalan yang sama dengan bentuk beda mencetak 1,2,3…N ke bawah dengan:
   - ✍ prifor.pas
   - ✍ priw.pas
   - ✍ prirep.pas

   Jelaskanlah sintaks dan eksekusi program lewat contoh ini (kondisi berhenti adalah batas N)

? Contoh wajib dengan kondisi berhenti suatu nilai yang aneh. Dalam hal ini untuk input positif, diakhiri dengan x= -999 membaca dan menjumlahkan bilangan yang dibaca
   - ✍ whilex.pas
   - ✍ repeatx.pas
   - ✍ repeatx1.pas
   - ✍ dengan for (tidak mungkin, katakan ini). Sehingga skema menjadi lain

? Contoh gabungan : menghitung rata-rata

? Contoh wajib yang berguna sekaligus loop dalam loop: skema menu

Butir penting untuk disampaikan:

1. Bentuk pengulangan yang diajarkan : WHILE, REPEAT, FOR yang diajarkan lewat contoh berpola sebagai berikut:

```
while (kondisi_ulang) do
begin
   Aksi;
end; (* not kondisi *)
```

```
repeat
   Aksi;
until (kondisi_berhenti);
```

```
for (nama_pencacah) := Awal to akhir do
begin
   Aksi;
end;
```

Sekema pengulangan

Instruksi pengulangan tidak pernah berdiri sendiri, sehingga diperoleh skema :

```
(* inisialisasi *)
while (kondisi_ulang) do
begin
   Aksi;  (* harus membuat suatu saat kondisi berhenti *)
end; (* not kondisi *)
(* terminasi *)
```

```
(* inisialisasi *)
repeat
   Aksi; (* harus membuat suatu saat kondisi berhenti *)
until (kondisi_berhenti);
(* terminasi *)
```

```
 (* inisialisasi *)
for (nama_pencacah) := Awal to akhir do
begin
   (* nama pencacah tidak boleh diubah *)
    (* hanya dapat dipakai nilainya dlm loop*)
   Aksi;
end;
(* terminasi *)
```

2. Bahas contoh yang ada di program kecil, Jelaskanlah sintaks masing-masing bentuk pengulangan, dan perbedaan cara eksekusinya dari contoh persoalan yang sama dan dituliskan dalam 3 bentuk. Ini untuk menunjukkan kepada murid, bahwa persoalan yang sama dapat ditulis dalam bentuk yang lain. Tapi masalahnya adalah, ada yang tepat atau tidak
3. Jelaskan bahwa kebanyakan pengulangan tidak berdiri sendiri, tapi didahului dengan Inisialiasasi dan diakhiri denga Terminasi (lewat contoh yang ada).

4. Pengulangan harus berhenti. Kondisi berhenti atau kondisi pengulangan diekspresikan dalam salah satu cara:
   - ? Batas akhir pengulangan (pola membaca N), pola ini yang banyak dipakai di IOI
   - ? Nilai akhir yang khusus (mark, -999 pada beberapa contoh)
   - ? Banyaknya pengulangan, yang akhirnya diimplementasi dengan FOR

Batasan :
Pada pengajaran pertama, bentuk pengulangan yang dipakai akan ditentukan karena merupakan bagian dari pembuatan spesifikasi program. Murid tidak diharapkan untuk dapat memilih bentuk pengulangan yang tepat. Tetapi, contoh yang diberikan harus mencerminkan pemakaian bentuk yang tepat. Murid tidak boleh (dan karena contohnya benar, maka tidak perlu) protes terhadap bentuk pengulangan yang diberikan.

Namun demikian, murid harus mampu untuk mengidentifikasi "loop tidak berhenti", yaitu adanya pengulangan terus menerus. Jika ada kasus loop tidak berhenti, murid harus "protes" terhadap spesifikasi sebelum mengubah spesifikasi yang diberikan, dengan memberikan usulan perubahan. Murid tidak boleh mengubah spesifikasi tanpa mengajukan usulan perubahan.

Contoh pengulangan yang minimal harus diajarkan:
1. Mencetak angka 1 s/d N
2. Membaca nilai kemudian menghitung rata-rata
3. Membaca data sampai benar
4. Membuat menu dan memproses prosedur (jika sudah diajarkan)
5. Mengulang membaca sandi 3x (dengan membandingkan apa yang diketikkan dengan suatu nilai yang disimpan program), jika 3 kali salah, maka berhenti.
6. Mencetak "segitiga" pascal

Subprogram (Fungsi dan Prosedur)

Subprogram dapat ditulis dalam bentuk prosedur atau fungsi.

Fungsi
Dalam pemrograman merepresentasi pengertian fungsi pada matematika : adalah pemetaan "domain" (list parameter formal) ke "range" (type hasil). Dengan fungsi, semua sire effect diharapkan tidak ada.

Pola kalimat spesifikasi fungsi adalah :

> ***Diberikan*** *…<deskripsi keadaan/nilai parameter formal> ..* ***menghasilkan*** *…<type hasil dan daerah nilainya>..* ***dengan deskripsi proses****/rumus…*

Prosedur
Mencerminkan "sub aksi", yaitu penentuan keadaan antara jika program dianggap suatu "aksi", yaitu efek neto yang dihasilkan dalam waktu yang tertentu, dengan Initial State (I.S., keadaan awal) dan Final State (F.S., keadaan akhir) yang jelas.

Pola kalimat spesifikai prosedur :

> ***I.S****. : nilai variabe global, parameter*
> ***F.S****. : nilai variabel global, parameter*
> ***Proses*** *: deskripsi proses*

Contoh wajib Fungsi:

? Fungsi dengan tipologi kasus sbb:

- Function FX2(x: real) return real; (* fungsi menerima type dasar menghasilkan type dasar dengan kalkulasi sederhana *)

- Function max2bil(a,b:integer); (* fungsi menerima type dasar menghasilkan type dasar dengan analisa kasus *)
- Function LGrs (P1,P2:point) return real; (*fungsi menerima type bentukan menghasilkan type dasar *)
- Function BuatPoint (x,y: integer) return Point (* fungsi menerima type dasar menghasilkan type bentukan *)
- Function TitikTengah (P1,P2:point) return Point; (* fungsi menerima type bentukan menghasilkan type bentukan . Perhatikan implementasi khusus karena bahasa Pascal standar tidak memungkinkan hal ini secara natural*)

Contoh wajib Prosedur:

- Prosedur tanpa parameter yang mencerminkan struktur mendasar: Input; Proses; Output
- Procedure Add (a,b:integer; VAR sum: integer); (* parameter input dan output jelas *)
- Procedure incr (VAR a:integer); (* mengubah diri sendiri, type dasar *)
- Procedure geserPoint (x,y:integer; VAR P:point); (* mengubah diri sendiri, type bentukan *)
- Swap.pas (untuk prosedur dengan VAR)

Butir penting bahasan :

1. Jelaskan efek eksekusi prosedur dan fungsi tanpa parameter
2. Jelaskan beda antara parameter by Value dan by Reference lewat contoh beserta efeknya
3. Jangan membingungkan murid dengan contoh yang sedikit variasinya padahal secara semantik tidak ada beda yang signifikatif
4. Jelaskan keterbatasan fungsi dalam Pascal yang tidak mampu mengembalikan type bentukan, dan cara implementasinya (function Titiktengah yang dijadikan prosedur). Jelaskan dengan baik, bahwa ini adalah hanya salah satu cara implementasi (cara mengatasi hal yang tidak mungkin) tanpa memperlebar diskusi.
5. Jelaskan struktur bersarang program Pascal dan **scope** (scope.pas) lewat contoh! Mintalah murid untuk mengeksekusi program yang diberikan, dan membuat catatan pengamatan tentang scope ini

Batasan :

1. Pada pengajaran pertama, bentuk subprogram yang dipakai (prosedur atau fungsi) akan ditentukan karena merupakan bagian dari pembuatan spesifikasi program. Murid tidak diharapkan untuk dapat memilih

implementasi suatu modul menjadi prosedur/fungsi. Tetapi, contoh yang diberikan harus mencerminkan pemakaian bentuk yang tepat.

