MODUL PRAKTIKUM

dasar algoritme dan pemrograman

KUG1D1

IF LAB

Gedung F Lt. 3 IFLAB1-IFLAB4
School of Computing
Telkom University

Telkom University

informatics lab

Lembar Pengesahan

Saya yang bertanda tangan di bawah ini:

Nama : Dade Nurjanah, Ph.D
NIP :
Dosen PJMP : Dasar Algoritma dan Pemrograman
KK : ICM

Menerangkan dengan sesungguhnya bahwa modul ini digunakan untuk pelaksanaan praktikum di semester genap tahun ajaran 2015/2016 di Informatics Lab Fakultas Informatika Universitas Telkom.

Bandung, Januari 2016

Menyetujui,

Febryanti Sthevanie, S.T., M.T.	Rimba Whidiana Ciptasari, Ph.D	Rita Rismala, S.T., M.T.
Yanti Rusmawati, Ph.D	Mohamad Syahrul Mubarok	Siti Sa'adah, S.T., M.T.,
Dody Qori Utama, S.T., M.T.	Said Al Faraby, S.T., M.Sc	Untari Novia Wisesty, S.T., M.T.

Mengesahkan,

Dosen PJMP
Dasar Algoritma dan Pemrograman

Dade Nurjanah, Ph.D

Mengetahui,

Kaprodi S-1 Fakultas Informatika

Moch. Arif Bijaksana, Ph.D

Peraturan Praktikum
Laboratorium Informatika 2015/2016

1. Praktikum diampu oleh dosen kelas dan dibantu oleh asisten laboratorium dan asisten praktikum.
2. Praktikum dilaksanakan di Gedung Kultubai Selatan (IFLAB 1 s/d IFLAB 5) sesuai jadwal yang ditentukan.
3. Praktikan wajib membawa modul praktikum, kartu praktikum, dan alat tulis.
4. Praktikan wajib mengisi daftar hadir *rooster* dan BAP praktikum dengan bolpoin bertinta hitam.
5. Durasi kegiatan praktikum S-1 = 3 jam (150 menit).
   a. 15 menit untuk pengerjaan Tes Awal atau wawancara Tugas Pendahuluan
   b. 45 menit untuk penyampaian materi
   c. 90 menit untuk pengerjaan jurnal dan tes akhir
6. Jumlah pertemuan praktikum:
   - 10 kali di lab (praktikum rutin)
   - 3 kali di luar lab (terkait Tugas Besar dan UAS)
   - 1 kali berupa presentasi Tugas Besar atau pelaksanaan UAS
7. Praktikan wajib hadir minimal 75% dari seluruh pertemuan praktikum di lab. Jika total kehadiran kurang dari 75% maka nilai UAS/ Tugas Besar = 0.
8. Praktikan yang datang terlambat :
   - <= 30 menit : diperbolehkan mengikuti praktikum tanpa tambahan waktu Tes Awal
   - > 30 menit : tidak diperbolehkan mengikuti praktikum
9. Saat praktikum berlangsung, asisten praktikum dan praktikan:
   - Wajib menggunakan seragam sesuai aturan institusi.
   - Wajib mematikan/ mengkondisikan semua alat komunikasi.
   - Dilarang membuka aplikasi yang tidak berhubungan dengan praktikum yang berlangsung.
   - Dilarang mengubah pengaturan *software* maupun *hardware* komputer tanpa ijin.
   - Dilarang membawa makanan maupun minuman di ruang praktikum.
   - Dilarang memberikan jawaban ke praktikan lain.
   - Dilarang menyebarkan soal praktikum.
   - Dilarang membuang sampah di ruangan praktikum.
   - Wajib meletakkan alas kaki dengan rapi pada tempat yang telah disediakan.
10. Setiap praktikan dapat mengikuti praktikum susulan maksimal dua modul untuk satu mata kuliah praktikum.
    - Praktikan yang dapat mengikuti praktikum susulan hanyalah praktikan yang memenuhi syarat sesuai ketentuan institusi, yaitu: sakit (dibuktikan dengan surat keterangan medis), tugas dari institusi (dibuktikan dengan surat dinas atau dispensasi dari institusi), atau mendapat musibah atau kedukaan (menunjukkan surat keterangan dari orangtua/wali mahasiswa.)
    - Persyaratan untuk praktikum susulan diserahkan sesegera mungkin kepada asisten laboratorium untuk keperluan administrasi.
    - Praktikan yang diijinkan menjadi peserta praktikum susulan ditetapkan oleh Asman Lab dan Bengkel Informatika dan tidak dapat diganggu gugat.
11. Pelanggaran terhadap peraturan praktikum akan ditindak secara tegas secara berjenjang di lingkup Kelas, Laboratorium, Fakultas, hingga Universitas.

Daftar Isi

Modul 1 Pengenalan Pascal

Tujuan
Praktikan diharapkan mampu : o Memahami tentang algoritma dan cara kerjanya. o Menggunakan bahasa Pascal untuk implementasi algoritma o Menggunakan *free Pascal*: *compile, run*, dan lain-lain

1.1 Pengertian Algoritma

Algoritma adalah urutan langkah-langkah untuk memecahkan masalah. Terdapat beberapa definisi lain dari algoritma yang dikutip dari berbagai literatur, antara lain:

- **Algoritma** adalah deretan langkah-langkah komputasi yang mentransformasikan data masukan menjadi keluaran.[COR92]
- **Algoritma** adalah deretan instruksi yang jelas untuk memecahkan masalah, yaitu untuk memperoleh keluaran yang diinginkan dari suatu masukan dalam jumlah waktu yang terbatas. [LEV03]
- **Algoritma** adalah prosedur komputasi yang terdefinisi dengan baik yang menggunakan beberapa nilai sebagai masukan dan menghasilkan beberapa nilai yang disebut keluaran. Jadi, algoritma adalah deretan langkah komputasi yang mentransformasikan masukan menjadi keluaran. [COR89]

*untuk lebih jelasnya silahkan buka buku "ALGORITMA DAN PEMROGRAMAN" hal 3.

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 dengan program. 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 (*game*), 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.

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).

Dalam membuat program, ada beberapa hal yang harus diketahui sebagai tahap awal atau konsep dasar dalam pemrograman. Konsep-konsep dasar tersebut antara lain kalkulasi (operasi aritmatika dasar maupun lanjut), kontrol blok program (dalam bentuk *sequence*, percabangan, perulangan), komunikasi (input/output), dan jumlah memori yang digunakan (variabel, list, *record*).

Dalam praktikum ini, bahasa pemrograman yang digunakan sebagai standar adalah bahasa pemrograman Pascal. Keunggulan bahasa Pascal adalah keteraturan dalam pembuatan program dan kelengkapan struktur data.

Dalam pemrograman Pascal, pokok dari pemrograman tersebut ada 3, yaitu:
1. Nama program
2. Kamus/pendefinisian variable
3. Pada bagian ini dilakukan pendefinisian variabel. Variabel adalah tempat menyimpan data.
4. Algoritma
5. Pada bagian ini notasi algoritma dituliskan. Notasi algoritma yang ditulis, akan dilaksanakan secara berurutan

1.2 Pengenalan IDE Free Pascal

Free Pascal adalah *compiler* untuk bahasa Pascal. *Free Pascal* ini didistribusikan secara gratis di bawah lisensi *GNU Public*. Versi terakhir *Free Pascal* (untuk berbagai platform) serta cara instalasinya dapat di unduh di situs *http://www.freePascal.org/*.

1.2.1 Menulis Program

Buka *Free Pascal* yang telah ter-*install*. Untuk membuat program. Anda dapat memulai membuat program dengan cara sebagai berikut:

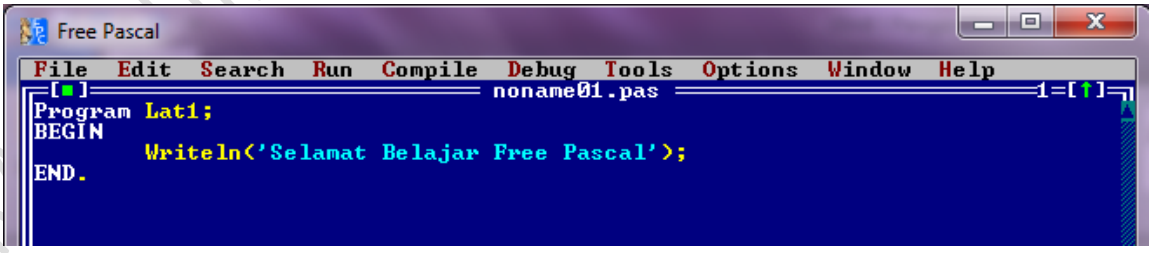
1. Pilih menu File(Alt+F),
2. Pilih New.

Setelah itu akan muncul jendela baru. Silahkan tuliskan program yang ingin anda tulis. Untuk mencoba anda dapat menulis program berikut persis seperti apa yang tertulis di buku:

Kode Bahasa Pascal

```
program Lat1;
begin
    writeln('Selamat Belajar Free Pascal');

end.
```

Gambar berikut akan menunjukkan isi editor setelah program diatas di ketik:

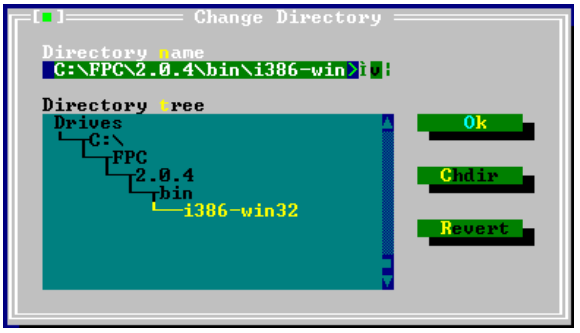


1.2.2 Menyimpan Program

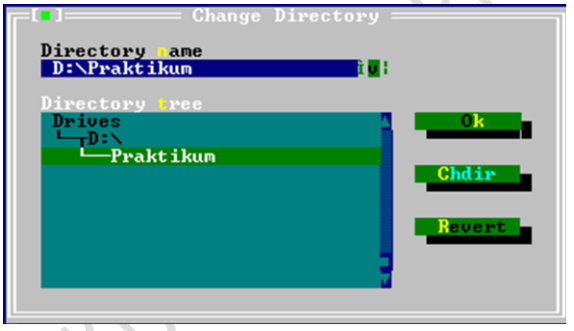
Program yang telah dibuat atau dimodifikasi sebaiknya disimpan ke *disk* sebelum dijalankan. Langkah ini perlu dilakukan agar program yang baru saja ditulis tidak hilang bila komputer mendadak mati pada saat program dieksekusi atau karena sebab-sebab lain.

Sebelum kita simpan, kita harus mengganti direktori untuk menyimpan atau membuka program. Sebaiknya anda membuat 1 folder untuk menempatkan hasil kerja anda. Caranya adalah sebagai berikut:

1. Pilih menu File,
2. Pilih Change Dir ...
3. Ubah menuju ke dalam folder yang telah anda buat (mis: D:/Praktikum),
4. Klik Change Dir.



Sebelum di Change Dir



Sesudah di Change Dir

Cara menyimpan program adalah sebagai berikut:

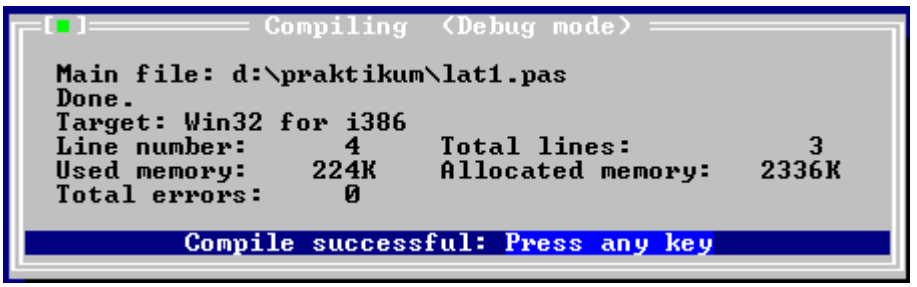
1. Pilih menu File,
2. Pilih File Save.
3. Masukkan nama yang sesuai dengan program.

1.2.3 Menjalankan/ Mengeksekusi Program

Untuk menjalankan program terlebih dahulu kita harus melakukan proses *compile* terhadap program. Proses *compile* adalah proses kompilasi terhadap program yang telah dibuat, dengan kata lain proses untuk merubah program yang telah dibuat menjadi sebuah aplikasi yang dapat dieksekusi. Proses kompilasi ini juga untuk mengecek apakah terdapat kesalahan dalam penulisan program. Adapun caranya adalah dengan:

1. Pilih menu Compile;
2. Pilih Compile

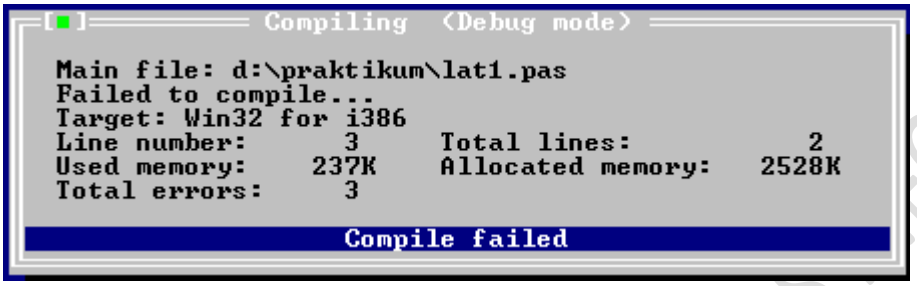
atau bisa disederhanakan dengan (Alt+F9). Apabila program yang ditulis benar akan muncul notifikasi sebagai berikut.

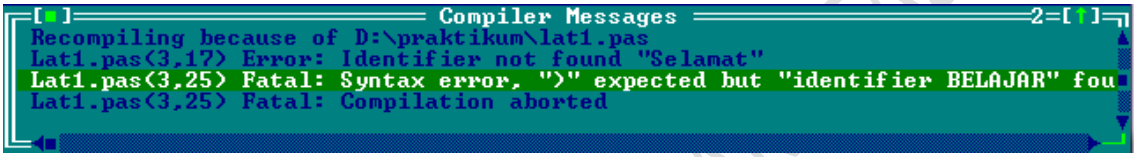


Pada layar diatas terdapat informasi yang diantaranya menyatakan bahwa:

- Jumlah memori yang digunakan adalah sebesar 224K,
- Jumlah baris program sebanyak 3 Baris.

Tetapi bila ada kesalahan dalam penulisan program. Misalnya kita hilangkan tanda " ' " pada awal penulisan `writeln`. Maka akan tampil sebagai berikut.





Kesalahan pada program, terdapat 2 jenis, yang pertama kesalahan sintaks atau penulisan program seperti yang telah dijelaskan di atas. Yang kedua, yaitu kesalahan pada logika pemrograman/semantik (misal program yang melakukan *looping* terus-menerus atau menampilkan keluaran yang tidak sesuai dengan yang diinginkan).

Anda dapat mengkompilasi suatu program menjadi file yang executable (dapat dieksekusi secara langsung dari prompt DOS) dengan cara compile seperti ini. Jadi setelah dilakukan Compile maka akan terdapat file executable pada folder tempat anda menyimpan program.

Apabila program sudah benar. Maka kita akan lanjut ke dalam tahap eksekusi program. Untuk menjalankan program yang telah berada pada editor, lakukan langkah berikut:

1. Pilih menu Run,
2. Pilih Run.

Kedua cara tersebut dapat disingkat menjadi Ctrl+F9. Setelah itu pada layar akan tampak output:

```
Selamat Belajar Free Pascal.
```

Komentar pada Program

Dalam pembuatan program, ada kalanya kita ingin memberikan sebuah komentar pada program untuk suatu tujuan tertentu, misal menjadi pengingat kegunaan sebuah fungsi atau blok program, memberikan penjelasan tentang program yang telah dibuat (dengan harapan dapat membantu orang lain yang membaca program tersebut) atau sebagai dokumentasi dari program yang telah dibuat (memberikan nama *programmer*, tempat dan tanggal program dibuat, deskripsi singkat program, dsb).

Komentar komentar yang diberikan pada program ini, nantinya tidak akan di proses pada saat proses kompilasi (diabaikan). Dalam Bahasa pemrograman *Pascal* terdapat 2 cara untuk memberikan komentar pada program, yaitu :

- Jika ingin memberikan komentar pada suatu baris tertentu, dapat memberikan tanda "//" pada awal baris pada suatu program
- Jika ingin memberikan komentar lebih dari 1 baris (berurutan), dapat memberikan tanda "{" pada baris pertama komentar dan tanda "}" pada baris akhir komentar

Tips menulis program yang baik

- Cara menulis program dengan diketik agak ke dalam, disebut dengan ***indentasi***. Cara ini berguna untuk memudahkan membaca blok-blok perintah program, terutama pada program yang memiliki banyak struktur subblok
- Berikan komentar secukupnya. Orang lain yang membaca program tersebut akan sangat terbantu pada saat membacanya. Demikian juga diri sendiri pun akan sangat terbantu, pada saat membaca ulang program tersebut di masa yang akan dating
- Jangan berlebihan dan bertele-tele dalam memberikan komentar! Usahakan singkat, padat, tepat dan jelas (kecuali, apabila komentar tersebut dipakai untuk keperluan pengajaran/tutorial)
- Berikan dokumentasi berupa penjelasan singkat tentang maksud program yang dibuat, tanggal pembuatan, identitas *coder*, ataupun instansi yang berkaitan

Soal

Bagian I

1. Dapatkah suatu program dikompilasi tanpa dijalankan? Bagaimana caranya?
2. Apa fungsi dari tombol Alt+X?
3. Mengapa anda sebaiknya menyimpan program anda sebelum melakukan eksekusi?
4. Mungkinkah suatu program yang baru dibuat dijalankan tanpa dikompilasi?

Bagian II

Perhatikan program berikut ini.

```
Program IFLAB;
uses crt; {*}
TYPE karyawan = record nama : String;
jml_lembur,gol : integer;
      gaji_pokok,bonus_tetap,bonus_lembur,gaji_bulan,gaji_total: real;
end;

var
    a         :
karyawan begin
clrscr;  {**}
   write('Masukkan nama karyawan : '); readln(a.nama);
   writeln('Golongan  Karyawan  :  ');
   writeln('        1. Golongan  I  ');
   writeln('        2. Golongan II  ');
   writeln('        3. Golongan III ');
   write('Masukkan  golongan  karyawan  :  ');  readln(a.gol);
   write('Masukkan jumlah lembur : '); readln(a.jml_lembur);
   case a.gol of
        1 : begin
              a.gaji_pokok := 250000;
            end;
        2 : begin
              a.gaji_pokok := 500000;
            end;
        3 : begin
              a.gaji_pokok := 750000;
            end;
    a.bonus_tetap := 0.25*a.gaji_pokok;
    a.bonus_lembur := a.jml_lembur*a.bonus_tetap;
    a.gaji_bulan := a.gaji_pokok+a.bonus_tetap+a.bonus_lembur;
    writeln('---Jumlah Gaji---');
    writeln('Nama Karyawan          : ',a.nama);
```

```
    writeln('Golongan Karyawan      : ',a.gol);
    writeln('Jumlah lembur          : ',a.jml_lembur);
    writeln('Total Gaji per bulan   : Rp. ',a.gaji_bulan:0:0);
    readln;
end.
```

a. Salin potongan program tersebut ke aplikasi Pascal!

b. Bila program tersebut di-compile apa yang terjadi? Sudah benarkah program tersebut? Jika belum perbaiki program tersebut hingga benar!

c. Hapus sintaks yang ditandai dengan "{**}", apa yang terjadi? Apa pula yang terjadi

d. bila sintaks yang ditandai dengan "{*}" yang dihapus?

e. Simpan program tersebut ke direktori masing-masing!

f. Jalankan program tersebut dan jelaskan maksud program tersebut!

