Pseudopascal
(Versi Olimpiade Sains Bidang Informatika/Komputer)
Penulis: Suryana Setiawan, Ketua Pembina TOKI
Tgl update: 18 Mei 2006

A. Pengantar

Mengingat dalam seleksi tertulis Olimpiade Informatika/Komputer algoritma-algoritma yang terkait dalam soal-soal tersebut dituliskan dengan Pseudocode Pascal, atau selanjutnya kita singkat dengan Pseudopascal saja, maka kami merasa perlu untuk mengeluarkan dokumen untuk dijadikan acuan peserta untuk memahaminya. Dokumen ini diharapkan juga menjadi acuan peserta dalam mempersiapkan diri dalam seleksi tersebut maupun para pembina dalam mengarahkan pelatihan siswa-siswanya.

Bagi pembimbing atau peserta seleksi yang cukup menguasai bahasa pemrograman Pascal, sebagian besar aturan penulisan Pseudopascal merupakan subset dari aturan Bahasa Pascal itu sendiri. Kecuali, ada beberapa hal yang di dalam Bahasa Pascal boleh dilakukan, dalam pseudopascal tidak ada atau tidak disarankan untuk dilakukan. Hal ini untuk memungkinkan kompatibilitas algoritma dengan bahasa lain seperti bahasa C sehingga peserta yang lebih menguasai bahasa selain Pascal dapat cukup mudah untuk mmahaminya juga. Jadi berkenaan dengan ini anda tinggal menemukan hal-hal yang menjadi "pantangan" tersebut.

Mengingat dokumen ini dituliskan lebih untuk acuan maka anda hanya akan menemukan penulisan substansi yang seringkas-ringkasnya tanpa disertai contoh-contoh yang memadai. Dalam kesempatan lain, semoga kami dapat menyediakan materi yang lebih sesuai untuk digunakan langsung dalam pembinaan-pembinaan peserta dalam pemrograman.

B. Terminologi dan Penjelasan Umum

Untuk memudahkan anda dalam pemaham dokumen ini beberapa terminologi terkait akan dijelaskan. Jika agak berbeda dari yang anda sudah ketahui kami harapkan itu tidak terlalu dipermasalahkan karena hal tersebut bukan tujuan utama dari dokumen ini.

Berikut ini mengenai algoritma

- **Algoritma** adalah urutan langkah-langkah sistematis yang terkait pada pemecahan suatu masalah; didalamnya bisa terdapat sejumlah **variabel**, **perintah, ekspresi** & **assignment**, **struktur kendali aliran** (control flow) dari algoritma, serta definisi **fungsi/prosedur**.

- **Pseudocode** adalah suatu cara penulisan algoritma agar ide dan logika dari algoritma dapat disampaikan/diekspersikan.
- **Pseudopascal** (alias **Pesudocode Pascal**) adalah pseudocode yang menggunakan (mengadopsi) beberapa notasi **Bahasa Pascal** berikut struktur penulisan programnya.

Berikut ini mengenai program komputer

- **Program komputer** atau kita singkat dengan kata **program** (istilah lainnya **code**) adalah susunan perintah-perintah dan operasi-operasi yang mengimplementasikan algoritma tertentu disertai yang ditulis dalam bahasa pemrograman tertentu.
- **Bahasa pemrograman** adalah bahasa yang di dalamnya terdapat aturan penulisan program.
- **Bahasa Pascal** adalah salah satu bahasa pemrograman, dan saat ini terdapat sejumlah versi dari bahasa Pascal diantaranya: Ansi Pascal, Turbo Pascal, Free Pascal, dlsb.

Algoritma yang ditulis dalam suatu pseudocode dibedakan dari programnya yang ditulis dalam suatu bahasa pemrograman akibat adanya perbedaan tujuan dari kedua hal itu. Algoritma dengan pseudocode bertujuan untuk menyampaikan ide dari algoritma bagi pembaca (dalam hal ini peserta seleksi), sementara program dalam suatu bahasa pemrograman untuk dapat dijalankan nantinya oleh komputer. Mengingat komputer "bodoh" maka dalam penulisannya suatu program harus 100% taat pada aturan-aturan penulisan programnya (istilahnya tidak ada kesalahan sintaks) sementara karena pembaca algoritma adalah manusia maka demi menyederhanakan dan memudahkan pemahaman maka aturan-aturan penulisannya digunakan secara fleksibel. Terkadang pesudocode dituliskan nyaris sama dengan versi programnya sendiri tetapi kadang-kadang diringkaskan menggunakan kalimat-kalimat bahasa manusia (dalam hal seleksi ini adalah bahasa Indonesia) bahkan beberapa bagian sengaja yang bukan fokusnya dihilangkan. Prinsip dalam penulisan pseudocode adalah "tuliskan seringkas-ringkasnya sejauh tidak mengurangi pengertian dari algoritma yang menjadi fokus pembahasan tersebut."