2. *Semua contoh yang ditampilkan menjadi prosedur/fungsi harus atomik, artinya menimbulkan efek yang "manunggal", tidak campur aduk.*
3. *Pada kuliah ini, semua spesifikasi Prosedur dan Fungsi, yaitu NAMA, PARAMETER dan JENISnya, akan diberikan. Murid tidak diharapkan mampu untuk mendesign Prosedur/Fungsi!*
4. *Pada kuliah ini murid tidak diharapkan mendesain variabel global dan lokal. Ini menyangkut spesifikasi program. Namun contoh yang diberikan harus mencerminkan lokalisasi variabel yang benar dan bukan sembarangan variasi yang menunjukkan efek.*

Tabel Berdimensi Satu (Array)

Kata kunci dalam menjelaskan tabel :

- satu nama banyak nilai
- akses seluruh tabel (operasi terbatas) dan a kses elemen lewat indeks
- struktur internal (masa hidup selama program dieksekusi)
- struktur kontigu: elemen demi elemen berdekatan letaknya di memori (ini tidak penting dijelaskan untuk pemula)
- struktur statik: alokasi memori dilakukan saat deklarasi (ini juga tidak perlu untuk pemula, tapi akan dijelaskan di struktur data)

Contoh wajib :

1. Tabel dengan type integer
   - ? tabel1.pas yang isinya : deklarasi, pengisian dan penulisan tabel **integer tanpa mendefinisikan nama type**. Tanpa prosedur dan tanpa fungsi
   - ? tabel2.pas : pendefinisian nama type, nama variabel dan pengisian serta penulisannya. Program ini sebenarnya sangat mirip dengan program sebelumnya
   - ? beberapa proses sederhana terhadap tabel integer : search (satu cara) dan sort (satu metoda)
2. Tabel dengan elemen bertype caharacter : string
3. Tabel dengan elemen bertype record : tabel of Point dengan kerangka yang sama dengan tabel2.pas
4. Pengenalan type enumerasi sebagai indeks table.
   Berikan satu contoh tabel dengan indeks type enumerasi. Ini merupakan contoh gabungan, dan hanya dapat diimplementasi dalam bahasa tertentu (dapat diimplementasi dalam Pascal). Misalnya indeksnya adalah nama hari, isinya nama hari dalam bahasa Inggris. Dalam hal implementasi dalam suatu bahasa tidak mungkin secara primitif, maka murid harus menuliskan implementasi yang tidak natural (tujuan tercapai dengan memprogram)

Butir yang perlu dibahas :

? Pengertian tabel dibandingkan type dasar : satu nama banyak nilai. Untuk ini diperlukan indeks

? Pengertian tabel dibandingkan record : satu nama komponennya sama

? Berikan contoh program tabel integer dengan batas elemen yang "terdefinisi" secara ekslisit ataupun implisit

? Berikan gambaran bahwa pemakaian tabel harus hati-hati, sebab memakan banyak memori dan sebelum program dijalankan, perancang harus tahu berapa space yang dipesan (indeks tabel)

? Jelaskan pentingnya pemakaian FOR untuk iterasi terhadap elemen tabel. 90% loop terhadap tabel menggunakan bentuk pengulangan ini

? Pengertian variabel internal: Jelaskan bahwa semua nilai yang disimpan dalam tabel akan “hilang” ketika program selesai, sama halnya dengan nilai yang lain. Jika yang lain hanya “sedikit”, maka untuk tabel (terutama jika berukuran besar), maka sayang bahwa kita harus kehilangan nilai. Ini merupakan introduksi yang baik ketika akan mengajarkan File eksternal.

? Review prosedur dan fungsi dihubungkan dengan tabel: jika tabel dipakai sebagai parameter untuk prosedur dan fungsi, maka harus punya nama type. Berikan sebuah contoh. Diberikan sebuah deklarasi type

```
TYPE
  Tabint = array [1..10] of integer;

procedure BacaIsiTab(var T:Tabint);
procedure TulisIsiTab (T:Tabint);
function SalinTab (T:TabInt): TabInt;
(* diberikan sebuah tabel, *)
(* hasilkan salinan tabel itu *)
```

? Tabel dengan batas elemen terdefinisi eksplisit dibandingkan implicit

```
TYPE
  Tabint = array [1..10] of integer; (* implisit*)

(* dibandingkan dengan *)
TYPE
 Tabint = record
             T: array [1..10] of integer;
             Len : integer;
           end; (* eksplisit*)
```

Batasan:

1. *Pada pengajaran pertama, murid tidak diharapkan mendesain suatu variabel menjadi tabel atau bukan. Ini masalah spesifikasi.*
2. *Murid juga tidak diminta untuk menentukan apakah deklarasi type tabel dengan batas elemen terdefinisi "implisit" atau "eksplisit"*
3. *Latihan soal untuk Pemrosesan terhadap tabel harus mencakup yang sudah diajarkan di kuliah ini.*
4. *Jika belum diperlukan, Tabel berdimensi dua (matriks) atau lebih belum dicakup dalam pengajaran pertama. Jika dikehendaki, hanya berikan introduksi dan tidak perlu dibahas secara khusus di kelas.*
5. *Perlu ditekankan mengenai tipe enumerasi standar (byte, integer, karakter) agar murid paham mengenai penggunaan instruksi case .. of, tapi tipe enumerasi bentukan (misalnya tipe enumerasi hari) tidak diperlukan untuk IOI.*

Tabel Ber-indeks Banyak (Matriks, Tabel 3 Dimensi)

Setelah mengenal tabel berdimensi satu, murid dikenalkan akan tabel berdimensi dua (matriks). Berdasarkan cara deklarasi dan pemakaian matriks, diperkenalkan tabel dengan indeks tiga. Lebih dari tiga sudah bukan cakupan kuliah pemrograman dasar (mungkin akan dicakup secara lebih spesifik pada kuliah struktur data)

Contoh wajib:

? Tab2dimp.pas
? Tab3dim.pas

File Eksternal

Setelah diperkenalkan tabel yang merupakan struktur internal, harus dijelaskan bagaimana penyimpanan informasi yang "persistent", yang tidak hilang ketika program sudah selesai dieksekusi. Inilah pentingnya file eksternal.

Contoh wajib: masing-masing satu contoh pola pembacaan dan penulisan:

- file teks: bacatxt.pas, rekam.txt
- file dengan elemen integer: rwint.pas
- file dengan elemen type bentukan point: frec.pas
- menyimpan isi tabel integer ke file eksternal (belum ada contohnya)

Butir yang perlu dibahas :

- Pengertian “file”. Bahwa sebuah program adalah sebuah file. Bahwa sebuah file data dibedakan dengan “file program”.
- Jelaskan beda **text file** dengan **file data** yang lain. Text file dapat dibuat dan dibaca oleh editor teks apa saja yang dipakai. File data (bukan teks) harus dibuat dan dibaca oleh sebuah program khusus. Untuk ini, murid harus mencoba membuka sebuah file of integer yang dibuat dengan program Pascal lewat editor!
- Jelaskan pengertian eksternal dibandingkan dengan internal (tabel).
- Tekankanlah bahwa ada banyak sekali cara untuk membaca, memproses, menulis file dan sangat tergantung kepada pemroses bahasa. Karena banyaknya cara ini, maka dipilih yang standard dan secukupnya untuk bahan perkuliahan ini. Dijamin cara standard ini akan dapat menjawab kebutuhan pemrograman.
- Berikan skema pembacaan, pemrosesan sederhana dan penulisan file lewat contoh. Tidak lewat teori.
- Jelaskan primitif yang selalu diperlukan dalam memproses file :
  - assign ke nama fisik
  - membuka file

- memproses / baca satu per Satu
- menulis sebuah rekaman
- menutup file

? Walaupun file baru dapat dibaca setelah ditulis, beberapa latihan / contoh program justru dimulai dari membaca file.

? Program membaca dan menulis untuk file data (bukan file teks) harus resiprok. Harus komplementer satu sama lain.

? Harus dibuat sebuah contoh yang menggabung file dan tabel, yaitu menyimpan isi tabel ke file eksternal supaya isinya tidak hilang. Jika perlu, berikan sebagai latihan ke murid.