Shortcut Key

1. Untuk membuka file (open), atau bahkan untuk membuat file baru, tekan F3. Ketikkan nama file yang ingin dibuka, bila file tersebut tidak ditemukan, FPC akan membuat file baru dengan nama tersebut.
2. Untuk menyimpan file (save) secara cepat, tekan tombol F2
3. Untuk meng-compile dan me-run program sekaligus tekan tombol CTRL+F9.
4. Untuk melihat output program di layar dos, tekan ALT+F5
5. Untuk change directory dengan cepat gunakan kombinasi tombol ALT+F kemudian C.
6. Untuk berpindah ke file lain yang sedang dibuka, tekan F6
7. CTRL+C dan CTRL+V tidak berfungsi di IDE FPC, gunakan kombinasi :
    a. CTRL+Insert untuk Copy b.
    SHIFT+Insert untuk Paste c.
    SHIFT+Delete untuk Cut

8. Hafalkan shortcut-shortcut diatas! Semakin jarang anda menggunakan mouse, semakin cepat anda mengetik sebuah program. Memindahkan tangan dari keyboard ke mouse untuk meng-klik File kemudian meng-klik Save akan makan waktu jauh lebih banyak dibandingkan dengan menekan tombol F2! Berikut adalah beberapa shortcut untuk meng-edit text (terutama untuk mem-block / men-select text) tanpa bantuan mouse :

   a. Undo : Alt + Backspace
   b. Mengawali blok Ctrl + K B
   c. Mengakhiri blok Ctrl + K K
   d. Menghilangkan blok Ctrl + K Y
   e. Menambah blok satu karakter ke kiri Shift + Panah kiri
   f. Menambah blok satu karakter ke kanan Shift + Panah kanan
   g. Menambah blok ke akhir baris Shift + End
   h. Menambah blok ke baris di atasnya Shift + Panah atas
   i. Menambah blok ke baris di bawahnya Shift + Panah bawah
   j. Menambah blok satu kata di kiri Ctrl + Shift + Panah kiri
   k. Menambah blok satu kata di kanan Ctrl + Shift + Panah kanan
   l. Menambah blok satu halaman ke atas Shift + Page Up
   m. Menambah blok satu halaman ke bawah Shift + Page Down
   n. Menambah blok hingga ke awal program Ctrl + Shift + Home

Modul 2 Variabel, *Record*, I/O, *Assignment* dan Operator

Tujuan
Praktikan diharapkan dapat : o Mengetahui tipe primitif maupun tipe bentukan dalam Pascal. o Mengetahui instruksi masukan dan assignment. o Memahami ekspresi dan operator. o Membuat program yang dapat dieksekusi. o Mengetahui prosedur standar dalam Pascal.

2.1 Variabel

Variabel dapat didefinisikan sebagai sesuatu yang memiliki nilai dan tipe data yang nilainya dapat berubah-ubah. Penamaan variabel ini memiliki aturan. Aturan penamaan tersebut adalah sebagai berikut:

- Maksimal 32 karakter.
- Hanya boleh terdiri dari karakter huruf abjad (A-Z, a-z), angka (0-9), dan garis bawah / *underscore* (_).
- Harus diawali dengan karakter huruf abjad.
- Tidak boleh sama dengan variabel lain.
- Tidak boleh sama dengan *keyword* (kata kunci), seperti **program**, **var**, **begin**, **end, if, case**, dan lain-lain

2.2 Input / Output

Perintah input dalam Pascal ada 2 macam yaitu: *read* dan *readln*. *readln* digunakan untuk memasukkan data perbaris, artinya setelah tombol Enter ditekan, maka akan ganti baris. Perintah *read* tidak ganti baris, masih dalam baris yang sama.

Perintah output dalam Pascal ada 2 macam yaitu: *write* dan *writeln*. *write* akan mengoutputkan data dalam satu baris. *writeln* akan mengoutputkan data lalu posisi kursor akan berganti baris.

Contoh Input / Output:

Pseudo Code	Kode Bahasa Pascal	Output
Program baca1; kamus a: integer; Algoritma input(a); output(a)	(*deskripsi : contoh membaca dengan readln kemudian menuliskan yang dibaca*) Program baca1; Var a: integer; Begin writeln('contoh membaca dan menulis, ketik nilai integer'); readln(a); writeln('nilai yang dibaca : ',a); readln; End.	Contoh membaca dan menulis, ketik nilai integer 6 nilai yang dibaca : 6

2.3 **Tipe** Data

Seperti yang telah disebutkan pada subbab 2.1, dalam pemrograman, ada yang disebut dengan variabel, yaitu sebuah nilai dengan tipe data tertentu yang nilainya dapat berubah-ubah. Variabel ini memiliki tipe data tertentu yang sudah didefinisikan sebelumnya. Variabel yang sudah didefinisikan dengan sebuah tipe data tertentu hanya dapat dimasukkan nilai dengan tipe data tersebut saja. Contoh, variabel abc sebelumnya didefinisikan sebagai variabel yang dapat menyimpan data bilangan bulat (integer) maka, variabel abc hanya dapat diisi oleh bilangan 1,2,3, dan lain sebagainya. Variabel abc tidak akan bisa diisi dengan bilangan desimal seperti 2.5. dengan kata lain, tipe data adalah sebuah pengklasifikasian sebuah data yang dapat disimpan dalam sebuah variable

2.3.1 Tipe Data Dasar

Tipe dasar adalah tipe yang sudah didefenisikan pada suatu bahasa pemrograman (dalam hal ini Pascal) dan siap untuk digunakan. Dalam algoritma, kita mengenal 5 tipe dasar. Berikut ini adalah tabel yang berisikan macam-macam tipe yang ada pada algoritma dan transisinya pada bahasa Pascal.

Tabel 2.1: tipe dasar pada algoritma dan bahasa Pascal

Dalam Algoritma	Dalam Pascal	Keterangan
Integer	**Integer** **Byte** **Longint** **Shortin**	Digunakan untuk variabel yang menyimpan bilangan bulat
String	**String**	Digunakan untuk variabel yang menyimpan
Real	**Real** **Single** **Double** **Extended** **Comp**	Digunakan untuk variabel yang menyimpan bilangan riil
Karakter	**Char**	Digunakan untuk variabel yang menyimpan
Booelan	**Boolean**	Digunakan untuk variabel yang menyimpan nilai TRUE (1) atau FALSE

Contoh penggunaan tipe data dasar :

Pseudo Code	Kode Bahasa Pascal	Output
Kamus a : word i: integer z: longint x,y: real found: boolean Algoritma a ← 15 output(a) i ← 15000 output(i) x← 0.5 output(x) y←1.56789 output(y) z←1000000 output(z) found ← true output(found)	Program TDasar; Var a : word; i: integer; z: longint; x,y: real; found: boolean; Begin a:=15; writeln('a= ',a); i:=15000; writeln('i= ',i); x:=0.5; writeln('x= ',x); y:=1.56789; writeln('y= ',y:0:2); z:=1000000; writeln('z= ',z); found := true ; writeln('found = ',found) ; readln; End.	a= 15 i=15000 x=5.0000000 E-001 y=1.57 z=1000000 found=TRUE

Contoh Tipe Data Dasar String :

Pseudo Code	Kode Bahasa Pascal	Output
Program bacaSTR1 kamus str1, str2: string Algoritma input(str1) output(str1) str2←str1 output(str2)	(*deskripsi : alokasi string, kemudian mengisi dengan membaca*) Program bacaSTR1; Var str1, str2: string; Begin writeln('Baca string, maks 256 karakter'); readln(str1); writeln('String yang dibaca : ',str1); str2:=str1; writeln('String yang disalin : ',str2); readln; End.	Baca string, maks 256 karakter Halo String yang dibaca : Halo String yang disalin : Halo
Program bacastr2 Kamus nim,nama: string Begin input(nim) input(nama) output(nim,nama)	(*deskripsi : contoh membaca string kemudian menuliskan yang dibaca*) Program bacastr2; Var nim,nama: string; Begin write('masukkan nim kamu : '); readln(nim); write('masukkan nama kamu : '); readln(nama); writeln('selamat datang ',nama, ' (',nim, ') '); readln; End.	masukkan nim kamu : 11314 masukkan nama kamu : beni selamat dating beni (11314)

2.3.2 Konstanta

Konstanta adalah sebuah tipe data yang bernilai tetap dan tidak akan pernah berubah selama program dijalankan. Konstanta dapat bernilai numerik (angka), karakter, ataupun sebuah String. Dalam penulisan program, konstanta didefinisikan didalam blok CONST yang ada di atas blok VAR.

Contoh Konstanta :

Pseudo Code	Kode Bahasa Pascal	Output
Program KONSTANTA Kamus r,luas: real Constant pi=3.1415 Algoritma input(r) luas←phi*r*r output(luas)	Program KONSTANTA; CONST pi = 3.1415; VAR r,luas: real; begin write('Jari-jari lingkaran = '); readln(r); luas:=pi*r*r; writeln('Luas lingkaran = ',luas:0:2); writeln('Akhir program'); readln; end.	Jari-jari lingkaran = 6 Luas lingkaran = 113.09 Akhir program

2.3.3 Record / Tipe Bentukan

Record / tipe bentukan adalah tipe yang didefinisikan sendiri oleh programmer untuk menampung elemen data yang tipenya tidak perlu sama dengan tujuan mewakili satu jenis objek.

Contoh tipe bentukan:

Pseudo Code	Kode Bahasa Pascal	Output
Program tipe1 TYPE Point = < X : integer Y : integer > Kamus P1,P2 : point Algoritma P1.x ←1 P1.y ←2 Output(P1.x,P1.y) input(P2.x) input(P2.y) output(P2.x,P2.y)	**{contoh pendefinisian dan pengisian struktur record point}** Program tipe1; TYPE Point = record X : integer; (*absis*) Y : integer; (*ordinat*) end; VAR P1,P2 : point; Begin writeln('contoh mengisi struktur dengan assignment'); 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 : '); readln(P2.x); write('ordinat : '); readln(P2.y); writeln('koordinat : ',P2.x, ', ',P2.y); readln; End.	contoh mengisi struktur dengan assignment titik P1, dengan P1.x dan P1.y : P1.x = 1 P1.y = 2 Baca titik P2 Absis : 3 Ordinat : 4 Koordinat : 3,4

Contoh tipe bentukan :

Pseudo Code	Kode Bahasa Pascal	Output
Program tipe2 TYPE MAHASISWA = < Nama : string NIM : string Nilai: real > Kamus Mhs : MAHASISWA Algoritma Mhs.nama ← 'Juliete' Mhs.nim ← '7473' Mhs.nilai ← 80 Output(mhs.nama) Output(mhs.nim) Output(mhs.nilai) {pemakaian WITH untuk RECORD} With Mhs do Output(nama) Output(nim) Output(nilai)	Program tipe2; TYPE MAHASISWA = record Nama : string; NIM : string; Nilai: real; end; VAR Mhs : MAHASISWA; {Harap diperhatikan bahwa "Mhs" adalah nama variabel dan "MAHASISWA" adalah nama tipe bentukan} Begin writeln('contoh mengisi struktur dengan assignment'); Mhs.nama := 'Juliete'; 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; readln; End.	contoh mengisi struktur dengan assignment Hasil assignment thd mhs nama = Juliete NIM = 7473 nilai = 80.00 Hasil assignment thd mhs nama = Juliete NIM = 7473 nilai = 80.00

2.4 Assignment

Assignment adalah proses pemberian nilai terhadap suatu variabel. dan tentunya setiap nilai yang dimasukkan harus sesuai dengan tipe data variabel.

Contoh Assignment:

Pseudo Code	Kode Bahasa Pascal	Output
Program asign kamus x: integer y: integer z: string Algoritma x←100 y←500 z← 'prolab' output(x)	(*deskripsi : assignment suatu variabel dan menampilkan nilai variabel yang diassign*) Program asign; Var x: integer; y: integer; z: string; Begin writeln('contoh program assignment! '); x:=100;	Nilai x : 100 Nilai y : 500 Nilai z : prolab Nilai max integer : 32767

```
    output(y)
    output(z)
    output(maxint)
    output(maxlongint)
```

```
    y:=500;
    z:= 'prolab';
    writeln('nilai x          :
',x);
    writeln('nilai y          :
',y);
    writeln('nilai z          :
',z);
    writeln('nilai max integer :
',maxint);
    writeln('nilai max longint :
',maxlongint);
    readln;
End.
```

```
Nilai max longint :
2147483847
```

2.5 Ekspresi dan Operator

Ekspresi merupakan suatu "rumus perhitungan", yang terdiri dari operand dan operator. Operand harus memiliki nilai, oleh karenanya operand dapat berupa variabel, konstanta atau nilai itu sendiri. Berdasarkan hasil dari ekspresi, ekspresi dibagi menjadi dua jenis, yaitu:

1. Ekspresi Boolean

Ekspresi	Hasil
True and False	False
True or False	True

2. Ekspresi Numerik

Ekspresi	Hasil
1 + 5	6
1 + 3 * 5	16
1/2	0.5
1.9/0.2	9.5
10 mod 3	1
10 div 3	3

Sama dengan ekspresi, operator juga dibagi menjadi 2 jenis, yaitu operator Boolean (operator logika) dan operator numerik. Berikut ini daftar operator boolean pada algoritma dan pada bahasa Pascal.

Tabel 2.2. notasi operator logika pada algoritma dan Bahasa Pascal

Dalam Algoritma	Dalam Pascal
<	<
>	<
≥	>=
≤	<=
=	=
≠	<>

Tabel 2.3. notasi operator numerik pada algoritma dan Bahasa Pascal

Dalam Algoritma	Dalam Pascal
+	+
-	-
*	*
/	/
mod	mod
div	div

Apa itu div dan mod?

Lihat contoh berikut :
5 / 2= 2.5 sedangkan 5 div 2=2, berarti operator "div" sama saja dengan operator "bagi", tetapi pada "div", angka di belakang koma dihilangkan. Mod adalah sisa hasil div. Sebagai contoh "11 mod 3=2" karena 11 div 3 = 3 sisa 2.

Pseudo Code	Kode Bahasa Pascal	Output
Program oparit Kamus i,j: integer x,y,hsl: real Algoritma i←100 j←200 Output(i) Output(j) Output(i+j) Output(i-j) Output(i*j) Output(i div j) Output(i mod j) x←100 y←3 Output(x) Output(y) Hsl ← x/y Output(Hsl)	(*deskripsi : contoh pengoperasian aritmatika*) Program oparit; Var i,j: integer; x,y,hsl: real; Begin writeln('Contoh operator Aritmatik'); i:=100; j:=200; writeln('i = ',i); writeln('j = ',j); writeln('hasil penjumlahan : ',i+j); writeln('hasil pengurangan : ',i-j); writeln('hasil perkalian : ',i*j); writeln('hasil div : ',i div j); writeln('hasil mod : ',i mod j); x:=100; y:=3; writeln('x = ',x); writeln('y = ',y); Hsl:x/y; writeln('hasil pembagian : ',Hsl:0:2); readln; End.	Contoh operator Aritmatik I = 100 J = 200 Hasil penjumlahan : 300 Hasil pengurangan : -100 Hasil perkalian : 2000 Hasil div : 0 Hasil mod : 100 X = 1.0000000000E+002 Y = 3.0000000000E+000 Hasil pembagian : 33.33

Pseudo Code	Kode Bahasa Pascal	Output
Program oplog Kamus bool1,bool2:Boolean Algoritma bool1←true bool2←false Output(bool1) Output(bool2) Output(bool1 AND bool2) Output(bool1 OR bool2)	(*deskripsi : contoh pengoperasian logika*) Program oplog; Var bool1,bool2:Boolean; Begin writeln('Contoh operator logika'); bool1:=true; bool2:=false; writeln('bool1 = ',bool1); writeln('bool2 = ',bool2); writeln(bool1, ' AND ',bool2, ' = ', bool1 AND bool2); writeln(bool1, ' OR ',bool2, ' = ',	Contoh operator logika Bool1 = TRUE Bool2 = FALSE TRUE AND FALSE = FALSE TRUE OR FALSE = TRUE NOT TRUE = FALSE

Output(NOT bool1) Output(bool1 XOR bool2)	bool1 OR bool2); writeln('NOT ',bool1, ' = ', NOT bool1); writeln(bool1, ' XOR ',bool2, ' = ',bool1 XOR bool2); readln; End.	TRUE XOR FALSE = TRUE

Apa itu Array?

Contoh program yang di bawah ini merupakan penggunaan Array atau Larik pada pemrograman Pascal. Apa itu Array/Larik? Pada dasarnya array sama seperti variabel yang berfungsi untuk menampung nilai, hanya saja pada array kita dapat menampung lebih dari suatu nilai. Hal ini berbeda dengan variabel yang hanya dapat menampung satu nilai saja. Layaknya variabel, array juga memiliki tipe data yang harus dideklarasikan terlebih dahulu, seperti integer, string, boolean, dan lain-lain. Dikarenakan array dapat menampung lebih dari suatu nilai maka untuk pengaksesan tiap-tiap elemennya dibutuhkan indeks. Untuk penjelasan lebih lanjutnya akan dibahas pada modul 7.

Pseudo Code	Kode Bahasa Pascal	Output
Program ApaItuArray Kamus larik : array [1..3] of integer Algoritma {Proses pengisian elemen array} input (larik[1]) input (larik[2]) input (larik[3]) Output (larik[1]) Output (larik[2]) Output (larik[3])	Program ApaItuArray; uses crt; Var larik : array [1..3] of integer; begin Clrscr; {Proses pengisian elemen array} write ('Masukkan bilangan ke-1 : '); readln (larik[1]); write ('Masukkan bilangan ke-2 : '); readln (larik[2]); write ('Masukkan bilangan ke-3 : '); readln (larik[3]); writeln ('Bilangan ke-1 : ',larik[1]); writeln ('Bilangan ke-2 : ',larik[2]); writeln ('Bilangan ke-3 : ',larik[3]); readln; end.	Masukkan bilangan ke-1 : 3 Masukkan bilangan ke-2 : 4 Masukkan bilangan ke-3 : 2 Bilangan ke-1 : 3 Bilangan ke-2 : 4 Bilangan ke-3 : 2

Soal

1. Tuliskan program yang dapat menerima input/masukan biodata diri mahasiswa meliputi: nim, nama, kelas;

   format tampilan :

   NIM : <mengisiNIM>
   Nama:<mengisiNama>
   Kelas :<mengisiKelas>
   <selang 2 baris>
   Selamat, <nama> anda memiliki NIM = <NIM> dengan kelas <Kelas>

   Program anda jalan.