Dalam kaitannya dengan materi seleksi, pseudocode yang digunakan adalah pseudopascal berdasarkan kenyataan bahwa notasi-notasi Bahasa Pascal jauh lebih mudah dipahami daripada notasi bahasa pemrograman populer lainnya saat ini terutama bagi pemula (misalnya bahasa C). Selain itu setiap notasi yang digunakan dijaga untuk selalu berpadanan dengan notasi yang ada dalam bahasa lain tersebut sehingga peserta yang lebih menguasai bahasa selain Pascal tetap dapat memahami ide dari algoritma terkait.

Terakhir, untuk menghindari polemik akan cara-cara penulisannya, sekali lagi Pseuodopascal yang dimaksud dalam dokumen ini adalah Pseudopascal versi Olimpiade Sain Bidang Informatika/Komputer & TOKI.

C. Elemen-elemen Algoritma dengan Pseudopascal

Seperti pada terminologi di atas terdapat sejumlah elemen dasar dalam pseudopascal.

- Variabel
- Perintah
- Assignment & Ekspresi
- Struktur kendali aliran
- Fungsi/prosedur
- komentar

Sebelum penjelasan masing-masing komponen itu (di bagian C.1. s.d. C.6) berikut ini suatu contoh algoritma dalam pseudopascal. Algoritma ini tidak berarti apa-apa karena hanya ditulis untuk membatu di bagian penjelasan masing-masing elemen algoritma tsb kemudian.

```
   function kubik(a: integer): integer;
   begin
        kubik := a*a*a;
   end;
   procedure sw(var a: real; var b: real);
   begin
      menukarkan isi a dan b
   end;

var
   status: array[0..99] of boolean;   { array 1 dimensi }
   Tbl: array[0..99,0..1] of integer; { array 2 dimensi }
   sum: record x,y: real end;         { struktur komposit }
begin
    proses mengisikan data ke dalam matriks Tbl dan array status
    sum.x := 0;
    sum.y := 0;
    for I := 0 to 99 do begin
       t0 := (Tbl[I,0] + Tbl[I,1])/2 – I*2;
       status[I] := (Tbl[I,0] = Tb l[I,1]) or (sum.x <> sum.y);
       if (status[I] or status[99-I]) then begin
          sum.x := sum.x + kubik(Tbl[I,1] – t0);
          sum.y := sum.y + kubik(Tbl[I,0] – t0);
```

```
            sw(sum.x, sum.y);
        end;
    end;
    Proses pencetakan harga-harga sum.x dan sum.y
    ....
    ....
end.
```

C.1. Variabel

Variabel adalah elemen dari algoritma untuk menyimpan suatu harga tertentu pada suatu saat dan pada saat lain harga dalam variable itu bisa diubah ke harga lain sesuai kebutuhan.

Berikut ini aturan-aturan untuk variabel.

- Suatu variabel dituliskan dengan suatu nama secara unik dan nama variabel dapat terbentuk dengan karakter alfanumeris[1] (hanya huruf dan angka) kecuali karakter pertama adalah alfabet. Misalnya:
  - Variabel-variabel dalam contoh di atas bernama sum, Tbl, status, I dan t0.

- Huruf besar dan huruf kecil sama saja (*case insensitive*[2]) namun penulisan variabel diharapkan selalu konsisten dalam penggunaan huruf besar/kecilnya.

- Suatu variabel tidak diberikan nama dengan kata yang akan digunakan secara khusus dalam notasi penulisan algoritma (lihat daftar kata *reserverd word*) dengan tujuan untuk menghindari kerancuan arti. Misalnya:
  - katak-kata **function**, **begin**, **end**, **for**, **do**, **integer**, **real**, **var**, **array**, dan **boolean** di contoh di atas memiliki arti khusus (*reserved word*),

- Variabel bisa berbentuk tunggal, komposit, atau bisa juga berbentuk majemuk yang ditandai elemen-elemennya dengan indeks.

- Variabel berbentuk majemuk atau yang dikenal dengan nama *array*, harga indeks-indeks array berkisar dari 0 hingga bilangan positif tertentu[3] dan penulisan indeksnya setelah nama variabel tersebut di dalam sepasang kurung siku. Nomor indeks dari array dapat digantikan oleh harga suatu variabel lain dengan menuliskan nama variabel tersebut. Misalnya:

---

[1] Dalam bahasa Pascal selain alfanumeris, sejumlah karakter lain dapat pula digunakan. Disini kita batasi saja hanya alfanumeris.