Batasan :

1. *Pada kuliah ini, murid tidak diharapkan mendesain file eksternal. Semua spesifikasi file eksternal diberikan*
2. *Proses yang dilakukan terhadap isi file adalah proses sederhana, dan pemrosesan secara Sequential*
3. *Pada kuliah ini belum dibahas sebuah file dengan isi campuran (seperti dlm beberapa bahasa lain: misalnya baris pertama berisi data N yaitu banyaknya data yang dibaca, baris kedua dan selanjutnya bertype record sebanyak N). Jika sempat, baru boleh ditambahkan. Tapi kalau waktu sudah sempit, jangan dulu.*
4. *Pada IOI yang diperlukan hanyalah input dan output file teks (text).*
5. *Di sistem operasi Linux, nama file bersifat case sensitive, jadi kita bisa membuat dua file yang bernama "Data" dan "data" (hanya beda kapitalisasi saja).*

Pembagian Sebuah Program Dalam Beberapa File

(Hanya diajarkan jika fasilitas tersedia pada lingkungan yang dipakai !!!)

Implementasi Unit dalam Pascal

Contoh wajib :

- buat sebuah unit point
- buat sebuah program utama yang memanfaatkan unit tsb

Butir yang dibahas:

- Unit memungkinkan satu file program dipecah menjadi banyak file, sehingga program dapat dibagi untuk dikerjakan beberapa orang. Jika tidak dibagi menjadi banyak file: percuma, karena satu file hanya dapat dibuka oleh satu user.
- Implementasi unit untuk program utama dan prosedur-prosedur yang merealisasi pemanggilan menu. Review pengulangan lewat menu.pas dengan realisasi menu yang ada di file lain.
- Buat unit POINT dan beberapa operasinya, serta berikan minimal dua buah main program untuk memanfaatkan point (ini di kemudian hari menjadi ADT. Tapi jangan katakan sekarang).
- Suatu unit harus disertai dengan main program yang berfungsi sebagai driver untuk mengetest unit tersebut. Driver harus disimpan, dan akan dibawa jika source code dibawa ke mesin lain. Dengan menyimpan driver ini, maka reusability program dijamin sampai dengan pengetesan ulang.
- Biasanya penyimpanan source code diorganisasi dalam sebuah sub direktori (dengan nama yang sama dengan nama unit), yang isinya adalah file sebagai berikut :
  - Primitif Unit
  - Driver (main program untuk mentest)

Batasan :

Pada pengajaran pertama, pembagian sebuah program menjadi unit-unit program telah ditentukan dalam spesifikasi, dan mengikuti aturan di atas. Murid tidak diharapkan untuk mendesain unit. Murid diploma hanya ditargetkan untuk mampu membaca spesifikasi, kemudian melakukan pengkodean instruksi berdasarkan spesifikasi, dan kemudian membuat driver untuk mentest unit yang bersangkutan.

Unit tidak dibutuhkan (dan tidak diijinkan pada IOI), namun berguna untuk diajarkan agar dapat memakai unit yang sudah tersedia dengan baik, karena murid mengetahui internal dari sebuah unit.

Topik Lanjut

Topik lanjut adalah topik yang tidak dibahas pada pengajaran program dasar, kecuali pengajaran berorientasi ke bahasa. Contoh : setelah mengenal beberapa bahasa dan mendapat pengetahuan dasar yang kuat mengenai algoritma dan struktur data, maka topik lanjut dapat diberikan. Yang penting adalah : murid harus menguasai hal-hal yang mendasar (dan banyak dipakai), baru diarahkan untuk mempelajari sendiri hal-hal yang spesifik (dan jarang dipakai)

Topik lanjut biasanya dikaitkan dengan struktur data akan diajarkan ketika kuliah Struktur Data .

- Type dan instruksi yang sangat spesifik terhadap bahasa
- Record variant
- Pointer
- Linked list
- Struktur spesifik

CONTOH PROGRAM KECIL

Dalam Bahasa Pascal

PROGRAM SEDERHANA

```
program hello;
(* File : HELLO.PAS *)
(* menuliskan Hello ke layar *)

begin
 writeln ( 'hello ') ;
end.
```

```
program hellodos;
(* File : HELLODOS.PAS *)
(* menuliskan Hello ke layar *)

uses crt;

begin
 clrscr;
 writeln ( 'hello ') ;
end.
```

INPUT/OUTPUT

```
program baca;
(* File : BACA.PAS *)
(* contoh membaca integer*)
(* kemudian menuliskan nilai yang dibaca *)
(* Kamus *)

var
   a : integer;

begin
(* Program *)
   writeln ('Contoh membaca dan menulis, ketik nilai integer: ') ;
   readln (a) ;
   writeln ('nilai yang dibaca : ', a) ;
end.
```

ASSIGNMENT

```
program asign;
(* File : ASIGN.PAS *)
(* Assigntment dan print *)

var
(* Kamus *)
 i : integer;

begin (*Algoritma *)
writeln ( 'hello' );
 i := 5;
 writeln ( 'Ini nilai i : ',i);
end.
```

```
program asign1;
(* File : ASIGN1.pas *)
(* Assignment dan print *)
(* Kamus *)

var
 i : integer;
 ii : longint;

begin (* Algoritma *)
writeln ( 'hello');
 i   := 1234;
 ii  := 123456 ;
 writeln ( 'Ini nilai i=1234   = : ',i);
 writeln ( 'Ini nilai ii=123456 : ',ii);
 writeln ( 'Ini nilai max integer: ',maxint);
 writeln ( 'Ini nilai max longint: ',maxlongint);
end.
```

TIPE DASAR

```
program TDasar;
(* File : TDasar.pas *)
(* Deklarasi, assignment dan penulisan type dasar *)
(* Kamus *)
var
    i : integer ;
    x,y : real ;
    found : boolean;
(* Algoritma *)

begin
    i:= 5; writeln ('i = ', i);
    x := 0.5; writeln ('x = ', x);
    y := 1.0e + 3; writeln ('y = ', y);
    found:= true; writeln ('Found = ', found);
end.
```

EKSPRESI DAN OPERATOR

```
Program oprator;
(* File : oprator.pas *)
(* Contoh pengoperasian variabel bertype dasar *)
(* Kamus *)

VAR
    Bool1, Bool2, TF : Boolean;
    i, j, hsl : Integer;
    x,y,res : real;

Begin
(*  algoritma *)
   writeln ('Utk program ini, baca teksnya dan tambahkan output');
   Bool1 := True; Bool2 := False;

(** contoh-contoh ekspresi: bukan untuk assignment berulang-ulang **)
   TF := Bool1 And Bool2 ;
   TF := Bool1 or Bool2 ;
   TF := Not Bool1 ;
   TF := Bool1 Xor Bool2 ;

(*  operasi numerik *)
   i := 5; j := 2 ;
   hsl := i+j; hsl := i - j; hsl := i div j; hsl := i * j;
   hsl := i div j ; (* pembagian bulat *)
   hsl := i Mod j ; (* sisa *)

(*  operasi numerik *)
   x := 5.0 ; y := 2.0 ;
   res := x + y; res := x - y; res := x / y; res := x * y;

(*  operasional relasional numerik *)
   TF := i < j; TF := i > j; TF := i <= j;
   TF := i >= j; TF := i <> y;

(*  operasional relasional numerik *)
   TF := x < y; TF := x > y; TF := x <= y;
   TF := x >= y; TF := x <> y;
end.
```

STRING

```
program manipstr;
(* manipulasi string sederhana *)

var
      str1, str2 : string;
      strr : string;
      stri : string;
          i, kode: integer;

begin
      str1 := 'Saya ';
      str2 := ' Belajar di ITB';
      strr := str1 + str2;
      readln (strr);

    (* koneversi string numerik ke nilai integer *)

    stri := '123';
    val (stri, i, kode ) ; (* amatilah nilai kode setelah eksekusi *)
    writeln(stri, '-', i,'-', kode);
end.
```

```
program BACASTR;
(* File : BACASTR.pas *)
(* alokasi string, kemudian mengisinya dengan membaca *)
(* Kamus *)

VAR
   str : string;
   str1 : string;
begin
(* Program *)
  writeln ( 'Baca string, maks 256 karakter: ');
  readln ( str);
  writeln ( 'String yang dibaca : ' ,str);
  str1 := str;
  writeln ( 'String yang disalin : ',str1);
end.
```