2. Apa perbedaan antara read dan readln?
3. Jelaskan perbedaan antara kedua pernyataan berikut:
   `WriteLn(279);` dan `WriteLn(279 : 8);`
4. Buatlah sebuah program untuk menghitung selisih waktu awal dan waktu akhir suatu percakapan dan menuliskan durasi waktu dalam detik. Gunakan tipe bentukan!

   Contoh :

   ---input---

   Jam Awal : 08.45.00

   Jam Akhir : 09.00.30

   ---output---

   Lamanya waktu pembicaraan : 930 detik

5. Buatlah sebuah program untuk menginputkan 3 mahasiswa (nama, nim, nilai) lalu menampilkan nilai rata-ratanya! (gunakan record)!
6. Buat program yang dapat menukar isi dua buah variabel yang bertipe integer.

   format tampilan :

   Masukan nilai A= <mengisi nilaiA> Masukan nilai B= <mengisi nilaiB> Setelah ditukar menjadi:

   Nilai A = <nilaiA> Nilai B = <nilaiB>

   Conto h :

   Masukan nilai A= 5

   Masukan nilai B= 3

   Setelah ditukar menjadi: Nilai A = 3

   Nilai B = 5

Modul 3 Percabangan

Tujuan
Praktikan diharapkan dapat : o Mengerti esensi penggunaan Percabangan dalam Algoritma dan Pascal. o Memahami bentuk umum Percabangan (pemilihan) o Mampu memecahkan masalah sederhana dengan menggunakan Percabangan serta mengimplementasikan ke dalam Pascal.

Analisis kasus adalah sebuah penentuan kasus-kasus yang mungkin terdapat didalamnya. Untuk setiap kasus ada aksi tertentu yang dilakukan. Adanya analisis kasus menyebabkan terjadi pemilihan instruksi (atau percabangan) dalam algoritma, tergantung dari kasus mana yang terpenuhi.

3.1 If – Then

3.1.1 If – Then 1 kondisi

Adalah sebuah kasus dimana aksi akan dilaksanakan bila kondisi bernilai benar (true). Bila kondisi salah (false), tidak ada aksi apapun yang dikerjakan.

Pseudo Code	Kode Bahasa Pascal
Program CekNilai Kamus Angka : integer Algoritma Input(angka) If(angka > 0) then Output(“lebih besar dari 0”)	(* Deskripsi : program untuk menentukan apakah angka yang diinputkan lebih besar dari 0 atau tidak *) Program CekNilai; Uses crt; Var Angka:integer; Begin Clrscr; write(masukkan suatu angka: „); readln(Angka); If (Angka>0) then write(“ Angka yang anda masukkan lebih besar dari 0“); readln; End.

3.1.2 If – Then 2 kondisi

Adalah sebuah kasus dimana $aksi_1$ akan dilaksanakan bila kondisi bernilai benar (true). Bila kondisi salah (false), maka $aksi_2$ yang akan dilaksanakan.

Pseudo Code	Kode Bahasa Pascal
Program Ganjil Uses crt Kamus Angka:integer Algoritma readln(Angka) If (Angka mod 2 = 1) then	(* Deskripsi : program untuk menentukan apakah angka yang diinputkan termasuk bilangan ganjil atau genap*) Program Ganjil; Uses crt; Var Angka:integer; Begin Clrscr; write('masukkan suatu angka: ');

Output(“ganjil”) Else Output(“genap”)	readln(Angka); If (Angka mod 2 = 1) then write('angka yang anda masukkan bilangan ganjil') Else write('angka yang anda masukkan bilangan genap'); readln; End.

3.1.3 If – Then lebih dari 2 kondisi

Adalah sebuah kasus dimana aksi tertentu akan dilaksanakan apabila memenuhi kondisi tertentu yang telah ditetapkan.

Pseudo Code	Kode Bahasa Pascal
Program Cek Kamus angka:integer Algoritma input(angka) If (angka>0) then Output(‘positif') Else if (angka<0) then Output (‘negatif') Else Output (‘nol')	(* Deskripsi : program untuk mengecek apakah angka yang diinputkan positif, negatif, atau nol *) Program Cek; Uses crt; Var angka:integer; Begin Clrscr; write('masukkan suatu angka: '); readln(angka); If (angka>0) then writeln('angka yang anda masukkan bernilai positif') Else if (angka<0) then writeln('angka yang anda masukkan bernilai negatif') Else writeln('angka yang anda masukkan adalah nol'); readln; End.

3.1.4 If – Then Bertingkat

Adalah sebuah kasus dimana didalam sebuah kondisi masih terdapat kondisi lainnya untuk menjalankan aksi tertentu.

Pseudo Code	Kode Bahasa Pascal
Program Kuadran Type Ttitik=< X:real Y:real > Kamus titik:Ttitik Algoritma input(titik.X) input (titik.Y) if (titik.X>0) and (titik.Y>0) then output('Titik terletak di Kuadran I') else if (titik.X<0) and	(* Deskripsi : program untuk mencari posisi sebuah titik pada kuadran dengan penanganan kasus titik berada pusat koordinat, sumbu x, atau sumbu y *) Program Kuadran; Uses crt; Type Ttitik=record X:real; Y:real; end; Var titik:Ttitik; Begin Clrscr; write('Masukkan Koordinat X : '); readln(titik.X); write('Masukkan Koordinat Y : '); readln(titik.Y); if (titik.X>0) and (titik.Y>0) then

```
(titik.Y>0) then
       output ('Titik terletak di
Kuadran II')
    else   if   (titik.X<0)   and
(titik.Y<0) then
       output ('Titik terletak di
Kuadran III')
    else   if   (titik.X>0)   and
(titik.Y<0) then
       output ('Titik terletak di
Kuadran IV')
    else
       if      (titik.X=0)      and
(titik.Y=0) then
         output  ('Titik  di  pusat
koordinat 0,0')
       else
         if (titik.X=0) then
             output   ('Titik   di
sumbu Y')
         else
             output   ('Titik   di
sumbu X')
```

```
       writeln('Titik terletak di Kuadran I')
    else if (titik.X<0) and (titik.Y>0) then
       writeln('Titik terletak di Kuadran II')
    else if (titik.X<0) and (titik.Y<0) then
       writeln('Titik terletak di Kuadran III')
    else if (titik.X>0) and (titik.Y<0) then
       writeln('Titik terletak di Kuadran IV')
    else
       begin
       if (titik.X=0) and (titik.Y=0) then
         writeln('Titik di pusat koordinat 0,0')
       else
         begin
         if (titik.X=0) then
             writeln('Titik di sumbu Y')
         else
             writeln('Titik di sumbu X');
         end;
       end;
    readln;
End.
```

Pseudo Code	Kode Bahasa Pascal
Program Kuadran Kamus kondisi1 : boolean kondisi2 : Boolean Algoritma kondisi1←true kondisi2←false If (kondisi1 AND kondisi2)then Output(kondisi1, ' AND ',kondisi2, ' TRUE') Else Output(kondisi1, ' AND ',kondisi2, ' FALSE') If (kondisi1 OR kondisi2) then Output(kondisi1, ' OR ',kondisi2, ' TRUE') Else Output(kondisi1, ' OR ',kondisi2, ' FALSE')	(* Deskripsi : program untuk mengecek status boolean *) Program Kuadran; Uses crt; Var kondisi1 : boolean; kondisi2 : Boolean; Begin Clrscr; kondisi1:=true; kondisi2:=false; If (kondisi1 AND kondisi2)then writeln(kondisi1,' AND ',kondisi2, ' bernilai TRUE') Else writeln(kondisi1,' AND ',kondisi2,' bernilai FALSE'); If (kondisi1 OR kondisi2) then writeln(kondisi1, ' OR ',kondisi2, ' bernilai TRUE') Else writeln(kondisi1, ' OR ',kondisi2, ' bernilai FALSE'); readln; End.

Pseudo Code	Kode Bahasa Pascal
Program Kuadran Uses crt Kamus nilai1 : integer nilai2 : integer Algoritma input(nilai1) input(nilai2) If (nilai1 > nilai2)then Output(nilai1, ' > ',nilai2)	(* Deskripsi : program untuk mengecek status boolean *) Program Kuadran; Uses crt; Var nilai1 : integer; nilai2 : integer; Begin Clrscr; writeln(„masukkan suatu nilai integer“); readln(nilai1);

```
    If (nilai1 < nilai2)then
       Output(nilai1, ' < ',nilai2)

    If (nilai1 >= nilai2)then
       Output(nilai1, ' >= ',nilai2)

    If (nilai1 <= nilai2)then
       Output(nilai1, ' <= ',nilai2)

    If (nilai1 <> nilai2)then
       Output(nilai1, ' <> ',nilai2)

    If (nilai1 = nilai2)then
       Output(nilai1, ' = ',nilai2)
```

```
    writeln(„masukkan nilai pembatas");
readln(nilai2);
    If (nilai1 > nilai2)then
       writeln(nilai1, ' > ',nilai2);

    If (nilai1 < nilai2)then
       writeln(nilai1, ' < ',nilai2);

    If (nilai1 >= nilai2)then
       writeln(nilai1, ' >= ',nilai2);

    If (nilai1 <= nilai2)then
       writeln(nilai1, ' <= ',nilai2);

    If (nilai1 <> nilai2)then
       writeln(nilai1, ' <> ',nilai2);

    If (nilai1 = nilai2)then
       writeln(nilai1, ' = ',nilai2);
    readln;
End.
```

3.1.5 Case of

Untuk masalah dengan dua kasus atau lebih, konstruksi CASE dapat menyederhanakan penulisan IF-THEN-ELSE yang bertingkat – tingkat. Konstruksi CASE memeriksa apakah nilai dari ekspresi tersebut sama dengan salah satu dari $nilai_1$, $nilai_2$, $nilai_3$, ..., $nilai_n$ (semua nilai berbeda). Jika nilai ekspresi sama dengan salah satu dari nilai –nilai tersebut, maka aksi yang bersesuaian dilaksanakan. Apabila tidak terdapat suatu kecocokan maka akan melakukan aksi yang terdapat dalam nilai otherwise/else. Otherwise/else bersifat optional.

Pseudo Code

```
Program menu

kamus
    pilihan:char

Algoritma

    Output('menu')
    input(pilihan)

    depend on pilihan
       case '1':
         Output('Selamat datang di menu
input
              anggota')
       case '2':
         Output('Selamat datang di menu
update
              anggota')
       case '3':
         Output('Selamat datang di menu
delete
              anggota')
       case '4':begin
         Output('Keluar')
```

Kode Bahasa Pascal

```
(* Deskripsi     : program memilih menu*)
Program menu;
uses     crt;

var
    pilihan:char;
 begin
    clrscr;
    writeln('Menu Utama');
    writeln('1. Input Data');
    writeln('2. Update Data');
    writeln('3. Delete Data');
    writeln('4. Keluar');
    write('Pilih : ');
    readln(pilihan);

    case pilihan of
       '1':begin
         writeln('Selamat   datang   di   menu
input
              anggota');
         readln;
         end;
       '2':begin
         writeln('Selamat   datang   di   menu
update
              anggota');
         readln;
         end;
       '3':begin
         writeln('Selamat   datang   di   menu
delete
```

	anggota'); readln; end; '4':begin writeln('Keluar'); readln; End END.

Contoh Indeks nilai

Pseudo Code	Kode Bahasa Pascal
Program nilai kamus nilai:integer nama, kelas, indeks:string Algoritma clrscr input(nama) input(kelas) input(nilai) depend on nilai 0..54 : indeks ← 'E' 55..64 : indeks ← 'D' 65..74 : indeks ← 'C' 75..84 : indeks ← 'B' 85..100 : indeks ← 'A' output(nama,kelas,indeks)	(* Deskripsi : program melakukan penentuan indeks nilai*) Program nilai; uses crt; var nilai:integer; nama, kelas, indeks:string; begin clrscr; write('nama mahasiswa : '); readln(nama); write('kelas : '); readln(kelas); write('nilai : '); readln(nilai); case nilai of 0..54 : indeks := 'E'; 55..64 : indeks := 'D'; 65..74 : indeks := 'C'; 75..84 : indeks := 'B'; 85..100 : indeks := 'A'; end; write(nama,' mahasiswa ',kelas,' mendapatkan indeks',indeks); readln; end.

Soal

1. Cari kesalahan dalam kode program berikut, kemudian tuliskan kode program yang benar (*correct*).

```
If Bil < 0 then
        writeln("negatif");
Else
        writeln("Positif");
writeln("selesai");
```

2. Tuliskan program Pascal untuk masing – masing persoalan berikut:
   a. Jika total pembelian lebih dari 100.000, tampilkan tulisan 'anda mendapatkan diskon 10%'.
   b. Tampilkan tulisan 'Digit" jika variabel Kar(Bertipe Char) berisi sebuah digit atau menuliskan 'Bukan Digit' kalau Kartidak berisi digit.

3. Suatu hari, Jono sedang belajar angka-angka. Tapi dia tidak bisa menentukan mana angka yang lebih besar dari angka yang lain. Kakak nya, Joni mencoba membantu jono dengan membuatkan sebuah program untuk mengetahui mana bilangan yang paling besar dan mana bilangan yang paling kecil. Coba bantu joni untuk mengerjakan program tersebut! (inputan 4 bilangan)

4. Tuliskan program untuk menghitung total harga barang yang mengikuti aturan berikut:
   a. Jika jumlah barang yang dibeli < 100 buah, maka harga per barang adalah Rp 10.000,00
   b. Jika jumlah barang yang dibeli ≥ 100 buah dan ≤ 150, maka harga per barang adalah Rp 9.500,00
   c. Jika jumlah barang yang dibeli ≥ 150, maka hargar per barang adalah Rp 9.000,00

5. Tuliskan program yang menghasilkan keluaran nama bulan tertentu berdasarkan angka yang dimasukkan oleh user.
   format tampilan :

   Masukkan angka: <mengisi angka>
   < selang 2 baris >

   Bulan ke <angka> adalah bulan <nama bulan ke-angka>

   Contoh :

   Masukkan angka: 4

   < selang 2 baris >

   Bulan ke 4 adalah bulan April

6. Dahulu, kalkulator hanya dapat melakukan 4 operasi sederhana, yaitu pertambahan, pengurangan, pembagian, dan perkalian terhadap HANYA 2 buah bilangan. tolong buatkan program yang bisa melakukan seperti apa yang bisa dilakukan oleh kalkulator dahulu kala dengan tampilan seperti dibawah ini.

   tampilan program:

Operasi yang dapat dijalanakan :
1. pertambahan
2. pengurangan
3. perkalian
4. pembagian
pilihan anda (1-4) : <user input>

==clrscr==
masukkan bilangan pertama :
masukan bilangan kedua :
hasil dari operasi : <keluar hasil operasi yang sudah dipilih dari 2 bilangan yang sudah dimasukan)

Modul 4 Fungsi dan Prosedur

Tujuan
Praktikan diharapkan dapat : o Mengerti esensi penggunaan Fungsi dan Prosedur dalam Algoritma dan Pascal. o Memahami bentuk umum fungsi dan prosedur o Memahami arti dari variabel global, lokal, parameter variabel, parameter konstanta, parameter formal, dan parameter aktual o Memecahkan masalah sederhana dengan menggunakan fungsi dan prosedur dan mengimplementasikan ke dalam Pascal.

4.1 Fungsi

Suatu teknik yang biasa diterapkan dalam pemrograman terstruktur adalah teknik rancang atas-bawah (top-down design). Berdasarkan falsafah rancang atas-bawah, maka suatu program dipecah menjadi beberapa sub bagin lebih kecil dengan tugas tertentu. Dalam Pascal proses pemecahan program menjadi sub program ada 2 macam, yaitu: fungsi dan procedure.

Dalam pembuatan sebuah program, ada kalanya 2 atau lebih *linecode* dibutuhkan untuk digunakan berkali-kali dalam alur jalannya program di tempat yang berbeda (tidak sekuensial), misal untuk menampilkan hasil sebuah operasi, maupun untuk menampilkan sebuah menu dalam sebuah program. Untuk melakukan hal tersebut, penggunaan *linecode* tersebut dapat direduksi atau dapat digantikan dengan pemanggilan sebuah subprogram. Subprogram terdapat 2 jenis, fungsi dan prosedur. Fungsi digunakan ketika bagian program yang ingin diulang menghasilkan satu nilai hasil atau keluaran. Prosedur digunakan ketika bagian program yang ingin diulang tidak menghasilkan apapun (seperti ketika menampilkan menu) atau menghasilkan keluaran lebih dari 1 buah.

Fungsi digunakan apabila modul program mengembalikan sebuah nilai, sementara **prosedur** digunakan bila modul menghasilkan efek netto dari (satu atau) sekumpulan aksi. Namun dalam praktek, sering perbedaan antara keduanya tidak jelas.

Bentuk umum dari suatu fungsi adalah sebagai berikut:

FUNCTION identifier(daftar parameter) : type;

Parameter merupakan variabel yang dituliskan di dalam kurung setelah nama prosedur dan fungsi. Dalam pembuatannya program biasanya diperlukan pertukaran informasi antara prosedur dan fungsi. Berdasarkan penulisannya terdapat 2 jenis parameter, yaitu: **parameter formal** dan **parameter aktual**. **Parameter formal** adalah parameter yang dituliskan pada saat pendefinisian prosedur dan fungsi itu sendiri.

Sedangkan **parameter aktual** adalah parameter yang disertakan pada saat pemanggilan prosedur dan fungsi tersebut di blok program utama.

Seperti yang sudah dijelaskan sebelumnya, sebuah fungsi digunakan ketika program yang dibuat akan menghasilkan satu nilai keluaran. Nilai keluaran ini disebut dengan *return value* dalam fungsi, wajib adanya pendefinisian ini nilai dari *return value* tersebut.

Agar lebih memahami tentang fungsi, perhatikan contoh berikut:

Perkalian dan penjumlahan:

Pseudo Code	Kode Bahasa Pascal
{sebaiknya ada deskripsi dari algoritma yang dibuat} Program xtambah kamus a,b,hasilkali,hasiljumlah:integer function kali (a,b:integer): integer function jumlah (a,b:integer): integer (*program utama*) Algoritma input(a) input(b) hasilkali ← kali(a,b) hasiljumlah ← jumlah(a,b) Output(hasilkali) Output(hasiljumlah) function kali (a,b:integer): integer Algoritma kali ← a * b function jumlah (a,b:integer): integer Algoritma jumlah ← a + b	(* Deskripsi : program melakukan perkalian dan penjumlahan fungsi kali dan jumlah*) Program xtambah; Uses crt; var a,b,hasilkali,hasiljumlah:integer; function kali (a,b:integer): integer; begin kali := a * b; end; function jumlah (a,b:integer): integer; begin jumlah := a + b; end; (*program utama*) Begin writeln('Masukkan angka pertama : '); readln(a); writeln('Masukkan angka kedua : '); readln(b); hasilkali := kali(a,b); hasiljumlah := jumlah(a,b); writeln('Hasil kali kedua angka adalah : ',hasilkali); writeln('Hasil bagi kedua angka adalah : ',hasiljumlah); readln; End.