KONSTANTA

```
program KONSTANTA;
(* File : konstant.pas *)
(* Membaca jari-jari, menghitung luas lingkaran *)
(* Latihan pemakaian konstanta *)
(* Kamus *)

   CONST
         pi=3.1415;

   VAR
         r : real;

begin
(* program *)

    write ('Jari-jari lingkaran =');
    readln (r) ;
    writeln ('Luas lingkaran = ', pi*r*r);
    writeln (' Akhir program ');

   (* Kompilasi, amatilah apa yang terjadi jika komentar sbb. dibuang *)
   (* perhatikan option pada kompilator anda *)
   (* pi := 10.0; *)

end.
```

```
program KONSTAN2;
(* File : KONSTAN2.PAS *)
(* Menghitung luas lingkaran, dari jari-jari yang dibaca *)
(* Latihan pemakaian konstanta *)
(* Kamus *)

CONST
   pi = 3.1415;
     dua = 2.0;

VAR
        r : real;
        luas : real;
        kel  : real;
```

```
begin
(* program *)
   writeln ( 'Jari-jari lingkaran =');
   readln (r) ;
   luas := pi * r * r;
   writeln ( 'Luas lingkaran = ', luas:6:2);
   kel := dua* pi * r ;
   writeln ('keliling lingkaran= ',luas:6:2);
   writeln ('akhir program ');
 end.
```

HIMPUNAN (SET)

```
program Himpunan;
(* File : Himpunan.pas *)
(* Pendefisinian dan pemanfaatan himpunan : SET *)
(* Kamus *)

type
     Hari = (senin,selasa,rabu,kamis,jumat,sabtu,minggu);
     weekday = SET of Hari;

var
     H : Hari;
     H0, H1, H2 : Hari;
     W : weekday;

(* Algoritma *)
begin
      (* Instruksi berikut salah :type set tidak dapat ditulis/baca*)
      (* writeln (' Hari = ', H); *)
      (* Assignment : boleh *)

      H1 := selasa;

      (* prosedur terdefenisi *)

      H2 := succ (H1);
      H0 := pred (H1);

      (* pemanfaatan untk mengontrol pengulangan *)
      (* Akan dibahas pada pengulangan *)

      for H := senin to minggu do
      begin
            writeln ( 'Selamat Pagi ...');
            writeln ( 'Ordinal : ', ord (H) );
      end;

      (* intruksi CASE : akan dibahas pada analisa kasus*)

      case H1 of
          senin :
               writeln ( 'senin' );
          selasa :
               writeln ( 'selasa' );
```

```
          rabu :
               writeln ( 'rabu' );
          kamis :
               writeln ( 'kamis' );
          jumat :
               writeln ( 'jumat' );
          sabtu :
               writeln ( 'sabtu' );
          minggu :
               writeln ( 'minggu' );
          else
               writeln ( 'tidak terdefinisi ');
     end;
end.
```

RECORD, Type Komposisi, Type Terstruktur

```
program tipe;
(* File : tipe.pas *)
(* contoh pendefinisian dan pengisian struktur *)

TYPE
     Point = record
                    X : integer; (* absis *)
                    Y : integer; (* ordinat*)
                  end;

     MAHASISWA = record
                       NIM : integer;
                       Nama : string;
                       Nilai : real;
                  end; (* mahasiswa *)

VAR
      P1 : Point;
      P2 : Point;
          Mhs : MAHASISWA;

begin
    writeln ( 'Contoh mengisi struktur dengan assinment : ');
    writeln ( 'Titik P1, dengan P1.x dan P1.y: ');
    P1.x := 1;
    P1.y := 2;
    writeln ( 'P1.X= ', P1.X);
    writeln ( 'P1.Y= ', P1.Y);
    writeln ( 'Baca Titik P2');
    write ( 'Absis : ');
    write ( 'Ordinat : ');
    readln (P2.Y) ;
    writeln ( 'Koordinat : ', P2.X, ', ', P2.Y);
    mhs.nama := 'Juliette';
    mhs.nim   := 7473;
    mhs.nilai  := 80;
    writeln ( 'Hasil assignment thd Mhs ');
    writeln ( 'Nama = ',Mhs.Nama);
    writeln ( 'Nim   = ',Mhs.NIM);
    writeln ( 'Nilai  = ',Mhs.Nilai:6:2);
```

```
(* pemakaian WITH untuk record *)
    writeln ( 'Hasil assignment thd Mhs ');
    with Mhs do
      begin
        writeln ( 'Nama = ',Nama);
        writeln ( 'Nim   = ', NIM);
        writeln ( 'Nilai  = ',Nilai:6:2) ;
      end;
end.
```

```
program bacarec;
(* File : Bacarec.PAS *)
(* contoh membaca record*)
(* kemudian menuliskan nilai yang dibaca *)
(* Kamus *)

TYPE Point= record
               x : integer; (* absis *)
               y : integer; (* ordinat *)
            end; (* type Point *)

var
    P1, P2 : Point;

begin
(* Program *)
   writeln ( 'Contoh membaca dan menulis titik ') ;
   write ( 'Absis = '); readln(P1.x);
   write ( 'Ordinat = '); readln(P1.y) ;
   writeln ( 'Titik yang dibaca : (', P1.x, ', ', P1.y , ')');

   with P2 do
   begin
     write ( 'Absis = '); readln (x) ;
     write ( 'Ordinat = '); readln (y);
     writeln ( 'Titik yang dibaca : ( ', x, ', ', y, ')');
   end;
end.
```

ANALISA KASUS, KONDISIONAL

```
(* File : IF1.PAS *)
(* contoh pemakaian IF satu kasus *)
(* membaca nilai integer, menuliskan nilainya jika positif *)

Program IF1;
(* Kamus *)
var
    a : integer;

begin
(* Program *)
   writeln ( 'Contoh IF satu kasus ');
   write ( 'Ketikkan satu nilai integer : ');
   readln (a);
   if (a >= 0) then
    begin
      writeln ( 'Nilai a positif... ', a);
    end;
end.
```

```
(* File :IF2.PAS *)
(* contoh pemakaian IF dua kasus komplementer *)
(* Membaca sebuah nilai, *)
(* menuliskan 'Nilai a positif , nilai a', jika a >=0 *)
(*                    'Nilai a negatif , nilai a', jika a <0 *)

program IF2;
(* Kamus *)
var
    a : integer;

begin
(* Program *)
   writeln ('Contoh IF dua kasus ');
   write ( 'Ketikan suatu nilai integer :');
   readln (a);
   if (a >= 0) then
    begin
        writeln ( 'Nilai a positif ', a);
    end else (* a<0 *)
    begin
        writeln ( 'Nilai a negatif ', a);
    end;
end.
```

```
(* File : IF3.PAS *)
(* contoh pemakaian IF dua kasus komplementer *)
(* Membaca sebuah nilai, *)
(* menuliskan 'Nilai a positif , nilai a', jika a>0 *)
(*      'Nilai a sama dengan nol , nilai a', jika a =0 *)
(*      'Nilai a negatif , nilai a', jika a <0 *)

program IF3;
(* Kamus *)
var
   a : integer;

begin
(* Program *)
  writeln ( 'Contoh IF tiga kasus');
  write ( 'Ketikkan suatu nilai integer :');
  readln (a);
  if (a > 0) then
   begin
      writeln ( 'Nilai a positif ', a);
   end else
   if (a=0) then
   begin
     writeln ( 'Nilai a sama dengan nol ', a);
   end else if (a<0) then
   begin
     writeln ( 'Nilai a negatif ', a);
   end;
end.
```

```
program KASUS;
(* File : KASUS.PAS *)
(* Contoh kasus dengan intruksi CASE *)
VAR
(* Kamus *)
  cc : char;

begin
(* Program *)
  writeln ( 'Ketikkan sebuah huruf, akhiri dengan RETURN ');
  readln (cc);
  case cc of
     'a' : begin
               writeln ( ' Yang anda ketik adalah a ' );
           end;
     'u' : begin
```

```
              writeln ( ' Yang anda ketik adalah u ' );
          end;
     'e' : begin
              writeln ( ' Yang anda ketik adalah e ' );
           end;
     'o' : begin
              writeln ( ' Yang anda ketik adalah o ' );
           end;
     'i' : begin
              writeln ( ' Yang anda ketik adalah i ' );
           end
    else writeln ( ' Yang anda ketik adalah huruf mati atau angka' );
  end;
end.
```

```
(* File : wujudair.PAS *)
(* contoh pemakaian IF tiga kasus : wujud air *)

program wujudair;
(* Kamus : *)
var
    T: integer;

begin
(* program *)
   writeln ( 'Contoh IF tiga kasus ' );
   write ( 'Temperatur (der. C) = ' );
   readln (T);
   if (T < 0) then
    begin
        writeln ( 'Wujud air beku ');
    end else
    begin
        if ( (0<=T) and (T<=100) ) then
         begin
             writeln ( 'Wujud air cair ');
         end else
         begin
           if (T>100) then
           begin
              writeln ( 'Wuju air uap/gas ');
           end;
          end;
    end;
end.
```

```
program MAX2;
(* File :MAX2.PAS *)
(* Maksimum dua bilangan yang dibaca *)
VAR
(* Kamus *)
 a,b : integer;

begin
(* Program *)
   writeln ( 'Maksimum dua bilangan : ' );
   write ( 'Ketikan bilangan pertama : ' );
   readln (a) ;
   write ( 'Ketikan bilangan kedua : ' );
   readln (a) ;
   if (a >= b) then
      begin
        writeln ( 'Nilai a yang maksimum ', a);
      end
   else
      begin
        writeln ( 'Nilai b yang maksimum ', b) ;
      end ;
end.
```

SUBPROGRAM

```
program subprg;
(* File : subprg.PAS *)
(* contoh pendefinisian dan pemanggilan FUNGSI dan PROSEDUR *)
(* deklarasi dan badan procedure/fungsi LOKAL *)
(* Konsep yang harus dijelaskan :*)
(* - perbedaan fungsi dan prosedur *)
(* - parameter formal dan aktual *)
(* - passing parameter by value dan by ref *)

var
    a, b : integer;

function maxab (a,b : integer) : integer;
begin (* mencari maksimum dua bilangan bulat *)
     if a>=b then
       begin
          maxab := a;
       end else
       begin
          maxab := b;
       end;
end;

procedure tukar (var a, b : integer);
(* menukar nilai dua buah variabel a dan b *)
(* parameter input/output *)

var
    temp : integer;

begin (* menukar dua bilangan bulat *)
  temp := a;
  a := b;
  b := temp;
end;

begin (* program utama *)

(* Membaca dua bilangan integer *)
(* Menuliskan maksimum dua bilangan yang dibaca dg memanggil fungsi*)
(* Menukar kedua bilangan dengan 'prosedur' *)
   writeln ( 'Maksimum dua bilangan : ');
   writeln ( 'Ketikkan bilangan pertama : ');
   readln (a) ;
```

```
   writeln ( 'Ketikkan bilangan kedua : ');
   readln (b) ;
   writeln ( 'Ke dua bilangan : a = ',a) ;
   writeln ( '                          b = ',b) ;
   writeln ( 'Maksimum =    ', maxab (a,b) ) ;
   writeln ( 'Tukar kedua bilangan... ') ;
   tukar ( a, b ) ;
   writeln ( 'Ke dua bilangan setelah tukar: a = ',a) ;
   writeln ( '                               b = ',b) ;
end.
```

LINGKUP (SCOPE)

```
program Lingkup;
(* File : Lingkup.pas *)
(* arti nama : Lingkup dan masa hidup variabel; parameter by value, by ref *)
(* Kamus *)
var
a, b : integer;

procedure Plusl (var x:integer);
(* x adalah parameter input/output *)
(* Menembah nilai thd Prosedur Plusl *)
(* Fungsi lokal thd Prosedur Plus1 *)

function Incr (i:integer) : integer;
(* mengirimkan nilai i ditambah 1 *)

begin
     Incr := i+1;
end;

begin
        x := Incr(x)  ;  (* nilai parameter input x ditambah 1 *)
end;

Procedure Swap (var a,b : integer );
(* perhatikan bada a dan b disini dengan deklarasi a dan b global *)

var
 temp : integer; (*variabel lokal *)

begin
 temp := a;
 a:= b;
 b:= temp;
end;

function Plus2 (i:integer) : integer;
(* Mengirimkan nilai i ditambah 2 *)

begin
    Plus2 := i+2;
end;
```

```
(* Algoritma program utama*)
begin
  (* berikut tidak dikenal *)
  (* writeln ("Nilai i= ',i); *)
   a := 2;

  (* Berikut ini salah *)
  (* b := Incr (a) ;    *)
   b := a;
   Plus2 (a);
   swap (a,b);

  (* Berapa hasilnya ?? *)
  writeln ('1. a= ',a, 'b= ', b );
   b := Plus2 (b);
  (* Berapa hasilnya ?? *)
  writeln ('2. a= ',a, 'b= ', b );
end.
```

```
program FuncRec;
(* File : FuncRec.pas *)
(* Fungsi yang harus mengembalikan type bentukan : tidak mungkin *)
(* Kamus *)
type
          Point = record
                    x : integer; (* absis*)
                    y : integer; (* ordinat*)
                  end;
var
     T, T1, T2 : Point;
     procedure Tulis (T:Point);

(* menuliskan sebuah titik T *)
begin
    writeln ( 'Titik T ( ',T.x, ', ',T.y, ') ');
end;

(* function MidPoint (T1,T2: Point) : Point; *)
(* Menghasilkan Titik tengah T1,T2 berupa titik *)
(* karena fungsi dalam bahasa Pascal tidak bisa mengembalikan *)
(* type record *)
(* berikut ini transformasi untuk mendapatkan efek yang dimaksud *)

procedure TTkTengah (T1,T2 : Point;
                                  var MidPoint : Point) ; (* titik hasil *)
```

```
(* Menerima T1 dan T2 dua buah Point *)
(* Menghasilakan Midpoint : sebuah VARIABEL bertype Point *)

begin
      MidPoint.x := ((T1.x + T2.x) div 2 );
      MidPoint.y := ((T1.y + T2.y) div  2 );
end;

(* Algoritma *)
begin
  T1.x := 0; T1.y := 0;
  T2.x := 10; T2.y := 10;
  Tulis (T1);
  Tulis (T2);
  TtkTengah (T1,T2,T);
  Tulis (T);
end.
```

PENGULANGAN

```
program PRIFOR;
(* File : PRIFOR.PAS *)
(* Baca N, Print 1 s/d N dengan FOR *)
(* Kamus *)
var
    i  : integer;
    N: integer;

begin
(* program *)
writeln ( 'Baca N, print 1 s/d N ');
write ( 'N = ');
readln (N) ;
for i:=1 to N do
    begin
       writeln (i);
    end; (* FOR *)
  writeln ( 'Akhir program ');
end.
```

```
program PRIW;
(* File : PRIW.PAS *)
(* Baca N, *)
(* Print i = 1 s/d N dengan WHILE *)
VAR
(* Kamus *)
  n : integer;
   i : integer;

begin (* Program loop WHILE *);
 write ( 'Nilai N = ');
 readln (N);
 i := 1 ;
 writeln ( 'Print i dengan WHILE: ');
 while (i<=N) do
    begin
      writeln (i);
      i := i + 1;
    end ; (* i>N *)
end.
```

```
program PRIREP;
(* File : PRIREP.PAS *)
(* contoh baca N, *)
(* print 1 s/d n dengan REPEAT *)
(* Kamus : *)
var
  N : integer;
  i   : integer;

begin (* Program *)
  write ( 'Nilai N= ');
  readln (N);
  i := 1 ;
  writeln ( 'Print i dengan REPEAT: ');
  repeat
     writeln (i);
     i := 1;
  until (i > N);
end.
```

```
program PRITER;
(* File : PRITER. Pas *)
(* Baca N, *)
(* Print i = 1 s/d N dengan ITERATE *)
(* Kamus : *)
VAR
  N : integer;
  i   : integer;
  stop : boolean;

begin
(* Program *)
   write ( 'Nilai N = ');
   readln (N);
   i := 1;
   writeln ( 'Print i dengan ITERATE : ');
   stop := false;
   repeat
     writeln (i);
   if (i=N) then stop := true else
     begin i := i + 1; end;
   until stop;  (* i= N *)
end.
```

```
program KASUSREP;
(* File : KASUSREP.PAS *)
(* Contoh kasus dengan switch dan pengulangan *)
(* membaca karakter sampai user mengetikkan q *)
VAR
(* Kamus *)
   cc : char;
   quit : boolean;

begin
(* Program *)
   repeat
      quit := false;
      write (' Ketikkan sebuah huruf, akhiri dengan q :');
      readln (cc);

      case cc of
        'a' : begin
                writeln (' Yang anda ketik adalah a ');
              end;
        'u' : begin
                writeln (' Yang anda ketik adalah u ');
              end;
        'e' : begin
                writeln (' Yang anda ketik adalah e ');
             end;
        'i' : begin
               writeln (' Yang anda ketik adalah i ');
              end;
        'q' : begin
                quit := true;
              end;
        else writeln (' Yang anda ketik adalah huruf mati ');
      end; (* case *)
   until (quit);
   writeln (' Akhir program... sebab anda mengetik q ');
end.
```