Indeks Nilai

Pseudo Code	Kode Bahasa Pascal
{sebaiknya ada deskripsi dari algoritma yang dibuat} Program nilai kamus nilai : integer nama, kelas, index : string function indeks (nilai : integer) :char; Algoritma input(nama) input(kelas) input(nilai) Index ← indeks(nilai) output(nama, kelas, index) function indeks (nilai : integer) :char Algoritma Depend on nilai 0..54 : indeks ← 'E' 55..64 : indeks ← 'D'	(* Deskripsi : program melakukan penentuan indeks nilai melalui fungsi indeks yang menghasilkan keluaran berupa karakter *) Program nilai; Uses crt; var nilai : integer; nama, kelas, index : string; function indeks (nilai : integer) :char; BEGIN Case nilai of 0..54 : indeks := 'E'; 55..64 : indeks := 'D'; 65..74 : indeks := 'C'; 75..84 : indeks := 'B'; 85..100 : indeks : = 'A'; end; end; Begin clrscr; write(' Nama Mahasiswa : '); readln(nama); write(' Kelas : ');readln(kelas);

```
    65..74 : indeks ← 'C'
    75..84 : indeks ← 'B'
    85..100 : indeks ← 'A'
```

```
    write(' Nilai          :  ');readln(nilai);
    Index := indeks(nilai);
    write(nama,  '  mahasiswa  ',  kelas,  '
mendapatkan indeks ',
      index);
End.
```

Soal

1. Tuliskan program untuk memeriksa sebuah bilangan bulat yang diberikan user merupakan bilangan genap atau bukan. Pendefinisian bilangan tersebut merupakan bilangan genap atau bukan dituliskan dalam sebuah **function IsGenap().**
2. Carilah letak kesalahan pada pendeklarasian subprogram berikut:

```
FUNCTION Kali (a, b: Integer);
Begin
    Kali := a * b;

End.
```

4.2 Prosedur

Seperti yang sudah dijelaskan sebelumnya prosedur adalah sebuah subprogram yang digunakan untuk kasus-kasus yang tidak menghasilkan keluaran apapun atau kasus-kasus yang membutuhkan nilai keluaran lebih dari sama dengan 2 (walaupun dapat deprogram untuk yang memiliki 1 nilai keluaran). Dalam pendefinisian sebuah prosedur, dalam algoritma ada yang disebut dengan parameter input, parameter output, dan parameter input/output. Namun, dalam Pascal, hanya disediakan dua jenis parameter yaitu parameter input, dan parameter input/output.

Parameter input, adalah parameter yang digunakan sebagai sebuah masukkan dari program utama ke prosedur yang dipanggil. Karena sifatnya hanya sebagai masukkan, maka nilai dari parameter tersebut akan tetap sama sebelum dan setelah eksekusi prosedur tersebut (walaupun dalam prosedur terdapat *code* yang mengubah nilai parameter input tersebut). Sedangkan parameter input/output adalah parameter yang digunakan sebagai masukkan sekaligus sebagai keluaran dari prosedur tersebut. Maka, parameter input/output akan memiliki nilai yang berbeda saat sebelum dan sesudah eksekusi prosedur (jika dalam prosedur terdapat *code* yang mengubah nilai parameter tersebut).

Selain parameter input, dan parameter input/output, dalam subprogram (fungsi dan prosedur) terdapat dua jenis variabel yang dapat digunakan, yaitu variabel lokal dan variabel global. Variabel global adalah variabel yang didefinisikan diawal program dan dapat digunakan oleh seluruh blok program (program utama dan subprogram) sementara variabel lokal adalah variabel yang didefinisikan dalam sebuah subprogram, dan hanya dapat digunakan oleh subprogram yang mendefinisikannya saja. Jika sebuah subprogram mendefinisikan sebuah variabel (variabel lokal) yang dengan nama yang sama seperti pada yang sudah didefinisikan di kamus awal sebelumnya (variabel global), maka dalam subprogram tersebut, jika variabel tersebut dipanggil, maka variabel tersebut akan mengacu ke variabel lokal.

4.2.1 Menukar bilangan (menggunakan parameter by reference)

Maksud dari parameter by reference adalah suatu parameter/variabel yang dimasukkan kedalam prosedur akan berubah nilai setelah keluar dari prosedur tersebut. Perubahan nilai dari suatu variabel tidak hanya berlaku di prosedur tersebut melainkan dikeseluruhan program.

Pseudo Code	Kode Bahasa Pascal
Program subprg kamus a,b : integer procedure tukar(input/output a,b:integer) algoritma input(a) input(b) output(a,b) tukar(a,b) output(a,b) procedure tukar (input/output a,b:integer)	Program subprg; Var a,b : integer; procedure tukar(var a,b:integer); (* menukar nilai dua buah variabel *) var temp : integer; begin temp := a; a := b; b := temp; end; begin writeln('ketikkan bilangan pertama : '); readln(a); writeln('ketikkan bilangan kedua : ');

```
(* menukar nilai dua buah variabel *)
kamus
    temp : integer
algoritma
    temp ← a
    a ←b
    b ← temp
```

```
    readln(b);
    writeln('ke dua bilangan: a = ',a);
    writeln('b   =   ',b);
    tukar(a,b);
    writeln('ke dua bilangan setelah di tukar: a= ',a);
    writeln('b=     ',b);
    readln;
end.
```

4.2.2 Menghitung luas dan keliling (menggunakan parameter by value)

Maksud dari parameter by value adalah suatu parameter/variabel yang dimasukkan kedalam prosedur tidak akan berubah nilainya setelah keluar dari prosedur tersebut. Perubahan nilai dari suatu variabel hanya berlaku di prosedur tersebut.

Pseudo Code	Kode Bahasa Pascal
program luaskel uses crt (* Deskripsi : plih menu, hitung luas dan keliling sesuai pilihan *) kamus bil:integer procedure menu(input pil:integer) algoritma output('Nama bidang :') output('1. Persegi') output('2. Segitiga siku-siku') output('3. Lingkaran') input(bil) menu(bil) procedure menu(pil:integer) kamus s,t,L,K:real algoritma depend on pil case 1: algoritma input(s) L←s*s K←4*s case 2: algoritma input(s) input(t) L←s*t/2 K←s+t+(sqrt(s*s+t*t)) case 3: algoritma input(s) L←s*s*3.14 K←2*s*3.14 else	program luaskel uses crt; (* Deskripsi : plih menu, hitung luas dan keliling sesuai pilihan *) var bil:integer; procedure menu(pil:integer); var s,t,L,K:real; begin case pil of 1: begin write('sisi : '); readln(s); L:=s*s; K:=4*s; end; 2: begin write('alas : '); readln(s); write('tinggi : '); readln(t); L:=s*t/2; K:=s+t+(sqrt(s*s+t*t)); end; 3: begin write('Jari-jari: '); readln(s); L:=s*s*3.14; K:=2*s*3.14; end; else write('invalid input'); end; writeln('Luas : ',L:0:2); writeln('Keliling: ',K:0:2); end; begin writeln('Nama bidang :'); writeln('1. Persegi'); writeln('2. Segitiga siku-siku'); writeln('3. Lingkaran');

```
        output('invalid input')
    end
    output('Luas    : ',L:0:2)
    output('Keliling: ',K:0:2)
```

```
    write('pilihan : '); readln(bil);
    menu(bil);
    readln;
end.
```

Soal

1. Buat procedure untuk menghitung jumlah uang yang harus dikeluarkan oleh mesin ATM dengan spesifikasi : mesin ATM menyimpan 3 pecahan mata uang yaitu pecahan Rp 20.000,- ; Rp10.000,-; dan Rp 5.000. Procedure menggunakan 1 parameter masukan untuk input jumlah uang, dan 3 parameter keluaran untuk jumlah masing – masing pecahan yang harus dikeluarkan. (Misal bila masukan user 35.000 maka akan dikeluarkan jumlah Rp 20.000,- = 1, jumlah Rp 10.000,- = 1, dan Rp5.000,- = 1).

2. Buatlah program untuk konversi suhu dari Celcius ke Reamur/Fahrenheit dengan menggunakan fungsi!

   Reamur=4/5*Celcius
   Fahrenheit=9/5*Celcius+32
   Contoh :

   Suhu Celcius = 33 <input> akan dikonversikan ke
   1. Reamur
   2. Fahrenheit
   Pilih (1/2) = 1 <input>
   Reamur = 26.40 <output>

   Suhu Celcius = 34 <input> akan dikonversikan ke
   1. Reamur
   2. Fahrenheit
   Pilih (1/2) = 2 <input> Fahrenheit = 93.20 <output>

3. Buat sebuah program dengan menggunakan prosedur dan fungsi untuk menghitung volume kubus, silinder, dan kerucut.
4. Andi diminta ibunya untuk membuat sebuah program yang dapat memasukkan tanggal hari ini dan menampilkan tanggal untuk besok hari (dd=mm-yy). Bantulah Andi untuk membuat program tersebut (Gunakan record dan gunakan fungsi untuk perhitungan tanggal besok hari, cek juga apakah tahunnya kabisat atau tidak!)

   Contoh :

   input
   hari : 31
   bulan : 12
   tahun : 92

output
besok adalah tanggal 1-1-93

5. Buatlah program yang mengandung prosedur untuk mentranslasikan suatu titik koordinat dengan menggunakan prosedur translasi (titikawal,translasi). Diwajibkan menggunakan record

   {contoh}

   Input : Koordinat awal

   absis : 3 {input}

   ordinat : 7 {input} Translasi ke

   absis : -7 {input} ordinat : 5 {input}

   Output :

   Koordinat (3,7) setelah ditranslasikan dengan T(-7,5) menjadi (-4,12)

Modul 5 Perulangan

Tujuan
Praktikan diharapkan dapat : o Mengerti esensi penggunaan Perulangan (Looping) dalam Algoritma dan Pascal. o Memilih bentuk pengulangan yang benar dan tepat untuk kelas persoalan tertentu. o Mampu memecahkan masalah sederhana dengan menggunakan Perulangan dan mengimplementasikan ke dalam Pascal.

5.1 Perulangan *For*

Dalam proses pembuatan program ada kalanya kita membutuhkan penggunaan 1 atau lebih *linecode* yang sama untuk digunakan berkali-kali secara sekuensial. Untuk itu terdapat blok kontrol yang memudahkan kita yang disebut dengan perulangan atau *looping*. Sekilas, definisi antara subprogram yang sudah dijelaskan pada modul 4 kurang lebih mirip dengan definisi perulangan di atas. Namun, yang perlu diperhatikan, subprogram digunakan untuk melakukan aksi yang sama namun tidak sekuensial, berbeda dengan perulangan yang harus sekuensial, misal menulis "Hello World" 5 kali secara berurutan. Selain itu, subprogram digunakan biasanya untuk mem-*package* atau membungkus beberapa *linecode* untuk suatu fungsionalitas tertentu. Untuk lebih jelasnya silahkan diperhatikan contoh dibawah ini.

Pseudo Code	Kode Bahasa Pascal
Program forto Kamus i : integer N : integer Algoritma input(N) i traversal [i…N] output(i) output('akhir program')	(* Deskripsi : baca N, print 1 s/d N dengan FOR TO DO *) Program forto; Var i : integer; N : integer; Begin writeln('baca N, print 1 s/d N'); write('N = '); readln(N); for i:= 1 to N do Begin writeln(i); end; writeln('akhir program'); readln; End.

Perlu anda tahu..

Tipe data yang bisa dipakai untuk perulangan for adalah tipe data yang mempunyai successor dan predecessor yaitu :

Integer (termasuk longint,shortint,byte) dan char. Contoh penggunaan char.

```
for c:=”A‟ to “Z‟ do
    writeln(c);
```

Selain itu, dalam perulangan, apabila code yang ingin diulang memiliki jumlah baris lebih dari 1, diperlukan sebuah blok kontrol yang dimulai dengan Begin dan end; sehingga, contoh diatas, apabila ditambahkan perintah writeln(1) misal, maka akan menjadi seperti berikut

```
for c:=”A‟ to “Z‟ do
Begin
    writeln(c);
    writeln(1);
End;
```

Untuk perulangan for sendiri, terdapat 2 jenis perulangan, yaitu perulangan *ascending* (**for – to – do**) dan perulangan *descending* (**for – downto – do**). Pada dasarnya proses perulangan kedua for tersebut adalah sama, yang membedakan adalah perubahan variable setiap iterasi, untuk **for – to – do** setiap iterasi, nilai dari variable bertambah, sedangkan **for – downto – do** nilai dari variable berkurang setiap iterasi. Perulangan for mencapai iterasi terakhir adalah ketika nilai variable sudah sama dengan nilai kedua (setelah kata **to**). Contoh :

Pseudo Code	Kode Bahasa Pascal
Program fordto kamus I, N :integer Algoritma readln(N) i traversal [N..1] Begin Output(i) End Output('akhir program')	(* Deskripsi : baca N, print N s/d 1 dengan FOR DOWNTO DO *) Program fordto; Var I, N :integer; Begin writeln('baca N, print N s/d 1'); write('N = '); readln(N); for i:= N downto 1 do Begin writeln(i); end; writeln('akhir program'); readln End.

Pada program tersebut, nilai variable **i** akan terus berubah. Kondisi awal variable I akan bernilai sama dengan N, karena tipe perulangan downto, maka pada iterasi berikut, variable i akan bernilai N-1, iterasi berikutnya N-2, dan seterusnya hingga i =1**.** Iterasi i = 1 adalah iterasi terakhir pada perulangan tersebut.

Seperti halnya pada bab III, tentang if – then, Penerapan for juga dapat digunakan di dalam for yang lain (perulangan di dalam perulangan) untuk memahaminya, silakan coba tracing program pembentuk pola berikut.

Pseudo Code	Kode Bahasa Pascal
Program fordto kamus i,j,N :integer Algoritma Input(N) i traversal [1..N] J traversal [1..N] output('*') Output(‘/n’) Output('akhir program')	(* Deskripsi : menampilkan bentuk segitiga siku-siku dengan ukuran n kali sesuai dengan inputan user *) Program fordto; Var i,j,N :integer; Begin writeln(„Masukkan nilai n : '); readln(N); for i:= 1 to N do begin for j:=1 to N do begin write('*'); end; writeln; end; writeln('akhir program'); readln; End.

Untuk lebih memahami proses perulangan for, buatlah program untuk membuat sebuah pola yang memberikan output seperti gambar di bawah dengan inputan N=5.

```
*****
****
***
**
*
```

5.2 Perulangan *while – do*

Teknik lain yang dapat digunakan untuk skema perulangan adalah while – do. **While** (kondisi) **do** akan mengulang perintah selama kondisi dipenuhi (bernilai true). Bisa dibilang, perulangan **while – do** ini lebih *powerful* dibandingkan dengan perulangan for, karena setiap permasalahan perulangan yang dapat diselesaikan dengan for, pasti dapat diselesaikan pula dengan while – do, namun tidak untuk sebaliknya, ada kasus dimana permasalahan perulangan hanya dapat diselesaikan dengan skema **while – do.**

Perulangan pada skema While – do akan memeriksa kondisi setiap iterasi. Jika kondisi benar, maka perintah akan dilakukan, oleh karena itu perulangan berhenti ketika kondisi bernilai salah. Hati – hati untuk skema ini, karena jika penentuan kondisi, karena jika kondisi selalu bernilai true, maka perulangan tidak akan pernah berhenti, dan program akan mengalami **error**. Berikut contoh sederhana penggunaan while – do :

Pseudo Code	Kode Bahasa Pascal

```
Program priw1
kamus
    I, N : integer

Algoritma
    input(N)
    i←1
    while (i<=N) do
       output(i)
       i←i+1
```

```
(* Deskripsi     : baca N, print 1 s/d N dengan
while *)
Program priw1;
Var
    I, N : integer;

Begin
    writeln('baca N, print 1 s/d N'); write('N
= ');   readln(N);
    i:=1
    while (i<=N) do
    Begin
       writeln(i);
       i:=i+1;
    end;                {  i>N }
    readln;
End.
```

Pseudo Code	Kode Bahasa Pascal

```
Program priw1
kamus
    I, N : integer

Algoritma
    input(N)

    i←3
    while (i<=N) do
       output(i)
       I←i+3
```

```
(* Deskripsi       : baca N, print bilangan
kelipatan 3 yang lebih kecil dari N dengan
while*)
Program priw2;
Var
    I, N : integer;

Begin
    writeln(baca N, print bilangan kelipatan 3
          yang lebih kecil dari N');
    write('N = ');   readln(N);
    i:=3
    while (i<=N) do
    Begin
       writeln(i);
       i:=i+3;
    end;                {  i>N }
    readln;
End.
```

5.3 Perulangan *repeat - until*

Teknik perulangan **Repeat – until** memiliki dasar yang mirip dengan While – do, yaitu perulangan dilakukan berdasarkan sebuah kondisi. Namun terdapat sedikit perbedaan, yaitu **repeat – until** akan melakukan perulangan selama kondisi bernilai salah, dan proses pemeriksaan kondisi dilakukan di akhir iterasi. Seperti halnya while–do, hati – hati dalam penentuan kondisi berhenti, jika kondisi tidak bisa bernilai true, maka program tidak akan berhenti berulang. Berikut contoh sederhana penggunaan **repeat– until.**

Pseudo Code	Kode Bahasa Pascal

```
Program pirep1
kamus
    I, N : integer

Algoritma
    Input(n)
    I←1
```

```
(* Deskripsi     : baca N, print 1 s/d N dengan
REPEAT *)
Program pirep1;
Var
    i : integer; N : integer;

Begin
    writeln('baca N, print 1 s/d N');
    write('N = '); readln(N);
```

```
    repeat
       output(i);
       i←i+1;
    until (i>N);
```

```
    i:=1
    repeat
       writeln(i);
       i:=i+1;
    until (i>N);
    readln;
End.
```

Penerapan Repeat dalam menu kalkulator

Pseudo Code

```
Program kalkulator
kamus
    a,b : integer
    pil : integer

    function  kali  (a,b:integer):
integer
    function jumlah (a,b:integer):
integer
    function kurang (a,b:integer):
integer
    procedure            input(var
a,b:integer)

Algoritma
    Repeat
       Output('Menu Utama')
       Output('1. Penjumlahan')
       Output('2. Pengurangan')
       Output('3. Perkalian')
       Output('4. Keluar')
       Input(pil)
       Depend on pil
         Case 1:
             Input(a,b)
             Output(jumlah(a,b))
         Case 2:
             Input(a,b)
             Output(kurang(a,b))
         Case 3:
             Input(a,b)
             Output(kali(a,b))
    until pil=4


function    kali    (a,b:integer):
integer
algoritma
    kali ← a * b


function   jumlah   (a,b:integer):
integer
algoritma
    jumlah ← a + b
```