TABEL (ARRAY)

```
program TABEL;
(* File : TABEL.PAS *)
(* latihan array : mengisi dg assignment, menulis *)
VAR
(* Kamus *)
  i : integer;
  tab : array [1..10] of integer;
  N : integer;

begin
(* Program *)
N := 5;
writeln ( 'Isi dan print tabel: ');

 (* isi dengan assignment *)
    for i := 1 to N do
       begin tab [i] := i;
         tab [i] := i;
       end;

 (* traversal : print *)
    for i := 1 to N do
       begin
         writeln ( 'i= ',i, ' tab[i]= ', tab[i] );
       end;
end.
```

```
program TABSTRU;
(* File : TABSTRU.PAS *)
(* latihan array yang isinya struktur : mengisi dg assignment, menulis *)
TYPE Point = record
              X : integer; (* absis *)
              Y : integer; (* ordinat *)
              end;
VAR
  (* Kamus *)
    i : integer (* indeks tabel *);
    tabpoint : array [1..10] of Point;
    N : integer;

begin
(* Program *)
  N := 5;
  writeln ( ' Isi dan print tabel struktur: ');
```

```
  (* isi dengan assignment *)
    for i := 1 to N do
      begin
        tabpoint [i].X := i;
        tabpoint [i].Y := tabpoint [i].X
      end;
  (* traversal : print *)
    writeln ( '--------------------' );
    writeln ( '    I    X     Y    ' );
    writeln ( '--------------------' );
    for i := 1 to N do
    begin
        writeln ( i:5, tabpoint [i].X:5, tabpoint [i].Y:5);
    end;
    writeln ( '----------------------' );
end.
```

TABEL MULTI DIMENSI

```
program Tab2dim;
(* File : Tab2dim.pas *)
(* Tabel integer dua dimensi (matriks) *)
(* Kamus *)
type
(* Cara I : sebagai array dua dimensi *)
    MatInt = array [1..3, 1..3] of integer;
(* Cara 2 : sebagai array of array *)
    MatArr = array [1..3] of array [1..3] of integer;
var
   M1 : MatInt;
   MA1 : MatArr;
   i , j : Integer ;

(* Algoritma *)
begin
   writeln (' Array dua dimensi : ');

   (* Mengisi Matrik dua dimensi *)
   for i := 1 to 3 do
     begin
       for j := 1 to 3 do
           begin
             M1 [i,j] := i * j;
           end; (* for j *)
     end;

   (* Menulis hasil isian di atas *)
    for i := 1 to 3 do
    begin
       for j:= 1 to 3 do
       begin
      write (' (i, j) = ', i, ',', j, ' => M1 [i, j] = ', M1 [i, j]);
       end;
      writeln;
    end;

    writeln ( ' Array of array : ');

   (* Mengisi array of array : perhatikan cara mengacu elemen *)
    for i := 1 to 3 do
       begin
          for j := 1 to 3 do
```

```
            begin
               MA1 [i] [j] := i*j;
            end;
          end;

   (* Menulis hasil isian di atas *)
    for i := 1 to 3 do
       begin
          for j := 1 to 3 do
             begin

(* Cobalah dua instruksi write sbb *)
(* write (' (i,j) = ', i, ', ', j, ' => MA1 [i,j] = ',MA1 [i,j] ); *)

write ('(i,j) = ', i, ', ', j, ', => MA1 [i,j] = ',MA1 [i][j] );
            end;
      writeln;
     end;
end.
```

```
program Tab3dim;
(* File : Tab3dim.pas *)
(* Tabel integer tiga dimensi *)
(* Kamus *)
type
(* Cara I : sebagai array tiga dimensi *)
    MatInt = array [1..2, 1..2, 1..2] of integer;
(* Cara 2 : sebagai array of array of array *)
    MatArr = array [1..2] of array [1..2] of array [1..2] of integer;
var
   M1: MatInt;
   MA1: MatArr;
   i,j,k : Integer;

(* Algoritma *)
begin
   writeln (' Array tiga dimensi : ');
   (* Mengisi Matriks tiga dimensi *)
   for i := 1 to 2 do
   begin
      for j := 1 to 2 do
         begin
            for k := 1 to 2 do
               M1 [i,j,k] := i*j;
            end;
       end;
```

```
   (* Menulis hasil isian di atas *)

   for i := 1 to 2 do
      begin
         for i := 1 to 2 do
            begin
               for k := 1 to 2 do
writeln ('(i,j,k) = ', i, ', ', k, ' => M1 [i,j,k] = ', M1 [i,j,k]);
            end;
         end;
   writeln ( 'Array of array of array : ' );

   (* Mengisikan array of array : perhatikan cara mengacu elemen *)
   for i := 1 to 2 do
      begin
         for j := 1 to 2 do
            begin
               for k := 1 to 2 do
                  begin MA1 [i] [j] [k] := i*j; end;
                end;
            end;

   (* Menulis hasil isian di atas *)
   for i := 1 to 2 do
      begin
         for j := 1 to 2 do
             begin
                for k := 1 to 2 do
                   begin
writeln('(i,j,k =', i, ',', j,',',k,', =>MA1[i,j,k]=' MA1[i][j][k]);
                end;
             end;
        end;
   end.
```

RECORD VARIANT

```
program RecVar;
(* File : RecVar.pas *)
(* Record Varian : dengan komponen yang variabel *)
(* Konsep : representasi data yang tidak "fix" type-nya*)
(* Kamus *)
(* Cell adalah sebuah sel spread sheet, yang mungkin isinya :*)
(* Formula : string; integer atau real *)
type
   trec = (rumus, int, float);

   cell = record
            adrbrs : char;
            adrkol : integer;
             case Tsel : trec of
                       rumus : ( form : string [5]);
                       int       : ( nili : integer);
                       float    : ( nilf : real ) ;
         end; (* cell *)
var
      Fcell, ICell, RCell : Cell;

begin              (* Algoritma *)

(* Cara mengisi nilai *)
(* Type cell adalh formula *)
   Fcell.adrbrs := 'A' ;
   Fcell.adrkol := 1;
   Fcell.Tsel := rumus;
   Fcell.form := 'XYZ12';

(* Type cell adalah integer *)
   Icell.adrbrs := 'A';
   Icell.adrkol := 2;
   Icell.Tsel := int;
   Icell.form := '10';

(* Type cell adalah bilangan *)
   Rcell.adrbrs := 'A';
   Rcell.adrkol := 3;
   Rcell.Tsel := float;
   Rcell.nilf := 10.55;
end.
```

```
program RecVarx;
(* File : RecVarx.pas *)
(* Record Varian dengan type bentukan *)
(* Kamus *)
(* Gambar adalah bentuk ayang dapat berupa garis, segi empat *)
type
    trec = ( garis, segi4 );
     Point = record
               x: integer;
               y: integer;
             end;
     TGrs = record
             Pawal : Point; (* titik awal *)
             PAkhir : Point (* titik akhir *)
            end;
      TS4 = record  (* Segi empat *)
               TopLeft : Point; (* Kiri atas *)
               BottRight : Point (* Kanan bawah *)
            end;
      Gambar =  record
                  id : integer; (* identitas gambar *)
                  case TBentuk : trec of
                              garis : (G : TGrs);
                              segi4 : (S4 : TS4);
                end;
var
  G1, G2 : Gambar;
  G3 : Gambar;

begin            (* Algoritma *)

(* Cara mengisi nilai *)
(* Gambar adalah garis *)
G1.id := 1;
G1.TBentuk := garis;
G1.G.PAwal.x := 10;
G1.G.PAwal.x :=10;
G1.G.PAkhir.x :=10;
G1.G.PAkhir.x :=10;

(* Gambar adalah segiempat *)
G2.id := 99;
G2.TBentuk := segi4;
G2.S4.TopLeft.x := 0;
G2.S4.TopLeft.x :=0;
G2.S4.BottRight.x :=10;
G2.S4.BottRight.x :=10;
```

```
(******* HATI - HATI *******)
(* Perhatikan apa yang terjadi saat kompilasi *)
(* dengan assignment berikut *)

G3.id :=99;
G3.TBentuk :=garis;
G3.S4.TopLeft.x := 0;
G3.S4.TopLeft.x :=0;
G3.S4.BottRight.x :=10;
G3.S4.BottRight.x :=10;

(* Komentar anda ???*)

end.
```