Kode Bahasa Pascal

```
 (*  Deskripsi          :  program  kalkulator
sederhana dengan fungsi dan perulangan *)
Program kalkulator;
Var
    a,b : integer;
    pil : integer;

function kali (a,b:integer): integer;
begin
    kali := a * b;
end;

function jumlah (a,b:integer): integer;
begin
    jumlah := a + b;
end;

function kurang (a,b:integer): integer
begin
    kurang := a - b;
end;

procedure input(var a,b:integer);
begin
    writeln('Masukkan angka pertama : ');
    readln(a);
    writeln('Masukkan angka kedua : ');
    readln(b);
end;

Begin
    Repeat
       Clrscr;
       writeln('Menu Utama');
       writeln('1. Penjumlahan');
       writeln('2. Pengurangan');
       writeln('3. Perkalian');
       writeln('4. Keluar');
       write('Pilih : ');
       readln(pil);
       Case pil of
         1:begin
             Input(a,b);
             writeln('Hasil    penjumlahan    :
',jumlah(a,b));
             readln;
         end;
         2:begin
```

```
function kurang (a,b:integer): integer
algoritma
    kurang ← a - b


procedure input(var a,b:integer)
algoritma
    input(a)
    input(b)
```

```
            Input(a,b);
            writeln('Hasil pengurangan : ',kurang(a,b));
            readln;
        end;
        3:begin
            Input(a,b);
            writeln('Hasil perkalian : ',kali(a,b));
            readln;
        end;
      end;
   until pil=4;
   readln;
End.
```

Soal

1. Tuliskan program untuk membuat sebuah pola angka. Inputan dari program adalah sebuah angka N. Jika N diberi nilai 5, maka output-nya adalah :

   1 2 3 4 5
   2 3 4 5 1
   3 4 5 1 2
   4 5 1 2 3
   5 1 2 3 4

2. Buat program untuk menentukan angka yang diinputkan adalah bilangan prima atau bukan.

   Contoh : Input
   :
   Masukkan angka : 2
   Output : Prima
   Input :
   Masukkan angka : 51
   Output : Bukan Prima

3. Buat program untuk membuat tulisan seperti di bawah ini :
   input :
   masukkan kata :
   abrakadabra output :

   a

   a b

   a b r

   a b r a

   a b r a k

   a b r a k a

   a b r a k a d

   a b r a k a d a

   a b r a k a d a b

   a b r a k a d a b r

   a b r a k a d a b r a

4. Buatlah sebuah program untuk melakukan konversi dari bilangan desimal menjadi bilangan biner.

   Gunakan fungsi : function biner (desimal:integer):string;
   Contoh:

   Bilangan Desimal : 22 {input}

   Bilangan Biner : 10110

   {output}

5. Buatlah program dengan prosedur untuk mencari nilai rata – rata, nilai tertinggi dan nilai terendah dari inputan yang dinputkan oleh user

   contoh :
   banyak data : 5 data ke-1 : 8 data ke-2 :

   10 data ke-3 : 7 data
   ke-4 : 5 data ke-5 : 9
   Nilai rata - rata = 7,8
   Nilai Tertinggi = 10
   Nilai Terendah = 5

Modul 6 Skema Pemrosesan Sekuensial

Tujuan
Praktikan diharapkan dapat : o Memahami pengertian sekuensial. o Memahami item-item sekuensial. o Memahami kasus kosong. o Memahami “mark”. o Memahami penanganan kasus kosong. o Memahami hubungan berulang dan studi kasusnya.

Skema pemrosesan sekuensial adalah cara menyelesaikan permasalahan secara sekuensial atau berurutan. Contoh persoalan yang diselesaikan dengan skema pemrosesan sekuensial adalah menjumlahkan setiap bilangan yang ada dalam jangkauan 1 hingga 100, atau menghitung fungsi faktorial dari 10! dengan mengalikan angka 1 hingga 10 secara satu per satu dan terurut. Modul ini sangat membutuhkan pemahaman yang baik mengenai modul perulangan yang telah dipelajari sebelumnya.

Dalam modul ini, akan dibahas dua jenis skema pemrosesan sekuensial: dengan penanda berakhir (mark) dan tanpa penanda berakhir. Mark adalah penanda yang memberikan sinyal agar perulangan berhenti. Mark biasanya berupa nilai yang tidak termasuk dalam elemen permasalahan.

6.1 Skema Pemrosesan Sekuensial dengan *Mark*

6.1.1 Tanpa Penanganan Kasus Kosong

Pada skema ini, pemrosesan yang dilakukan secara sekuensial menggunakan penanda (mark). Mark disini adalah penanda kapan sebuah pemrosesan sekuensial tersebut akan berhenti. Pada contoh di bawah ini, tidak ada penanganan kasus kosong, dengan kata lain suatu proses ada kemungkinan tidak dilakukan sama sekali pada kondisi tertentu. Pada bagian manakah yang menentukan sebuah kasus kosong atau tidak? Kondisi apa yang dapat membuat sebuah kasus kosong?

Pseudo Code	Kode Bahasa Pascal
Program Seq1 kamus I ,N, Sum : integer algoritma input(N) Sum←0 I←1 {First-Elmt} while I <= N do {EOP} output(i) {Current-Elmt} sum←sum + 1 I←I+1 {Next-Elmt} output(sum)	Program Seq1; {Deskripsi: Menjumlahkan bilangan 1+2+3+4+..N, dengan N > 0 dan diperoleh melalui masukan (input) pengguna} var i : integer; {bilangan yang akan dijumlahkan} n : integer; sum : integer; {jumlah} begin write('Masukkan banyaknya bilangan : '); readln(n); sum := 0; i := 1; {First-Elmt}

	while i <= N do {EOP} begin writeln('Nilai i adalah : ', i); {Current-Elmt} sum := sum + 1; i:=i + 1; {Next-Elmt} end; writeln('Jumlah bilangan adalah ', sum); readln; end.

6.1.2 Dengan Penanganan Kasus Kosong

Berbeda dengan skema sebelumnya, skema kali ini terdapat penanganan kasus kosong. Pada skema ini proses dilakukan minimal satu kali. Hal ini di sebabkan proses pemeriksaan penanda (mark) dilakukan di akhir skema. Salah satu contoh kondisi yang mengakibatkan proses dilakukan hanya satu kali adalah ketika N diinputkan dengan angka 1.

Pseudo Code	Kode Bahasa Pascal
program Seq2 kamus I,N, Sum : integer Algoritma input(N) Sum←0 I←1 repeat sum ← sum + 1 I←I+1 until I > N{Mark} output(sum)	Program Seq2; { Deskripsi : Menjumlahkan bilangan 1+2+3+4+..N, dengan N dibaca dari keyboard} Uses Crt; Var I : integer; {bilangan yang akan dijumlahkan } N : integer; {banyaknya bilangan yang dijumlahkan, N > 0} Sum : integer;{jumlah} Begin {Program Utama} write('Masukan banyaknyabilangan yang akan dijumlahkan: '); readln(N); Sum:=0; I:=1; repeat sum:=sum + 1; I:=I+1; until I > N;{Mark} writeln('Jumlah bilangan adalah ', sum); readln; End.

6.2 Skema Pemrosesan Sekuensial Tanpa Mark

Skema Pemrosesan Sekuensial Tanpa Mark. Skema pemrosesan seperti ini adalah skema pemrosesan yang tidak memakai penanda (mark). Sehingga jumlah iterasi sudah dapat diketahui di awal.

Pseudo Code	Kode Bahasa Pascal
Program Seq3 Kamus I,N,sum : integer Algoritma input(N) Sum←0 i traversal	Program Seq3; { Deskripsi : Menjumlahkan bilangan 1+2+3+4+..N, dengan N dibaca dari keyboard} Uses Crt; Var I : integer; {bilangan yang akan dijumlahkan } N : integer; {banyaknya bilangan yang dijumlahkan,

[1..N] sum←sum + 1 output(sum)	N>0} Sum : integer;{jumlah} Begin {Program Utama} write('Masukan banyaknya bilangan yang akan dijumlahkan : '); readln(N); Sum:=0; for I:=1 to N do sum:=sum + 1; writeln('Jumlah bilangan adalah ', sum); readln; End.

Berikut adalah salah satu contoh penerapan skema pemrosesan sekuensial untuk masalah Menghitung Deret S = 1/2 + 1/ 4 + 1/8 +… (Hubungan Berulang)

Pseudo Code	Kode Bahasa Pascal
Program Seq4 Kamus I, N : integer Temp, S : real Algoritma input(N) I ← 1 temp ← 1 while I <= N do temp ← temp * 0.5 S ← S + temp I ← I + 1 output(S)	{Deskripsi : Menjumlahkan bilangan 1/2+1/4+1/8+..N, dengan N dibaca dari keyboard} Program Seq4; Uses Crt; Var I : integer; {bilangan yang akan dijumlahkan } N : integer; {banyaknya bilangan yang dijumlahkan, N > 0} temp : real; {manampung bilangan pecahan yang akan dijumlahkan} S : real; {jumlah bilangan} Begin {Program Utama} write('Masukan banyak bilangan : '); readln(N); I := 1; temp := 1; while I <= N do begin temp := temp * 0.5; S := S + temp; I := I + 1; end; writeln('Jumlah bilangan adalah ', S); readln; End.

Pemrosesan sekuensial juga dapat digunakan untuk menampilkan faktorial ke n

Pseudo Code	Kode Bahasa Pascal
Program seq1l Kamus I,n,faktorial :integer Begin Faktorial←1 Input(n) i traversal [n..1] Output (faktorial) Faktorial ← faktorial * I	Program seq1l Var I,n,faktorial :integer; Begin Faktorial:=1; write('Faktorial keberapa ?? '); readln(n); for i:= n downto 1 do Begin writeln('Faktorialnya adalah = ', faktorial);

	Faktorial := faktorial * I; end; readln; End.

Contoh lain adalah memangkatkan m^n

Pseudo Code	Kode Bahasa Pascal
Program seq1l Kamus I,n,m,pangkat:integer Algoritma Pangkat←1 Input(m) Input(n) for i:= 1 to n do Pangkat← Pangkat*m Output (pangkat)	Program seq1l Var I,n,m,pangkat:integer; Begin Pangkat:=1; write('masukkan M '); readln(m); write('Masukkan N'); readln(n); for i:= 1 to n do Begin Pangkat:=Pangkat*m; end; writeln ('M pangkat N = ',pangkat); readln; End.

Soal Latihan

1. Buatlah algoritma untuk mencetak bilangan ganjil sebanyak N bilangan.
   Contoh masukan : N = 4
   Contoh keluaran : hasil = 1,3,5,7
   Gunakan model dengan mark dan tanpa mark juga pemrosesan kasus kosong

2. Buatlah algoritma untuk memberikan output dari 2^1 sampai 2^N.
   Contoh input : N = 3
   Contoh output : hasil = 2,4,8
   Catatan : Buatlah asumsi sendiri

3. Buatlah algoritma untuk menghitung deret fibonaci
   Contoh input : N = 7
   Contoh Output : deret = 1 1 2 3 5 8 13 21

4. Buatlah program untuk menampilkan n buah bilangan prima dimana n adalah inputan user.
   Contoh input : N = 5
   Contoh Output : deret = 2, 3, 5, 7, 11

5. Buatlah program untuk menghitung jumlah x bilangan prima pertama.

6. Buat program yang terus meminta inputan user berupa angka. Program akan berhenti ketika angka yang diinputkan user adalah angka prima.
   Contoh Input :
   Masukkan angka : 4
   Masukkan angka : 6
   Masukkan angka : 29
   Contoh Output : 29 adalah prima

Modul 7 Larik (*Array*)

Tujuan
Praktikan diharapkan dapat : o Mengerti definisi, deklarasi tipe tabel/larik dan mengetahui penggunaan tipe larik tersebut. o Dapat memprogram tabel dengan fungsi dan prosedur. o Memahami semua skema searching (*sequential*, *binary*, dan mencari harga ekstrim). o Dapat mengimplementasikan skema searching tersebut terhadap kasus-kasus tertentu. o Dapat menuliskan kode program seluruh skema sorting yang telah diberikan terhadap berbagai jenis data (integer dan string).

Array merupakan sebuah tipe data yang dapat menyimpan banyak data bertipe sama. Array dapat diakses berdasarkan nomor indeksnya. Array dapat dibangun dalam satu, dua atau lebih dimensi. Dimensi array ini disesuaikan dengan kebutuhan.

7.1 Dimensi *Array*

7.1.1 Array Satu Dimensi (1-D)

Contoh kasus yang membutuhkan array satu dimensi misalnya ketika kita ingin menyimpan bilangan-bilangan (bertipe `integer`) atau nama-nama (bertipe `string`) yang dimasukkan pengguna, atau kasus sejenis. Selain `integer` dan `string`, array tentu juga bisa dideklarasikan dengan tipe data yang lain. Tipe data yang digunakan juga dapat berupa tipe data buatan (*record*). Yang perlu diingat adalah bahwa tipe data array haruslah sama. Berikut adalah beberapa contoh pendeklarasian array satu dimensi pada bagian `var`.

```
a : array[1..10] of integer;
b : array[0..9] of integer;
c : array[2..11] of integer;
```

Perhatikan bahwa jangkauan indeks dapat dideklarasikan sesuai keinginan. Dalam berbagai bahasa pemrograman termasuk Pascal, indeks array seringkali dimulai dari nol, bukan satu. Namun banyak orang seringkali memulai indeks dari 1 pada program yang dibangun dengan bahasa Pascal. Ini karena kita lebih terbiasa menghitung sesuatu diawali dari angka 1, bukan 0.

Berikut adalah contoh kode untuk menyimpan bilangan-bilangan yang dimasukkan oleh pengguna.

Kode Bahasa Pascal

```
program TABEL;
{latihan array Untuk memasukkan bilangan sebanyak n dan
Menampilkannya tanpa dengan menggunakan prosedur}
Var
    Bilangan : array[1..10] of integer; { deklarasi array }
    i,banyak : integer;

begin
    {awal dari memasukkan suatu elemen ke dalam sebuah array}
    writeln ('Masukkan Banyak bilangan yg diinput (< 10)');
    readln(banyak);
    if (banyak <=10)then
       for i:= 1 to banyak do
       begin
         writeln('masukkan bilangan ke ',i);
```

```
         readln(Bilangan[i]);
       end;
    {akhir dari memasukkan suatu elemenke dalam sebuah array}

    {awal menampilkan elemen yang ada di dalam array}
    For i:= 1 to banyak do
       Writeln('Isi elemen ke – ',i, ' adalah',bilangan[i]);
    Readln;
end.
```

7.1.2 Array Multi Dimensi

Array dapat dibuat dalam lebih dari satu dimensi. Contoh cara pendeklarasian array multidimensi adalah sebagai berikut:

```
2 dimensi  a : array[1..10, 1..10] of integer;
3 dimensi  b : array[1..10, 1..10, 1..10] of integer;
```

Contoh permasalahan yang membutuhkan array multidimensi misalnya adalah representasi matriks. Berikut adalah contoh program yang merepresentasikan matriks dua dimensi.

Kode Bahasa Pascal

```
Program Tab2Dim;
{tabel integer dua dimensi (matriks)}
Type
{cara 1: 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;

procedure initM2DInt(var tabel2d:MatInt);
var
    i,j:integer;
begin
    for i:=1 to 3 do
       for j:=1 to 3 do
         tabel2d[i,j]:=i*j;
end;

procedure initM2DArr(var tabel2d:MatArr);
var
    i,j:integer;
begin
    for i:=1 to 3 do
       for j:=1 to 3 do
         tabel2d[i][j]:=i*j;
end;

procedure outM2DInt(tabel2d:MatInt);
var
    i,j:integer;
begin
    for i:=1 to 3 do
    begin
```

```
      for j:=1 to 3 do
        write('M1[',i,',',j,']= ',tabel2d[i,j]);
      writeln;
    end;
end;

procedure outM2DArr(tabel2d:MatArr);
var
    i,j:integer;
begin
    for i:=1 to 3 do
    begin
      for j:=1 to 3 do
        write('M1[',i,',',j,']=',tabel2d[i][j]);
      writeln;
    end;
end;

Begin
    initM2DInt(M1);
    initM2DArr(MA1);
    outM2DInt(M1);
    outM2DArr(MA1);
    readln;
End.
```

7.2 Array Statis dan Dinamis

Array yang telah dibahas di atas adalah array statis. Array statis adalah array yang ukurannya (jumlah elemen) telah dideklarasikan dan tidak dapat diubah (dikurangi atau ditambah) lagi.

Array dinamis adalah array yang jumlah elemennya belum ditentukan. Awalnya, jumlah elemen dalam array ini adalah nol. Ukuran array dapat ditambah (serta dipotong) dengan prosedur bawaan Pascal, yaitu `setlength`. Prosedur `setlength` akan mengatur ukuran array sesuai dengan masukan yang diberikan pada parameter. Berikut adalah contoh potongan program yang menggunakan array dinamis.

Kode Bahasa Pascal

```
var
    a : array of integer;

begin
    writeln( length(a) ); {di awal, jumlah elemen array a adalah nol}
    setlength(a,4);
    writeln( length(a) ); {elemen array a berubah menjadi 4}
    readln
end.
```

Pada contoh potongan program di atas, array a akan memiliki indeks yang dimulai dari 0 hingga 3 (ukuran array dikurangai satu).

7.3 Array sebagai Parameter Fungsi/Prosedur

Array juga dapat digunakan sebagai parameter fungsi maupun prosedur. Array sebagai parameter tidak perlu ditentukan jumlah elemennya, karena secara otomatis akan menyesuaikan dengan ukuran array yang diacu. Berikut adalah contoh programnya.

Kode Bahasa Pascal

```
program TABEL;
{latihan array Untuk memasukkan bilangan sebanyak n dan menampilkannya tanpa
dengan menggunakan prosedur}
Var
    a:array[1..10] of integer;
    banyak : integer;

procedure inittab(c:integer;var tabel:array of integer);
{mengisi array dengan assignment}
Var
    i : integer;
begin
    for i := 1 to c do
       tabel[i] := i;
end;

procedure outtab(c:integer;tabel:array of integer);
{traversal: print}
var
    i:integer;
begin
    for i:=1 to c do
       writeln('i=',i,' tabel[i]=', tabel[i]);
end;

begin
    writeln ('Masukkan Banyak bilangan yg    diinput(< 10) ');
    readln(banyak);
    if (banyak <= 10) then
    begin
       inittab(5,a);
       outtab(5,a);
    end;
    readln;
end.
```

Modul 8 *Searching* dan *Sorting*

Tujuan
Praktikan diharapkan dapat : o Memahami dan dapat mengaplikasikan algoritma *sequential dan binary search.* o Memahami dan dapat mengaplikasikan algoritma *sorting* seperti *bubble sort, insertion sort, selection sort* dan *counting sort.*

8.1 *Searching*

Diberikan sebuah array yang berisi banyak data nama-nama mahasiswa, kita dapat mencari nama seseorang yang kita inginkan dengan metode-metode *searching*. Berikut adalah contoh program sederhana pencarian nama seseorang, dengan data nama-nama orang yang telah disimpan sebelumnya.

Pseudo Code	Kode Bahasa Pascal
Program pencarian Kamus nama : array [1..5] of string[50] Nama_dicari : string I : integer Algoritma Nama[1]← 'harry' Nama[2]← 'hermione' Nama[3]← 'ron' Nama[4]← 'neville' Nama[5]← 'luna' Input(nama_dicari) i traversal [1..5] If (nama_dicari=nama[i]) then Output('Nama ditemukan')	Program pencarian; Var nama : array [1..5] of string[50]; nama_dicari : string; i : integer; begin {pengisian elemen dengan nama} nama[1]:= 'harry'; nama[2]:= 'hermione'; nama[3]:= 'ron'; nama[4]:= 'neville'; nama[5]:= 'luna'; {awal pencarian} writeln('masukkan nama yang dicari'); readln(nama_dicari); {mulai mencari} for i:=1 to 5 do {dicari satu dalam elemen array} If (nama_dicari=nama[i]) then writeln('Nama ditemukan'); { jika ditemukan } end.

Contoh di atas adalah program yang sangat sederhana untuk mencari (*search*) data dalam *array*. Jika kita mencari 'luna', maka program akan mengeluarkan 'Nama ditemukan' karena nama luna berada dalam array nama yang ingin dicari. Hal yang berbeda akan berbeda jika kita mencari 'cedric' misalnya, karena nama 'cedric' tidak terdapat dalam array nama. Dalam hal ini, program tersebut akan langsung keluar (terminasi) tanpa mengeluarkan *output* apapun. Padahal, akan sangat membantu jika kita memberikan keterangan pada pengguna bahwa data yang dicari tidak ditemukan. Pada contoh selanjutnya, kita akan melihat program yang dapat melakukan hal ini dengan memanfaatkan sebuah variable bertipe Boolean. Variabel ini akan diset true jika data ditemukan, dan tetap bernilai false jika hingga ujung pencarian array, data tidak juga ditemukan.

Kemudian, perhatikan bahwa jika kita ingin mencari ‘harry’, maka sebenarnya program akan menemukan data ‘harry’ pada iterasi pertama. Kondisinya, walaupun tujuan dari program ini telah selesai (yaitu menemukan data yang ingin dicari), namun sayangnya program sederhana di atas akan tetap meneruskan mencari keseluruhan *array*. Hal ini juga akan diperbaiki pada pembahasan berikut, yaitu program akan berhenti ketika data yang ingin dicari telah ditemukan.

8.2 Metode Pencarian Datum

Pada modul ini, akan dijelaskan dua dari sekian banyak metode pencarian yang ada, yaitu pencarian berurut (*sequential search*) dan pencarian biner (*binary search*).

8.2.1 *Sequential Search*

Pada pembahasan berikut, diberikan contoh skema *sequential search* dengan serta tanpa bantuan variabel Boolean. Kedua contoh program berikut ini sama-sama melakukan tugas yang sama: menampilkan keterangan ‘data ditemukan pada indeks ke-‘ jika data ditemukan, dan menampilkan keterangan ‘data tidak ditemukan’ jika hal sebaliknya terjadi.

Skema sequential search **tanpa** boolean

Pseudo Code	Kode Bahasa Pascal
Procedure SEQSearch1(T:TabIntN,x:integeri/o idx:integer) kamus i : integer algoritma i←1 While ((i<N) and (T[i]<>x)) do i←i+1 If T[i]=x then output(‘data ditemukan pada indeks ke-‘,i) Else output(‘data tidak ditemukan’)	Procedure SEQSearch1(T:TabInt;N,x:integer;var idx:integer); Var i : integer; Begin i:=1; While ((i<N) and (T[i]<>x)) do Begin i:=i+1; end; If T[i]=x then writeln(‘data ditemukan pada indeks ke-‘,i); Else writeln(‘data tidak ditemukan’); end;

Skema sequential search **dengan** boolean (benar)

Pseudo Code	Kode Bahasa Pascal
Procedure SEQSearch2(T:TabIntN,x:integer i/o idx:integerfound:boolean) kamus i : integer algoritma i←1 found←false While (i<=N and not(found)) do If T[i]=x then found←true else i←i+1	Procedure SEQSearch2(T:TabInt;N,x:integer; var idx:integer;found:boolean); Var i : integer; Begin i:=1; found:=false; While (i<=N and not(found)) do Begin If T[i]=x then found:=true else i:=i+1; end;

`` If found then output('data ditemukan pada indeks ke-',i) Else output('data tidak ditemukan') ```	``` If found then writeln('data ditemukan pada indeks ke-',i); Else writeln('data tidak ditemukan'); end; end; ```

Variasi lain dari sequential search adalah Skema sequential search dengan sentinel. Coba perhatikan program berikut :

Pseudo Code	Kode Bahasa Pascal
`` Procedure SEQSearchSentinel(T:TabInt; NMax,x:integer; i/o idx:integer) kamus i : integer algoritma i←1 T[NMax+1]←x {pasang sentinel} While (T[i]<>x) do i←i+1 If i < NMax +1 then output('data ditemukan pada indeks ke-',i) Else output('data tidak ditemukan') ```	``` Procedure SEQSearchSentinel(T:TabInt; NMax,x:integer; var idx:integer); Var i : integer; Begin i:=1; T[NMax+1]:=x; {pasang sentinel} While (T[i]<>x) do Begin i:=i+1; end; If i< NMax +1 then writeln('data ditemukan pada indeks ke-',i); Else writeln('data tidak ditemukan'); end; ```

8.2.2 *Binary Search*

Skema *binary search* adalah skema pencarian yang hanya dapat dilakukan pada data yang **sudah terurut**, baik ascending, maupun descending. Pada sebagian besar kasus, proses *binary search* lebih cepat bila dibandingkan dengan sequential search. Mengapa demikian?

Skema pencarian *binary search* dengan boolean

Pseudo Code	Kode Bahasa Pascal
`` Procedure BinSearch1(input T:tabint,n:integer, x:integer, Output found:boolean, ix:integer) kamus Atas,bawah,tengah:integer algoritma Atas←1 Bawah←n Found←false Ix←0 While (atas<=bawah and not(found)) do Tengah←(atas+bawah)div 2 If T[tengah]=x then ```	``` Procedure BinSearch1(input T:tabint, n:integer, x:integer, Output found:boolean, ix:integer); var Atas,bawah,tengah:integer; begin Atas:=1; Bawah:=n; Found:=false; Ix=0; While (atas<=bawah and not(found)) do begin Tengah:=(atas+bawah)div 2; If T[tengah]=x then ```

```
       Found←true
       Ix←tengah
    Else if T[tengah]<x then
       atas←tengah+1
    Else
       bawah←tengah-1
```

```
       Found:=true;
       Ix:=tengah;
    Else if T[tengah]<x then
       atas:=tengah+1
    Else
       bawah:=tengah-1;
 end;
{atas>bawah or found,
 harga found menentukan hasil pencarian}
end;
```

Skema binary search tanpa Boolean

Pseudo Code	Kode Bahasa Pascal
Procedure BinSearch2(input T:tabint, n:integer, x:integer, Output ix:integer) kamus Atas,bawah,tengah:integer algoritma Atas←1 Bawah←n Tengah←(atas+bawah)div 2 While (atas<bawah and T[tengah]<>x) do if T[tengah]<x then atas←tengah+1 Else bawah←tengah-1 Tengah←(atas+bawah)div 2 if (T[tengah]=x) then ix←tengah else ix←0	Procedure BinSearch2(input T:tabint, n:integer, x:integer, Output ix:integer); Var Atas,bawah,tengah:integer; Begin Atas:=1; Bawah:=n; Tengah:=(atas+bawah)div 2; While (atas<bawah and T[tengah]<>x) do begin if T[tengah]<x then atas:=tengah+1 Else bawah:=tengah-1; Tengah:=(atas+bawah)div 2; end; {atas>=bawah or T[tengah]=x} if (T[tengah]=x) then ix:=tengah else ix:=0; end;

8.3 *Sorting*

Dalam ilmu komputer, terdapat banyak metode pengurutan data (*sorting*) yang telah ditemukan dengan segala kekurangan dan kelebihannya masing-masing. Dalam modul ini, akan dibahas empat metode pengurutan data yang sederhana dan fundamental, yaitu *bubble sort*, *insertion sort*, *selection sort* dan counting sort.

Di bawah ini adalah contoh program utama yang akan memanggil prosedur pengurutan data tertentu. Metode-metode pengurutan data akan dijelaskan pada subbab berikutnya dalam bentuk kode procedure atau function.

Pseudo Code	Kode Bahasa Pascal
Program pengurutan {program ini berfungsi sebagai main program yang akan memanggil beberapa prosedur sorting}	Program pengurutan; {program ini berfungsi sebagai main program yang akan memanggil beberapa prosedur sorting}

```
CONST
 Nmax = 100
TYPE
 Tabel = array[1..Nmax] of integer
kamus
 T   : Tabel    {tabel integer}
 N : integer
 I,x  : integer Kondisi : boolean

algoritma
 While (not(kondisi)) do
    input(N)
    if  ((N>=1)  and  (N<=Nmax+1))
then
       kondisi←true

 i traversal [1..n]
    input(T[I])

{lakukan pemanggilan prosedur sort
disini……!}
```

```
CONST
 Nmax = 100
TYPE
 Tabel = array[1..Nmax] of integer;
VAR
 T   : Tabel;    {tabel integer}
 N : integer;
 {indeks efektif,
    maksimum tabel yang terdefinisi 1 ≤ N ≤
Nmax+1}
 I,x  : integer; Kondisi : boolean;

Begin
 While (not(kondisi)) do
 Begin
    readln(N);
    if ((N>=1) and (N<=Nmax+1)) then
       kondisi:=true;
 end;

 for I :=1 to N do
 begin
    write('masukkan nilai ke ',I,' = ');
    readln(T[I]);
 end;

{lakukan    pemanggilan    prosedur    sort
disini……!}

End.
```

8.4 Metode Pengurutan Data

8.4.1 Bubble Sort

Pseudo Code	Kode Bahasa Pascal
procedure bubbleSort(i/o a : tabel) kamus int i,j, tmp : integer algoritma i traversal [size..2] J traversal [2..i] if (a[j-1] > a[j]) then tmp ← a[j-1] a[j-1] ← a[j] a[j] ← tmp	procedure bubbleSort(var a : tabel); var int i,j, tmp : integer; begin for i:= size downto 2 do for j:= 2 to i do if (a[j-1] > a[j]) then begin tmp := a[j-1]; a[j-1] := a[j]; a[j] := tmp; end; end;

8.4.2 Insertion Sort

Bekerja mirip seperti bubble sort, metode insertion sort juga membandingkan antardua data. Bedanya, sebuah datum (satu buah 'data') yang tidak berada pada tempatnya akan langsung dipindah atau disisipkan (*insert*) pada tempat yang selayaknya saat itu juga.

Pseudo Code	Kode Bahasa Pascal
procedure insertionSort(I/o a : tabel) kamus i,j,tmp :integer algoritma i traversal [2..size] tmp←a[i] j←i while ((j>1) and (tmp < a[j-1])) do a[j]←a[j-1] j←j-1 a[j]<-tmp	procedure insertionSort(var a : tabel); var i,j,tmp :integer; begin for(i:=2 to size)do begin tmp:=a[i]; j=i; while ((j>1) and (tmp < a[j-1])) do begin a[j]:=a[j-1]; j:=j-1; end; a[j]:=tmp end; end;

8.4.3 Selection Sort

Metode ini akan memilih (*select*) nilai terkecil dari setiap iterasi, kemudian menempatkannya pada tempat yang seharusnya.

Pseudo Code	Kode Bahasa Pascal
Procedure selectionSort(i/o a : tabel) kamus i,j,min,tmp :integer algoritma i traversal [1..size-1] min ← i j traversal [i + 1 .. size] if (a[j] < a[min]) then min ← j tmp ← a[min] a[min] ← a[i] a[i] ← tmp	Procedure selectionSort(var a : tabel); var i,j,min,tmp :integer; begin for i:=1 to size-1 do begin min := i; for j := i + 1 to size do if (a[j] < a[min]) then min := j; tmp := a[min]; a[min] := a[i]; a[i] := tmp; end; end;

8.4.4 Counting sort

Berbeda dengan ketiga jenis metode pengurutan data sebelumnya yang bekerja dengan cara membandingkan dua data, metode counting sort tidak bekerja dengan cara demikian. Counting sort bekerja selayaknya perhitungan suara dalam pemilu, yaitu menghitung frekuensi kemunculan suatu datum, dan menyimpannya pada array.

Pseudo Code	Kode Bahasa Pascal
procedure CountSort(i/o TabInt:tabel) kamus i,j,k, : integer TabCount : array[Min..Max] of integer	procedure CountSort(var TabInt:tabel); var i,j,k, : integer; {min dan max adalah batas minimum dan maksimum, harga yang tersimpan dalam T, harus diketahui} TabCount : array[Min..Max] of integer;

```
Algoritma
 i traversal [min..max]
    TabCount[i]←0

 i traversal [i..n]
    TabCount[TabInt[i]]←
       TabCount[TabInt[i]]+1

 k<-0
 i traversal [min..max]
    if TabCount[i] <> 0 then
       j                  traversal
[1..TabCount[i]]
    k←k+1
         TabInt[k]←i
```

```
begin
 for  i:=Min to Max   do
 begin
    TabCount[i]:=0;
 end;

 for  i:=1 to n  do
 begin

 TabCount[TabInt[i]]:=TabCount[TabInt[i]]+1;
 end;

 k:=0;
 for i:=Min to Max do
    if TabCount[i] <> 0 then
       for j:=1 to TabCount[i] do
       begin
         k:=k+1;
         TabInt[k]:=i;
       end;
end;
```

Catatan:

- Lakukan pengujian terhadap semua prosedur sorting tersebut dengan menggabungkan antara program utama pengurutan dan semua prosedurnya.
- Manakah yang lebih mangkus? Selection, insertion, atau bubble sort?
- Lakukan pengujian terhadap
  - Karakter (mengurutkan abjad) dan
  - String (mengurutkan nama).

Soal

1. Buat sebuah prosedur untuk mencari hasil perkalian matriks. Inputan matriks(kolom,baris,dan isi matriks) terserah kepada user tapi dengan syarat jumlah kolom matriks pertama harus sama dengan jumlah baris matrik kedua

   Contoh keluaran program
   Baris matrik 1: 3
   Kolom matrik 1: 2
   Baris matrik 2: 2
   Kolom matrik 2: 1

   Isi matrik 1

   matrik1 [1,1] = 1

   matrik1 [1,2] = 2

   matrik1 [2,1] = 3

   matrik1 [2,2] = 4

   matrik1 [3,1] = 5

   matrik1 [3,2] = 6

   Isi matrik 2

   matrik2 [1,1] = 1

   matrik2 [2,1] = 2

   Hasil perkalian matriks
   Hasil [1,1] = 5
   Hasil [2,1] = 11
   Hasil [3,1] = 17

2. Buatlah sebuah program untuk mengurutkan array of record berdasarkan field tertentu!
3. Buatlah program untuk melakukan perhitungan modus pada suatu array berdimensi 2!
4. Buatlah program yang dapat mengurutkan angka secara descending!

   Contoh Input :

   masukkan jumlah angka : 7

   angka-1 : -2

   angka-2 : 43

   angka-3 :2

   angka-4 : 56.5

   angka-5 : -28.4

angka-6 : 243

angka-7 : 67

Contoh Output :

urutan angka yang akan dimakan oleh Pepi adalah :

243, 67, 56.5, 43, 2, -2, -28.4

5. Buatlah program untuk melakukan inisialisasi array berjumlah 100 item dengan nilai random kemudian lakukan pengurutan dengan metode sorting tertentu!

Modul 9 *Sequential File*

Tujuan
Praktikan diharapkan dapat : o Memahami bentuk-bentuk file sekuensial dalam algoritma. o Memahami bentuk algoritma konsolidasi. o Membuka file, menutup file, menuliskan data ke file, menampilkan data, menampilkan data baik dalam bentuk file skuensial dalam algoritma maupun dalam bahasa Pascal.