POINTER

```
program Ptint;
(* File : Ptint.pas *)
(* Pointer ke integer *)
(* Kamus *)
var
  i : integer;
  Pti : ^integer ;

Begin (* Algoritma *)
  i := 5;
  new (Pti); (* alokasi *)
  Pti^ := 10;
  writeln ('i = : ', i );
  writeln ( 'Nilai yang ditunjuk Pti = ', Pti^ );
  dispose (Pti); (* dealokasi *)
end.
```

```
program PTab;
(* File : PTab.pas *)
(* Pointer ke tabel integer *)
(* Kamus *)
   type TabInt = array [1..10] of integer;
var
   T : TabInt;
   i  : Integer;
   pt: ^TabInt;

begin (* Algoritma *)
   for i := 1 to 10 do
      begin
        T [i] :=i;
   end;

   (* alokasi Pointer ke tabel integer *)
    new (pt) ;  (* alokasi *)
    Pt^ := T;  (* pendefinisian isi/nilai yang ditunjuk *)

   (* akses elemen *)
    for i := 1 to 10 do
       begin
         writeln ( 'i= ', i, ' pt&[i] = ', pt^[i] );
       end;
     dispose (pt);
end.
```

```
program PRec;
(* File : PRec.pas *)
(* Pointer ke record *)
(* Kamus *)
type
   Point = record
              x: integer;
              y: integer;
           end;
var
   T : Point ;
   Pt : ^Point;

begin (* Algoritma *)
   new (Pt); (* alokasi *)
   Pt^.x := 10;  Pt^.y := 5;     (* akses komponen titik *)
   with Pt^  do
      begin    (* dengan WITH *)
         writeln ( 'Absis P : ', x);
         writeln ( 'Ordinat P : ', y);
      end; (* kenapa tidak dilakakan dispose ? *)
end.
```

LIST LINIER SEDERHANA

```
program list;
(* File ; list.pas *)
(* contoh deklarasi list dan pengisian nilai Info *)
TYPE
     infotype = integer;
     address = ^ ElmtList; (* alamat elemen *)
     Elmtlist = record  (* type elemen *)
                   info : infotype;
                   next : address;
                 end;
VAR
    First : address;
    P      : address;

begin

(* Program *)
(* GetMem (First, Sizeof (ElmtList )) : *)

    First :=nil;
    GetMem (P, Sizeof (Elmtlist));
    P^.Info := 10;
    P^.Next := NIL;
    First := P;
    with First^ do begin
      writeln  ( 'Info : ', info );
    end;
    dispose (p);
end.
```

FILE EKSTERNAL

```
program BacaText;
(* File : Bacatext.pas *)
(* Membaca sebuah text file diakhiri '.' ,dan menuliskan apa adanya ke layar *)
(* Program ini tidak memanfaatkan EOF. *)
(* Jadi tidak boleh ada file yang hanya mengandung EOF.*)
(* File kosong berisi sebuah karakter '.' *)
(* Kamus *)
var
  f : Text;
  CC : char; (* karalter yang dibaca*)

(* Algoritma *)
begin
  Assign (f,'pitakar.txt');
  Reset (f); (* Buka dengan modus Read Only *)
  if (CC = '.') then
    begin
      writeln ( 'Arsip Kosong ' );
    end else (* CC bukan '.' *)
   begin
      repeat
         read (f,CC);
          write (CC);
      until (CC = '.') ;
   end;
   Close (f) ;
end.
```

```
(* File : Rekamtxt.pas *)
program RekamText;
(* Membaca karakter demi karakter dari keyboard, *)
(* dan menyimpan ke text file. *)
(* Akhiri pembacaan dengan "#" File diakhiri EOF *)
(* Kamus *)
var
  f : text;
  CC : char;

(* Algoritma *)
begin
  Assign (f, 'TextOut.TXT');
  Rewrite (f); (* Buka dengan modus rekam *)
  write ( 'Masukan karakter, akhiri dengan # '); read (CC);
  while (CC <> '#' ) do
```

```
    begin
      Write (f,CC);
      read (CC);
    end; (* CC= '#' *)
   Close (f);
end.
```

```
(* File : RWInt.pas *)
program TulisInt;
(* Tahap I : Membaca (angka integer) dari keyboard,*)
(* dan menyimpan ke file. Akhiri pembacaan dengan 999 *)
(* Tahap II : membaca dan menulis ke layar, hasil dari penulisan Tahap I *)
(* Kamus *)
var
  f : file of integer;
  I : integer;

(* Algoritma *)
begin
(* Tahap I : Pembuatan/penulisan file *)
  Assign ( f, 'Myint.dat' );
  Rewrite (f);
  writeln ( 'Input angka yang disimpan, akhiri dengan 999' );
  write ( 'Nilai int : ' ); readln (I);
  while (I <> 999) do
   begin
     Write (f,I);
     write ( 'Nilai Int : ' ); readln (I);
   end;
  Close (f); (* File diakhiri dengan EOF, angka 999 tidak direkam *)
(* Tahap II : Pembacaan file hasil untuk ditulis *)
  Reset (f);
  if not eof (f) then
  begin
    repeat
       read (f,I);
       Writeln (I:4);
   until eof (f);
  end
  else (* eof *)
    begin
         writeln ( 'File kosong' );
    end;
  Close (f);
  (*Cobalah membuka myint.dat dengan editor text. Apa yang terjadi?*)
end.
```

```
program Frec;
(* File : Frec.pas *)
(* Membaca sebuah file of record, dan menuliskan isinya ke layar *)
(* Kamus *)
type
   Point = record
             x : integer;
             y : integer;
            end;
var
  f : file of Point;
  T : Point; (* karakter yang dibaca *)

(* Algoritma *)
begin
(* Tahap I : membaca data titik untuk direkam *)
  Assign (f, 'mytitik.dat');
  rewrite (f);
  writeln ( 'Input titik, diskhiri dengan x=-999 dan y=-999 ');
(* Baca nilai titk yang akan direkam *)
  write ( 'Absis = ' ); readln (T.x);
  write ( 'Ordinat = ' ); readln (T.y);
  while (T.x <> -999) and (T.y <> -999) do
    begin
(* rekam ke file *)
      write (f,T);
(* Baca data berikutnya *)
      write ( 'Absis    = ' ); readln (T.x);
      write ( 'Ordinat = ' ); readln (T.y);
    end;
  Close (f);
(* Tahap II : menuliskan hasil rekaman ke layar *)
Reset (f);
if eof (f) then
   begin
     writeln ( 'Arsip Kososng ' );
   end else
   begin
     repeat
        read (f,T);
(* Tulis data titik *)
        writeln ( 'T (x,y) = ( ', T.x, ', ', T.y, ' ) ');
      until eof (f) ;
   end;
 Close (f);
end.
```

Unit dalam Turbo Pascal (dan FreePascal)

Catatan penting :