Pada modul ini, kita akan belajar bagaimana menyimpan data pada sebuah file. Sebelum membahas lebih lanjut perlu diketahui beberapa operasi terhadap file yang dapat dilakukan, yaitu

- Assign
  Berfungsi untuk membuka berkas atau file yang akan dioperasikan.
  Sintaks: `Assign(Variabel_File,'File_Name')`
- Rewrite
  Berfungsi untuk membuat file baru dan menempatkan posisi kursor/pointer pada awal berkas (posisi nol). Perlu diperhatikan, bila ternyata file tersebut sudah ada maka secara otomatis data sebelumnya akan terhapus.
  Sintaks: `Rewrite(Variable_File)`
- Write
  Berfungsi untuk menuliskan data dari variabel penampung ke dalam file. Posisi pointer akan secara otomatis maju ke posisi selanjutnya setelah prose penulisan selesai.
  Sintaks: `Write(Variable_File,Variable_Penampung)`
- Read
  Berfungsi untuk membaca data dari file dan selanjutnya disimpan ke dalam variable penampung. Posisi pointer juga secara otomatis akan maju.
  Sintaks :
  ```
  Read (Variable_File,Variable_Penampung)
  ```
- Reset
  Berfungsi mengembalikan posisi pointer ke awal file. Sintaks :
  ```
  Reset (Variable_File)
  ```
- Seek
  Berfungsi untuk menempatkan pointer pada posisi data tertentu yang ada di dalam file.
  Sintaks :
  ```
  Seek (Variable_File,Posisi_Pointer)
  ```
- FileSize
  Berfungsi untuk mendapatkan jumlah data yang tersimpan dalam file.
  Sintaks :
  ```
  Variable_Penampung := FileSize(Variable_File)
  ```

- FilePos
  Berfungsi untuk mendapatkan posisi pointer pada saat eksekusi operasi ini dilakukan.
  Sintaks :
  ```
  Variable_Penampung := FilePos(Variable_File)
  ```

- Erase
  Berfungsi untuk menghapus berkas.
  Sintaks :
  ```
  Erase (Variable_File)
  ```

- Rename
  Berfungsi untuk mengganti nama berkas yang sudah ada.
  Sintaks :
  ```
  Rename(Variable_File,'New_Name')
  ```

- EOF
  Berfungsi untuk mengetahui apakah posisi pointer sudah berada di data terakhir (TRUE) atau tidak (FALSE).
  Sintaks :
  ```
  Variable_Penampung := EOF (Variable_File)
  ```

- Close
  Berfungsi untuk menutup berkas bila tidak akan dioperasikan lagi.
  Sintaks :
  ```
  Close(Variable_File)
  ```

Berikut contoh Penerapan File Sekuensial untuk membuat data Mahasiswa :

Pseudo Code	Kode Bahasa Pascal
Program DataMahasiswa Type Mahasiswa = < NIM : String Nama : String Nilai : Real > Kamus ArsipMhs : file Of Mahasiswa RekMhs : Mahasiswa I :integer Algoritma Assign(ArsipMhs,"MHS.DAT") Rewrite(ArsipMhs) While i<=5 do input(RekMhs.Nim) input(RekMhs.Nama) input(RekMhs.Nilai) write(ArsipMhs,RekMhs) Reset (ArsipMhs)	Program DataMahasiswa Uses crt; Type Mahasiswa = Record NIM : String; Nama : String; Nilai : Real; End; Var ArsipMhs : file Of Mahasiswa; RekMhs : Mahasiswa; I :integer; Begin {Program Utama} Assign(ArsipMhs,"MHS.DAT"); Rewrite(ArsipMhs); {Pada contoh kali ini kita akan menginputkan 5 data mahasiswa} While i<=5 do Begin Write("Input Nim : ");readln(RekMhs.Nim); Write("Input Nama : ");readln(RekMhs.Nama); Write("Input Nilai :

```
    i traversal [1..5]
       Read(ArspiMhs,RekMhs)
       Write("Nim
:",RekMhs.Nim)
       Write("Nama
:",RekMhs.Nama)

    Write("Nilai:",RekMhs.Nilai
)
    close(ArsipMhs)
```

```
");readln(RekMhs.Nilai);
    write(ArsipMhs,RekMhs)
    end;

    Reset (ArsipMhs);
    {Membaca isi data yang kita tulis di file
MHS.DAT}
    For i:=1 to 5 do
    Begin
       Read(ArspiMhs,RekMhs);
       Write("Nim  : ",RekMhs.Nim);
       Write("Nama : ",RekMhs.Nama);
       Write("Nilai : ",RekMhs.Nilai);
    End;
    Close(ArsipMhs);
    Readln;
End.
```

Penerapan File Sekuensial untuk menghitung mencari nilai tertinggi dan terendah dari file

Pseudo Code	Kode Bahasa Pascal
Program Mencari Nilai Max dan Min dari File Type Mahasiswa = < NIM : String Nama : String Nilai : Real > kamus ArsipMhs : file Of Mehasiswa RekMhs,RekMhsMax,RekMhsMin : Mahasiswa algoritma RekMhsMax.Nilai←0 RekMhsMin.Nilai←100 Assign(ArsipMhs,"MHS.DAT") Reset(ArsipMhs) While not Eof(ArsipMhs) do Read(ArsipMhs,RekMhs) If RekMhsMax.Nilai<=RekMhs.Nilai then RekMhsMax←RekMhs If RekMhsMin.Nilai>=RekMhs.Nilai then RekMhsMin←RekMhs Writeln(RekMhsMax.Nilai,RekMhsMax.Nim) Writeln(RekMhsMin.Nilai,RekMhsMin.Nim) Close(ArsipMhs)	Program Mencari Nilai Max dan Min dari File Uses crt; Type Mahasiswa = Record NIM : String; Nama : String; Nilai : Real; End; Var ArsipMhs : file Of Mehasiswa; RekMhs,RekMhsMax,RekMhsMin : Mahasiswa; {Dengan Asumsi Range nilai 0-100} Begin {Program Utama} RekMhsMax.Nilai:=0; RekMhsMin.Nilai:=100; Assign(ArsipMhs,"MHS.DAT"); Reset(ArsipMhs); While not Eof(ArsipMhs) do begin Read(ArsipMhs,RekMhs) If RekMhsMax.Nilai <= RekMhs.Nilai then RekMhsMax=RekMhs; If RekMhsMin.Nilai>=RekMhs.Nilai then RekMhsMin=RekMhs; end; Writeln("Mahasiswa dengan nilai tertinggi: ",RekMhsMax.Nilai,"dengan Nim",RekMhsMax.Nim); Writeln("Mahasiswa dengan nilai terendah:

	",RekMhsMin.Nilai,"dengan Nim",RekMhsMin.Nim); Close(ArsipMhs); End.

Penerapan File Sekuensial untuk menghapus data dari file

Pseudo Code	Kode Bahasa Pascal
Program hapus Type Mahasiswa = Record NIM : String Nama : String Nilai : Real kamus ArsipMhs,TempArsipMhs : file Of Mehasiswa RekMhs : Mahasiswa X:String algoritma Assign(ArsipMhs,"MHS.DAT") Assign(TempArsipMhs,"TEMP.DAT") Reset(ArsipMhs) Rewrite(TempArsipMhs) input (x) While not Eof(ArsipMhs) do Read(ArsipMhs,RekMhs) If(RekMhs.Nim <> x)then Write(TempArsipMhs,RekMhs) Erase(ArsipMhs) Rename(TempArsipMhs,"MHS.DAT")	Program hapus; Uses crt; Type Mahasiswa = Record NIM : String; Nama : String; Nilai : Real; End; Var ArsipMhs,TempArsipMhs : file Of Mehasiswa; RekMhs : Mahasiswa; X:String; Begin {Program Utama} Assign(ArsipMhs,"MHS.DAT"); Assign(TempArsipMhs,"TEMP.DAT"); Reset(ArsipMhs); Rewrite(TempArsipMhs); Write("Masukkan Nim Mahasiswa yang ingin di hapus :"); readln(x); While not Eof(ArsipMhs) do begin Read(ArsipMhs,RekMhs) If(RekMhs.Nim <> x)then Write(TempArsipMhs,RekMhs); end; Erase(ArsipMhs); Rename(TempArsipMhs,"MHS.DAT"); End.

Penerapan File Sekuensial untuk mengecek file

Pseudo Code	Kode Bahasa Pascal
Program Cek File Type Mahasiswa = Record NIM : String Nama : String Nilai : Real kamus ArsipMhs: file Of Mahasiswa Filename :string function cekfile(namafile:string) :	Program Cek File; Uses crt; Type Mahasiswa = Record NIM : String; Nama : String; Nilai : Real; End; Var ArsipMhs: file Of Mahasiswa; Filename :string; function cekfile(namafile:string):boolean;

```
boolean
    assign(ArsipMhs,namafile)
    {$I-}
    reset(ArsipMhs)
    {$I+}
    if IOresult=0 then
       cekfile<-false
    else
       cekfile<-true



algoritma
    input(Filename)
    If(cekfile(filename))then
       output("File sudah ada")
    Else
       output ("File belum ada")
```

```
begin
    assign(ArsipMhs,namafile);
    {$I-}
    reset(ArsipMhs);
    {$I+}
    if IOresult=0 then
       cekfile:=false
    else
       cekfile:=true;
end;

Begin {Program Utama}
    Writeln("Inputkan Nama File : ");
    readln(Filename);
    If(cekfile(filename))then
       Write("File sudah ada")
    Else
       write ("File belum ada");
    Readln;
End.
```

Menggabungkan Dua File Menjadi Satu (*Merge*)

Pseudo Code	Kode Bahasa Pascal
Program GabungData Type Mahasiswa = < NIM : String Nama : String Nilai : Real > Var ArsipMhs1,ArsipMhs2,ArsipMhsOut : file Of Mahasiswa RekMhs1,RekMhs2,RekMhsOut: Mahasiswa algoritma Assign(ArsipMhs1,"MHS1.DAT") Reset(ArsipMhs1) Assign(ArsipMhs2,"MHS2.DAT") Reset(ArsipMhs2) Assign(ArsipMhsOut,"MHSOUT.DAT") Rewrite(ArsipMhsOut) While not Eof(ArsipMhs1) do read(ArsipMhs1,RekMhs1) write(ArsipMhsOut,RekMhs1) seek(ArsipMhsOut,filesize(ArsipM hsOut)) While not Eof(ArsipMhs2) do read(ArsipMhs2,RekMhs2) write(ArsipMhsOut,RekMhs1) Close(ArsipMhs1)	Program GabungData; Uses crt; Type Mahasiswa = Record NIM : String; Nama : String; Nilai : Real; End; Var ArsipMhs1,ArsipMhs2,ArsipMhsOut : file Of Mahasiswa; RekMhs1,RekMhs2,RekMhsOut: Mahasiswa; Begin {Program Utama} Assign(ArsipMhs1,"MHS1.DAT"); Reset(ArsipMhs1); Assign(ArsipMhs2,"MHS2.DAT"); Reset(ArsipMhs2); Assign(ArsipMhsOut,"MHSOUT.DAT"); Rewrite(ArsipMhsOut); While not Eof(ArsipMhs1) do begin read(ArsipMhs1,RekMhs1); write(ArsipMhsOut,RekMhs1); end; seek(ArsipMhsOut,filesize(ArsipMhsOut)); While not Eof(ArsipMhs2) do begin read(ArsipMhs2,RekMhs2); write(ArsipMhsOut,RekMhs1); end;

```
    Close(ArsipMhs2)
    Close(ArsipMhsOut)
```

```
    Close(ArsipMhs1);
    Close(ArsipMhs2);
    Close(ArsipMhsOut);
End.
```

Soal

1. Buat sebuah program untuk menampilkan data mahasiswa tertentu saja untuk data mahasiswa dengan menggunakan file sekuensial!

2. Buat sebuah program untuk menampilkan rata-rata nilai mahasiswa apabila satu mahasiswa dapat memiliki lebih dari satu mata kuliah!

3. Buat sebuah program untuk mengubah data Mahasiswa berdasarkan NIM- nya!

4. Buatlah sebuah program yang mampu membaca dan menulis data diary dalam mode text. Setiap elemen diary terdiri dari tanggal, judul dan isi diary. Main menu program ini adalah :
   1. Lihat diary
   2. Tambah diary baru
   3. Exit
5. Buat sebuah program untuk mengelompokkan file pada mahasiswa berdasarkan program studi yang diketahui dari Nimnya! Contoh:

   NIM : 613060001 ada di file D3.DAT Sedangkan NIM : 113060001 ada di file S1.DAT

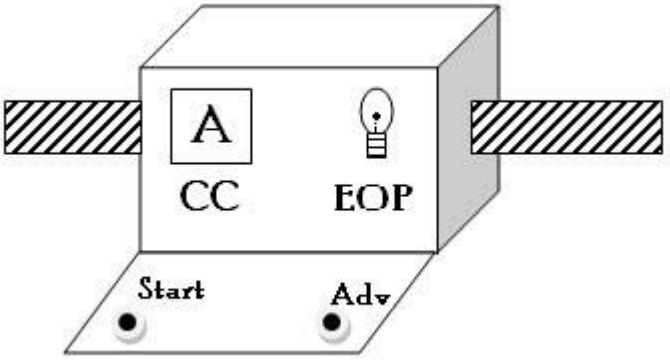
Modul 10 Mesin Abstrak

Tujuan
Praktikan diharapkan dapat : o Memahami pengertian mesin abstrak. o Memahami item-item mesin abstrak. o Memahami penggunaan mesin abstrak dalam studi kasus tertentu.

Mesin abstrak adalah yang dapat melakukan sebuah tugas namun tidak mempunyai fisik. Mesin ini hanya khayalan, oleh sebab itu disebut abstrak. Terdapat dua jenis mesin abstrak yang akan dibahas dalam modul ini, yaitu mesin karakter dan mesin pencacah.

10.1 Mesin Karakter

Kali ini akan dibahas mesin karakter, yaitu mesin yang dapat memproses karakter. Mesin abstrak akan membaca karakter per karakter yang diambil dari sebuah file teks yang disebut sebagai pitaChar.



Gambar 10.1 Visualisasi Mesin Karakter

Seperti tampak pada gambar Gambar 10.1, mesin karakter divisualisasikan sebagai sebuah mesin sederhana yang hanya memiliki dua tombol untuk dioperasikan: tombol start dan adv. PitaChar sendiri adalah "kertas" berisi karakter-karakter yang akan dibaca. Dalam gambar yang sama, pitaChar divisualisasikan sebagai persegi dengan pola strip hitam putih. Mesin abstrak kita akan membaca karakter yang tertulis dalam pitaChar ini hingga ujung pita. Penting untuk diingat bahwa mesin sederhana ini hanya mampu membaca satu karakter per satuan waktu.

Ketika tombol start ditekan (dalam program yang akan kita buat pada subbab berikutnya, cara "menekan" mesin khayalan kita adalah dengan memanggil function yang bersangkutan), mesin akan menyala dan mulai membaca karakter pertama pada pitaChar. Ketika tombol adv ditekan, mesin akan menggeser pitaChar selangkah ke kanan dan membaca karakter selanjutnya yang ada dalam pitaChar.

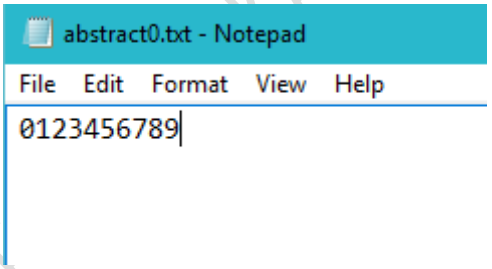
Selain dua tombol tadi, mesin sederhana ini juga dilengkapi dengan dua buah indikator: CC dan EOP.

Sebelumnya harus diketahui dulu hal – hal dasar yang harus diketahui mengenai mesin karakter. CC atau *current character* adalah layar yang menampilkan karakter yang sedang dibaca oleh mesin. Indikator EOP adalah sebuah lampu (atau dalam program kita nantinya, sebuah variabel bertipe Boolean) yang berfungsi sebagai indikator habisnya pita untuk dibaca. Lampu EOP yang tadinya padam, akan menyala (dalam program kita berarti, variabel EOP yang tadinya bernilai false akan berubah menjadi true) ketika pitaChar telah habis dibaca.

Tipe data yang paling cocok untuk menampung PitaChar adalah dengan menggunakan file bertipe char, yang dapat diimplementasikan sebagai `file of char` ataupun `textfile`, karena string adalah kumpulan dari banyak karakter. Dalam modul ini, textfile akan digunakan sebagai tipe data file.

Setiap karakter dapat diproses dengan mengakses variable CC. untuk mendapatkan CC, kita dapat memakai bantuan variable CI (*current integer*). Pada mesin karakter, dikenal ADV untuk maju satu karakter kedepan, namun tidak ada procedure untuk mundur satu karakter kebelakang. Berikut adalah contoh sederhana dari sebuah mesin karakter. CI pada mesin karakter dimulai dari angka 1, karena pada bahasa Pascal, index pertama sebuah string dimulai dari angka 1, berbeda dengan bahasa C yang dimulai dari angka 0.

Berikut adalah contoh program untuk membaca karakter yang dibaca oleh mesin abstrak dari file pitaChar, kemudian menampilkannya ke layar komputer kita. Sebelum mencoba program berikut, buatlah terlebih dahulu sebuah file bertipe txt dengan nama abstract0.txt yang di dalamnya berisi '0123456789' tanpa tanda petik.



Gambar 10.2 Isi file abstract0.txt

Pseudo Code	Kode Bahasa Pascal
kamus CC : char pitaChar : file of char procedure START; {- mesin abstrak khayalan dinyalakan - indikator layar CC berisi karakter pertama dalam pitaChar } algoritma assign(pitaChar, 'abstract0.txt') reset(pitaChar) read(pitaChar, CC) procedure ADV {- pitaChar digeser ke kanan selangkah	var CC : char; pitaChar : file of char; procedure START; {- mesin abstrak khayalan dinyalakan - indikator layar CC berisi karakter pertama dalam pitaChar } begin assign(pitaChar, 'abstract0.txt'); reset(pitaChar); read(pitaChar, CC); end; procedure ADV; {- pitaChar digeser ke kanan selangkah - indikator layar CC berisi karakter yang

- indikator layar CC berisi karakter yang sedang dibaca } algoritma read(pitaChar, CC) function EOP:Boolean algoritma if EOF(pitaChar) then EOP ← true else EOP ← false; Algoritma program utama START {mulai nyalakan mesin} while (not EOP) do {ulangi selama sinyal EOP tidak menyala} begin ADV {geser pita ke kanan selangkah} writeln(CC) {tulis apa yang tampil pada layar CC} end	sedang dibaca} begin read(pitaChar, CC); end; function EOP:boolean; begin if EOF(pitaChar) then EOP := true else EOP := false; end; begin START; {mulai nyalakan mesin} writeln(CC); while (not EOP) do {ulangi selama sinyal EOP tidak menyala} begin ADV; {geser pita ke kanan selangkah} writeln(CC); {tulis apa yang tampil pada layar CC} end; readln; end.

10.2 Mesin Pencacah

Mesin abstrak yang lain adalah mesin pencacah atau mesin integer. Mesin abstrak ini mirip dengan alat penghitung lompat tali. Hanya ada dua tombol dalam mesin sederhana ini: tombol INC untuk menambah hitungan, dan tombol RESET untuk mengembalikan hitungan menjadi nol. Selain kedua tombol tersebut, mesin ini juga dilengkapi indikator CI (*current integer*) yang memperlihatkan banyaknya hitungan yang telah dilakukan.

Gambar 10.3. Alat penghitung lompat tali, representasi nyata dari mesin pencacah

Berikut adalah contoh mesin integer yang digunakan untuk menyelesaikan kasus sederhana: menghitung panjang pitaChar. Sebelum mencoba, buatlah file abstract2.txt terlebih dahulu yang berisi ‘abcde’ tanpa tanda petik. Kemudian cobalah apakah program di bawah ini memberikan hasil

jumlah karakter dalam pitaChar yang benar, yaitu 5. Pada contoh di bawah ini, mesin integer digunakan bersama dengan mesin karakter agar dapat berfungsi.

Pseudo Code	Kode Bahasa Pascal
kamus CC : char; pitaChar : file of char; CI : integer; procedure START; algoritma assign(pitaChar, 'abstract2.txt') reset(pitaChar) read(pitaChar, CC) procedure ADV algoritma read(pitaChar, CC) procedure RESET algoritma CI ← 0 procedure INC algoritma CI ← CI + 1 function EOP:boolean; algoritma if EOF(pitaChar) then EOP ← true else EOP ← false algoritma START INC while (not EOP) do ADV INC output(CI)	var CC : char; pitaChar : file of char; CI : integer; procedure START; begin assign(pitaChar, 'abstract2.txt'); reset(pitaChar); read(pitaChar, CC); end; procedure ADV; begin read(pitaChar, CC); end; procedure RESET; begin CI := 0; end; procedure INC; begin CI := CI + 1; end; function EOP:boolean; begin if EOF(pitaChar) then EOP := true else EOP := false; end; begin START; INC; while (not EOP) do begin ADV; INC; end; writeln(CI); readln; end.

Catatan:

- Cobalah untuk melakukan perhitungan jumlah kata dengan memodifikasi program untuk menghitung panjang suatu kalimat
- Modifikasi kembali program untuk menghitung panjang suatu kalimat untuk melakukan perhitungan jumlah kemunculan suatu pasangan huruf pada kalimat.

Soal

1. Hitung Kemunculan jumlah huruf konsonan yang ada pada suatu kalimat
2. Buatlah program untuk menghitung frekuensi kemunculan suatu kata tertentu pada kalimat yang diinputkan oleh user.
3. Buatlah program untuk menghilangkan blank yang berlebihan, yaitu antara lain:
    a. Satu atau lebih blank sebelum huruf pertama yang bukan blank.
    b. Satu atau lebih blank sebelum huruf terakhir yang bukan blank.
    c. Lebih dari satu blank di antara dua buah kata
4. Buatlah program untuk melakukan konversi pada pita karakter yang mengandung bilangan biner dan mengubahnya menjadi bilangan integer!

Modul Overview

Tujuan
Praktikan diharapkan dapat : ▪ Memahami konsep assignment dan operasi variabel ▪ Memahami tipe-tipe dasar yang ada dalam bahasa pemrograman Pascal ▪ Memahami dan menerapkan konsep analisa kasus ▪ Memahami dan menerapkan konsep fungsi dan prosedur di berbagai kasus ▪ Memahami dan menerapkan konsep perulangan di berbagai kasus

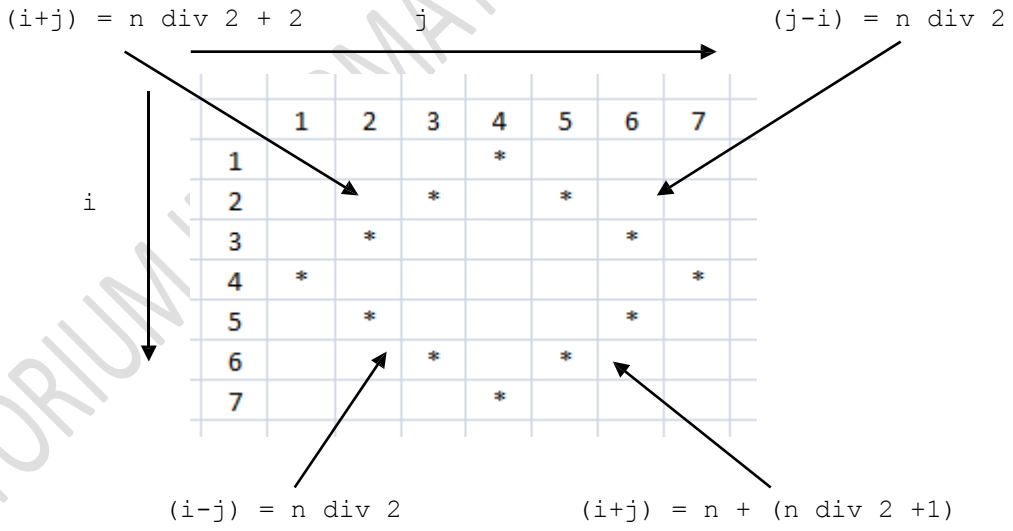
```
Program TubuhIdeal;
{ Nama File  : TubuhIdeal.pas
  Deskripsi  : Mengkategorikan berat badan yang diinputkan user ke
  dalam
  kategori ideal, di atas normal, ataupun di bawah normal
  berdasarkan nilai acuan (tinggi badan – 100 – ((tinggi badan-
  100)x0.1) }

uses crt;
var
{---Kamus Global---}

   a : real;
   b,c : real;

begin
{---Program Utama---}
       clrscr;
       write('Masukkan tinggi badan : '); readln(a);
       write('Masukkan berat badan : '); readln(b);
       c := a - 100 - ((a-100)*0.1);
       if (b = c) then
          write ('berat badan anda ideal')
       else
           if (b>c) then
                  write('berat badan anda di atas
                              normal')
           else
                write('berat badan anda di bawah normal');
        readln;
end.
```

```
Program PecahinUang;
{ Nama File  : PecahinUang.pas
  Deskripsi  : Menentukan jumlah masing-masing pecahan uang 20000,
  10000, dan 1000 terhadap inputan user }

uses crt;
var
{---Kamus Global---}

    a : longint;
    p1, p2, p3, sisa : longint;

begin
{---Program Utama---}

   clrscr;
    write('Masukkan jumlah uang yang akan diambil : ') ;readln(a);
    sisa := a;
   if (a >= 20000) then
    begin
        p1 := a div 20000;
        sisa := a - (p1*20000);
    end;
    if (sisa >= 10000) and (sisa < 20000) then
    begin
               p2 := sisa div 10000;
               sisa := sisa - (p2*10000);
     end;
   if (sisa < 10000) then begin
               p3 := sisa div 1000;
     end;

    writeln('Jumlah penarikan yang dilakukan adalah : ');
    writeln(p1, ' lembar 20.000an, ');
    writeln(p2, ' lembar 10.000an, '); writeln('dan ', p3, ' lembar
    1000an.'); readln
end.
```

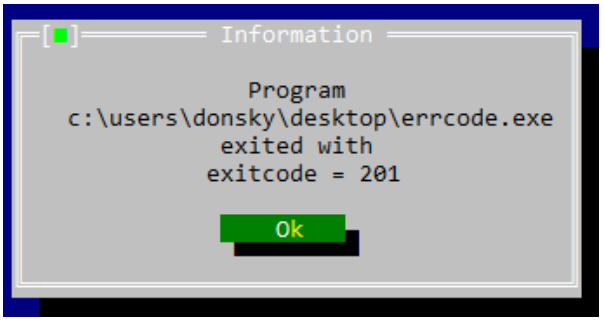
```
Program Garis;
{ Nama File    : Garis.pas
  Deskripsi : Menentukan dua koordinat titik yang diinputkan termasuk
kategori garis vertikal, horizontal, miring, ataupun bukan garis }

uses crt;
var
{---Kamus Global---}
    x1, x2, y1, y2, delta_x, delta_y : integer;
                                  m : real;
begin
{---Program Utama---}
   clrscr;
       write ('Masukkan titik x1 : '); readln(x1);
       write ('Masukkan titik y1 : '); readln(y1);
       write ('Masukkan titik x2 : '); readln(x2); write ('Masukkan
       titik y2 : '); readln(y2); delta_x := x2-x1;
       delta_y := y2-y1;
       if ( (delta_x = 0) and (delta_y = 0) ) then
                       write('Bukan merupakan suatu garis')
       else
       if ( delta_x = 0 ) then
                      write('Ini merupakan garis Vertikal')
       else
       if ( delta_y = 0 ) then
          write('Ini merupakan garis Horizontal')
       else begin
       m := delta_y / delta_x;
       write ('ini merupakan garis miring dengan gradien = ',m:0:2);
       end;
       readln;
end.
```

```
Program IPK;
{ Nama File    : ipk.pas
  Deskripsi  :  Menghitung  nilai  IP  dengan  asumsi  terdapat  4  mata
kuliah, yakni Kalkulus I, Prokom, PengLing, dan Bahasa Indonesia }
uses crt;
TYPE Nilai = record
           indeks : char;
           jumSKS : integer; nilai : integer; end;
var
   m1,m2,m3,m4 : Nilai;
   tot_sks, tot_ip : integer; IP : real;
   function konversi(a:char) : integer;
   begin
       case a of
            'A' : konversi := 4;
            'B' : konversi := 3;
            'C' : konversi := 2;
            'D' : konversi := 1;
            'E' : konversi := 0;
       end;
   end;
   function Total (s, t:integer) : integer;
   begin
        Total := s*t;
   end;
begin
     clrscr;
     write ('Masukkan Indeks Kalkulus I      : ');
     readln(m1.indeks);
     write ('Masukkan Jumlah SKS Kalkulus I  : ');
     readln(m1.jumSKS);
     write ('Masukkan Indeks PROKOM          : ');
     readln(m2.indeks);
     write ('Masukkan Jumlah SKS PROKOM      : ');
     readln(m2.jumSKS);
     write ('Masukkan Indeks PENGLING        : ');
     readln(m3.indeks);
     write ('Masukkan Jumlah SKS PENGLING    : ');
     readln(m3.jumSKS);
     write ('Masukkan Indeks BHS INDO        : ');
     readln(m4.indeks);
     write ('Masukkan Jumlah SKS BHS INDO    : '); readln(m4.jumSKS);
     m1.nilai:=konversi(m1.indeks);      m2.nilai:=konversi(m2.indeks);
     m3.nilai:=konversi(m3.indeks); m4.nilai:=konversi(m4.indeks) ;
     tot_ip  :=  Total(m1.nilai  ,  m1.jumSKS)  +  Total(m2.nilai  ,
     m2.jumSKS)
              + Total(m3.nilai , m3.jumSKS) + Total(m4.nilai ,
              m4.jumSKS);
     tot_sks := m1.jumSKS + m2.jumSKS + m3.jumSKS + m4.jumSKS;
     IP := tot_ip / tot_sks;
     writeln ('Total jumlah SKS yang diambil : ',tot_sks);
     writeln ('Total IP : ',IP:0:2);
     readln;
end.
```

```
Program Diamond;

{ Nama File    : diamond.pas
  Deskripsi  :  Menerapkan  perulangan  for  untuk  membuat  pola  belah
  ketupat
dengan ukuran sesuai inputan user }

uses crt;
var
   i,j,n : integer;
begin
clrscr;
       write('masukkan bilangan : '); readln(n);
       for i:=1 to n do
       begin
            for j:=1 to n do begin
                 if ( (i+j) = n div 2 + 2 ) OR ( (i-j) = n div 2 )
                   OR ( (i+j) = n + (n div 2 +1)) OR  ( (j-i) = n div 2
                   )  then write('*')
                 else
                     write(' ');
             end;
             writeln;
        end;
         readln;
end.
```



Lampiran 1 – *Exit Code* dalam Free Pascal



Contoh exit code 201 saat *runtime* dalam IDE Free Pascal

1	: invalid function number	160	: device write fault
2	: file not found	161	: device read fault
3	: path not found	162	: hardware failure
4	: too many open files	200	: div by zero
5	: file access denied	201	: range error
6	: invalid file handle	202	: overflow stack
12	: invalid file access code	203	: heap overflow
15	: invalid drive number	204	: invalid pointer operation
16	: cannot remove current directory	205	: floating point overflow
17	: cannot rename across	206	: floating point underflow
18	: no more files	207	: invalid floating point operation
100	: disk read error	208	: overlay manager not installed
101	: disk write error	209	: overlay file read error
102	: file not assigned	210	: object not initialized
103	: file not open	211	: call to abstract method
104	: file not open for input	212	: stream registration error
105	: file not open for output	213	: collection index out of range
106	: invalid numeric format	214	: collection overflow error
150	: disk is write protected	215	: overflow checking when doing computation with integer
151	: bad drive request structure length		
152	: drive not ready	216	: general protection fault
154	: CRC error in data	217	: invalid operation code
156	: disk seek error	227	: assertion failed
157	: unknown media type	300	: file IO error
158	: sector not found	301	: non-matched array bounds
159	: printer out of paper	302	: non-local procedure pointer

303 : procedure pointer out of scope
304 : function not implemented
305 : breakpoint error
306 : break by ctrl/c
307 : break by ctrl/break
308 : break by other process
309 : no floating point coprocessor
310 : invalid variant type operation

Lampiran 2 - Fungsi dan Prosedur Bawaan Free Pascal

Sintaks	Fungsi / Prosedur	Keterangan
`clrscr`	prosedur	[gunakan `uses crt`] Membersihkan layar dan kursor diletakkan di kiri atas *window.*
`sqr`	fungsi	pemangkatan dua (*square*).
`sqrt`	fungsi	akar pangkat dua (*square root*).
`str(i,s)`	prosedur	Mengubah nilai variabel i yang bertipe integer menjadi variabel s yang bertipe string.
`val(s,i)`	prosedur	Kebalikan dari str; mengubah nilai variabel s yang bertipe string menjadi i yang bertipe integer.
`chr(i)`	fungsi	Mengubah i yang bertipe integer ke dalam karakter ASCII. Fungsi chr(45) akan mengembalikan karakter -.
`ord(c)`	fungsi	Mengubah karakter c menjadi kode ASCII dalam integer. Fungsi ord('-') akan mengembalikan 45.
`inc(i)`	prosedur	Menambah variabel integer i dengan 1. Fungsi ini identic dengan `i := i + 1.`
`succ(i)`	fungsi	Sama seperti inc. Bedanya, succ merupakan fungsi.
`dec(i)`	prosedur	Identik dengan `i := i – 1.`
`pred(i)`	fungsi	Sama seperti dec. Bedanya, pred merupakan fungsi.
`length(s)`	fungsi	Mengembalikan panjang string s. Misalnya, length("Potter") akan menghasilkan nilai 6.

Lampiran 3 – *Programming is Fun*

Library

Jika anda pernah memanggil prosedur "clrscr;" di program yang anda buat, maka itu artinya anda sudah menggunakan salah satu prosedur yang sudah dibuat dalam library "crt". Sebelum bisa memanggil prosedur clrscr, anda harus meng-ikutkan library crt dalam program anda dengan menuliskan perintah "uses crt;"

Library adalah kumpulan dari rutin-rutin (prosedur dan fungsi) yang dirasa akan sering digunakan oleh program lain, seperti misalnya clrscr untuk membersihkan layar. Anda tidak perlu lagi membuat sendiri prosedur untuk membersihkan layar, anda cukup menggunakan prosedur clrscr yang sudah disediakan oleh pembuat bahasa pemrograman Pascal.

Beberapa prosedur lain yang akan anda temukan dalam listing program Pong.pas di bagian 10.2 adalah :

- GotoXY : pergi ke posisi X dan Y tertentu pada sebuah layar
- Inc : Increase, atau menambahkan 1 ke sebuah variable integer
- Dec : Decrease, atau menambahkan -1 ke sebuah variable integer
- Randomize : Menyiapkan sebuah seed angka random, cukup dipanggil satu kali saja.
- Random : Mengeluarkan angka secara acak dari sebuah range. Misal dengan memanggil fungsi Random(100) maka keluarannya bisa saja sebuah angka dari mulai 0 sampai 100.
- Keypressed : Digunakan untuk mendteksi kejadian sebuah penekanan tombol keyboard. Biasanya penggunaannya dibarengi dengan pemanggilan fungsi readkey (untuk membaca tombol yang barusan ditekan)
- readkey : Menerima inputan satu buah tombol dari keyboard. Tidak seperti read/readln, readkey hanya menerima satu

inputan saja (tidak menunggu ada penekanan tombol enter). Output dari fungsi readkey adalah nilai karakter dari tombol
yang di tekan (tipe datanya : char).

- Delay : Membuat program menunggu sampai batas waktu tertentu (seperti di pause). Waktunya dihitung dalam satuan millisecond (ms).

Listing program Pong.Pas

Berikut adalah contoh sebuah game sederhana yang ditulis kedalam bahasa Pascal.

```
program Pascalpongby0263;
uses crt;
type
     TipePosisi =
     record x, y :
       integer;
  end;

const
  panjangPapan = 4;
var
  player1, player2 : TipePosisi;
  i    : integer;
  ch : char;
  posBola, arahBola : TipePosisi;
  playerKalah, computerKalah : boolean;

  procedure Gambar;
  begin
    clrscr; (* bersihkan layar, biar ada efek "perubahan"
    *) (* gambar player*)
    for i:=0 to panjangPapan-1 do begin
      gotoXY(player1.x, player1.y + i);  write('|');
      gotoXY(player2.x, player2.y + i);  write('|');
    end;
```

```
    (*gambar bola*)
    gotoXY(posBola.x, posBola.y); write('0');
  end;

  procedure updatePosisi;
  begin
    if   (keypressed)   then
      begin ch := readkey;
      case ch of
        'w' : dec(player1.y);
        's' : inc(player1.y);
      end;
      (* normalisasi player 1 *)
      if (player1.y <= 1) then player1.y := 1;
      if (player1.y >= 24-panjangPapan +1) then player1.y :=
24- panjangPapan+1;
    end;
  end;

  procedure updateBola;
  begin
    posBola.x := posBola.x + arahBola.x;
    posBola.y := posBola.y + arahBola.y;
    if (posBola.y >= 24) or (posBola.y <= 1)
      then arahBola.y := arahBola.y * -1;
    if (posBola.x >= 80) then computerKalah := true;
    if  (posBola.x <= 1)  then  playerKalah  :=
    true; (* cek tabrakan bola vs. player1 *)
    if (posBola.x = 4) then

      if (posBola.y >= player1.y) and
         (posbola.y <= player1.y + panjangPapan)
         then arahBola.x := arahBola.x * -
    1; (* cek tabrakan bola vs. player2 *)
    if (posBola.x = 76) then
      if (posBola.y >= player2.y) and
                (posbola.y <= player2.y +
                     panjangPapan)
          then arahBola.x := arahBola.x * -
                          1;
  end;
```

```
  procedure UpdateKomputer;
  var ran : integer;
  begin
    ran := random(100);
    if (ran <= 50) then begin (* % chance komputer bergerak *)
      if (posBola.y <= player2.y + (panjangPapan div 2))
      then dec(player2.y)
      else inc(player2.y);
    end;
    (* normalisasi player 2 *)
    if (player2.y <= 1) then player2.y := 1;
    if (player2.y >= 24-panjangPapan +1)
      then player2.y := 24-panjangPapan+1;
  end;



(* begin utama *)
begin
  Randomize;
  playerKalah := false; computerKalah := false;
  player1.x  :=  3;  player1.y  :=
  10;  player2.x  :=  77;  player2.y
  :=   10;   posBola.x   :=   40;
  posBola.y := 10; arahBola.x:= -
  1; arahBola.y := 1;

  repeat
    delay(100); (* di delay tiap frame, biar ga terlalu cepet *)
    updatePosisi;
    Gambar;
    updateBola;
    updateKompute
    r;
  until (ch = #27) or playerKalah or computerKalah;
  (* charakter ke 27 adalah kode untuk tombol ESCAPE *)
  if (playerKalah) then writeln('cupu ah..');
  if (computerKalah) then writeln('wah kamu jago..');
  readln;
end.
```

Tugas

Siapa yang tidak kenal game snake? Dalam game ini kita harus mengendalikan seekor ular yang harus makan apel (ularnya vegetarian??) untuk menambah score. Namun semakin banyak apel yang dimakan, semakin panjang pula tubuh sang ular. Ular akan mati ketika ia (kepalanya) menabrak bagian tubuhnya sendiri.

Sederhana sekali bukan? Bisakah anda membuat game tersebut dengan bahasa pemrograman Pascal?

Daftar Pustaka

[1] Anonim. *Ayo Membuat Program Pascal / Dasar – Dasar Pemrograman*. http://id.wikibooks.org/wiki/Ayo_Membuat_Program_Pascal/Dasar-Dasar_Pemrograman (diakses tanggal 7 September 2012).

[2] Liem, Inggriani. 1990. *Diktat Kuliah IF223 Algoritma dan Pemrograman.* Bandung: ITB.

[3] Munir, Rinaldi. 2007. *Algoritma & Pemrograman Dalam Bahasa Pascal dan C.* Bandung: Informatika.

[4] Windra Swastika, Fauzan Joko. 2004. *Referensi Pemrograman Bahasa Pascal*. Departemen Pendidikan Nasional RI.

informatics lab