1. Unit bukan merupakan Pascal standard, bahkan "berlawanan" dengan konsep bahasa Pscal yang "nested", berstuktur blok.
2. Unit baru diajar ketika siswa sudah memprogram dalam skala "agak besar", dan kompleks, dimana sebuah program utuh dipecah-pecah menjadi beberapa file supaya implementasinya dapat dilakukan oleh lebih dari satu programmer
3. Bagian ini adalah bagian penutup (rangkuman dari berbagai konsep kecil yang pernah dipelajari), yang tidak diajarkan jika tidak ada waktu.
4. Unit ini harus diajarkan jika Pascal dipakai sebagai bahasa untuk memprogram pelajaran struktur data. Unit dipakai misalnya untuk merepresentasi ADT. Atau mesin
5. Setiap unit harus mempunyai "driver", yaitu main program yang dipakai untuk mentest semua primitif yang ditulis dalam unit.
6. Implementasi sebuah program yang dibagi-bagi menjadi unit, dengan banyak primitif, harus dilakukan secara incremental. (Nomor tahapan dalam angka romawi) yang ada pada contoh ini menunjukkan urutan pengerjaan / implementasi kode pada unit

```
(* file   : upoint.pas                              *)
(* pemrogram     : Inggriani                 *)
(* tanggal       : 11 Oktober 2000  *)
(* deskripsi     : unit,isinya definisi type point& *)
(*                   beberapa primitifuntuk manipulasi typepoint*)
unit upoint;

interface
Const
    Maxpoint=8;

(* Kamus *)
type Point = record
              x:integer; (* absis*)
              y:integer; (* ordinat*)
             end;
var
  TabPoint : array [1..MaxPoint] of Point;
  Npoint : integer ; (* banyaknya elemen array 1 s/d 8*)
  F_point : file of Point;
```

```
Procedure MakePOint (x,y:integer; var P:Point);
(* Membentuk Point dari x dan y*)

Procedure TulisPOint (P:Point);
(* Menulis nilai sebuah point dengan format *)
(* (x,y)  *)

function EqPoint (P1,P2:Point):boolean;
(* mengirim true jika P1=P2; *)
(* yaitu absisnya sama dan ordinatnya sama *)

(******* prosedur manipulasi array TabPoint *******)

procedure BuatTabKosong;
(* membuat tabel kosong, supaya bisa ditambah dg AddElmt*)

procedure AddElmt(P:Point);
(* menambahkan P sebagai elemen tabPoint, N bertambah 1 *)
(* I.S. tabel kosong sudah ada *)

procedure TulisTab;
(* menuliskan isi tabel*)

(******* prosedur memindahkan  array TabPoint ke file*******)

procedure TabToFile;
(* memindah isi tabel TabPOint ke f_point*)

(******* prosedur memindahkan  isi file f_point ke *)
(* array TabPoint *******)

procedure FileToTab;
(* memindah isi tabel TabPOint ke f_point*)

(******* prosedur untuk menulis isi file *)

procedure TulisFile;
(* menulis isi file f_point*)

Implementation
Procedure MakePOint (x,y:integer; var P:Point);
(* Membentuk Point dari x dan y*)
begin
  P.x:=x;
  P.y := y;
end;
```

```
Procedure TulisPOint (P:Point);
(* Menulis nilai sebuah point dengan format *)
(* (x,y)  *)
begin
  writeln ('(',P.x, ',', P.y, ')');
end;

function EqPoint (P1,P2:Point):boolean;
(* mengirim true jika P1=P2; *)
(* yaitu absisnya sama dan ordinatnya sama *)
begin
  EqPOint:= (P1.x=P2.x) and (P1.y=P2.y)
end;


(******* prosedur manipulasi array TabPoint *******)

procedure BuatTabKosong;
(* membuat tabel kosong, supaya bisa ditambah dg AddElmt*)
begin
  NPoint:=0;
end;

procedure AddElmt(P:Point);
(* menambahkan P sebagai elemen tabPoint, N bertambah 1 *)
(* I.S. tabel kosong sudah ada, N pasti < MaxPOint *)
begin
  NPoint := NPoint + 1;
  TabPoint[NPoint] := P;
end;

procedure TulisTab;
(* menulis isi tabel *)
(* Jika tabel kosong, menulis pesan: tabel kosong*)
VAR
  i:integer;
begin
  if (NPoint=0) then
  begin
     writeln('Tabelnya kosong');
  end else
  begin
    writeln ('Isi tabel');
    for i:=1 to NPOint do
    begin
       TulisPoint(tabPoint[i]);
```

```
    end;
  end;

end;

(******* prosedur memindahkan  array TabPoint ke file*******)

procedure TabToFile;
(* memindah isi tabel TabPOint ke f_point*)
(* nama filenya fpoint.dat *)

(* kamus lokal *)
VAR
  I:integer;

begin
   assign (F_point, 'fpoint.dat');
   rewrite (F_Point);
   I := 1;
   While (I<=NPoint ) do
   Begin
      Write(F_point, TabPoint[I]);
      I := I + 1;
   End;
   Close (F_point);
end;


(******* prosedur memindahkan  isi file f_point ke *)
(* array TabPoint *******)
procedure FileToTab;
(* memindah isi tabel TabPOint ke f_point*)
(* file mungkin kosong, menghasilkan tabel kosong *)
(* kamus lokal *)
VAR
  P:Point;

begin
   NPoint := 0;
   Assign (F_Point, 'fpoint.dat');
   reset (F_Point);
   NPoint := 0;
   if eof(F_Point) then
   begin
      writeln ('File kosong');
   end else
   Begin
```

```
    writeln('isi file adalah : ');
    Read(F_Point,P);
    TabPoint[NPoint] := P;
   End;
   Close (F_Point);

end;


procedure TulisFile;
VAR
P: POint;
begin
    assign (f_point, 'fpoint.dat');
    reset(f_point);
    if eof(f_point) then
    begin
      writeln ('file kosong');
    end else
    begin
      writeln ('Isi file ');
      repeat
        read(f_point, P);
        TulisPOint(P);
      until eof (f_point);
      close (f_point);
    end;
end;

begin

end.
```

```
(* file   : mainpoint.pas                                  *)
(* pemrogram     : Inggriani                  *)
(* tanggal       : 11 Oktober 2000  *)
(* deskripsi     : main program untuk mentest upoint       *)

program mainpoint;
uses upoint;
(* Kamus *)

var
  P:Point;

(* fase III *)
  P1:Point;
```

```
(* fase VI *)
  i: integer ;

begin (* algoritma progam utama *)
 writeln('main');

(* fase I*)
 MakePoint (5,5,P);

(* Fase II: kode TulisPOint *)
 TulisPoint (P);

(* fase III: setelah EqPOint selesai *)
(* tambahkan deklarasi P1 *)
 MakePOint(0,0,P);
 if EQPOint(P,P1) then
 begin
   writeln('P=P1');
 end
 else begin
   writeln('P tidak sama dnegan P1');
 end;

(* fase IV : buat tabel kosong *)
  BuatTabKosong;

(* fase V: tulis ke file, hasilnya file kosong *)
  TabToFile;

(* Fase VI : akan muncul tulisan tabel kosong *)
  FiletoTab;

(* Fase VII: isi tab dg 5 elemen. Tidak apa-apa isinya semua sama! *)
  for i:= 1 to 5 do
  begin
    AddElmt(P);
  end;
 TulisTab;

(* fase VIII : simpan ke file *)
 TabToFile;

 (* Fase IX *)
 TulisFile;
end. (* main *)
```

Index

I

K

L

M

N

O

P

R

S

T

U

V

W

BIODATA SINGKAT PENULIS

DR. Ir. M.M. Inggriani Liem, menjadi staf Pusat Komputer ITB 1977 hingga tahun 1981. Sejak tahun 1978 juga merangkap sebagai pengelola Pendidikan Ahli Teknik Jurusan Penggunaan Komputer ITB (PAT JPK) yang didirikan sebagai cikal bakal Departemen Teknik IF ITB (sekarang, PAT menjadi Politeknik).

Pada tahun 1981, status kepegawaiannya dipindahkan ke Departemen Teknik Informatika ITB ketika Departemen tersebut didirikan. Menjadi pengajar programming sejak Departemen Teknik Informatika didirikan, sampai sekarang.

Pernah mengambil studi lanjut di Prancis, di bidang informatika keahlian programming.