[2] Dalam bahasa C nama variabel *case sensitive* sementara bahasa Pascal *case insensitive*

[3] Indeks dibatasi demikian untuk menjaga portabilitas (kemudahan untuk diimplementasikan) dengan bahasa C walaupun dalam bahasa Pascal indeks ini bisa menggunakan banyak cara.

- Status[I] menunjukan elemen array status ke I dan harga I sendiri dalam algoritma dapat berubah-ubah sehingga status[I] mengacu pada berbagai elemen.

- Array bisa berindeks lebih dari satu (disebut berdimensi lebih dari 1) misalnya array dua dimensi dan di dalam tanda kurung siku ada dua harga indeks yang dipisahkan oleh tanda koma[4] dengan alternatif sepasang kurung siku tutup & buka[5] ("]["). Misalnya:
  - status adalah array dengan indeks berkisar dari 0 s.d. 99.
  - Tbl adalah array dua dimensi (dengan 2 indeks) yang masing-masing indeks memiliki kisaran dari 0 s.d. 99 dan 0 s.d. 1. Penulisan elemen Tbl[i,0] dapat juga diganti dengan Tbl[i][0].

- Jika cukup jelas jenis dan penggunaanya maka suatu variabel tidak perlu dideklarasikan[6]. Jika dideklarasikan maka penulisan mengikuti aturan deklarasi variabel dalam bahasa Pascal yaitu dituliskan di awal program atau di awal fungsi/prosedur. Misalnya:
  - Tbl, status dan sum dideklarasikan tetapi I dan t0 tidak.

- Struktur komposit (variabel yang di dalamnya terdiri atas subvariabel[7]) sedapat mungkin akan dihindari. Seandainya diperlukan maka akan mengikuti aturan penulisan Record dalam bahasa Pascal dan penggunaan komponennya menggunakan aturan dalam bahasa Pascal yaitu nama variabel dan nama subvariabel dituliskan dengan dipisahkan tanda titik.
  - Misalnya variabel sum memiliki dua subvariabel yaitu x dan y. Masing-masing bisa dianggap sebagai variabel tersendiri dengan nama sum.x, dan sum.y.

C.2. Perintah

Perintah adalah satuan operasional dari suatu algoritma maupun algoritma.

- Perbedaannya, perintah-perintah dalam algoritma bisa dinyatakan dalam kalimat-kalimat bahasa sehari-hari sementara untuk program perintah-perintah harus bisa "dipahami" dan dijalankan oleh komputer sehingga karena keterbatasan komputer harus mengikuti aturan-aturan penulisan yang rinci. Misalnya:

[4] Mengikuti aturan dalam bahasa Pascal

[5] Mengikuti atuaran dalam bahasa C

[6] Deklarasi variabel dilakukan dalam sebagian besar bahasa pemrograman untuk menyatakan jenis kemungkinan harga yang dapat diisikan padanya, ukuran/dimensi (khusus untuk array), dan scope (bagian-bagian mana) dari program ia bisa digunakan.

[7] Record dalam Pascal atau struct dalam bahasa C.

- Algoritma contoh di atas berisikan perintah-perintah yang menggunakan kalimat bahasa Indonesia biasa dan perintah-perintah mengikuti aturan penulisan program bahasa Pascal.

- Perintah dalam bahasa sehari-hari dibedakan dalam font penulisannya dari perintah yang mirip perintah program. Misaalnya:
  - dalam algoritma contoh dituliskan dengan huruf miring sementara perintah dengan aturan bahasa pemrograman dengan huruf biasa dan diakhiri tanda ";" (sebagai kebiasaan saja, keduanya menggunakan huruf courier untuk dibedakan dari bagian teks yang lain).

- Kadang-kadang perintah dalam bahasa sehari-hari digunakan untuk mewakili sejumlah baris perintah dalam bahasa pemrograman sehingga diharapkan penulisan algoritma tidak terlalu bertele-tele namun tetap atau bahkan dapat lebih mudah untuk dihapami.
  - Misalnya: "Proses mengisikan data ke dalam …..(dst)" menggantikan sederetan operasi mengenai pengisian Tbl dan"Proses pencetakan …." yang menggantikan printah-perintah untuk mencetak harga variabel tsb.

- Sederetan perintah yang tidak menjadi fokus untuk ditampilkan dalam algoritma kadang-kadang diganti dengan sejumlah beris yang berisi titik-titik ("….."). Misalnya pada contoh di atas pada baris terakhir.

- Notasi-notasi penulisan perintah mengikuti notasi yang digunakan dalam bahasa Pascal yaitu ada yang bersifat **assignment** yang diikuti oleh suatu **ekspresi** atau hanya satu pemanggilan prosedur tertentu.

C.3. Assignment & ekspresi

Assignment adalah pemberian nilai pada suatu variable yang bisa berupa harga literal[8], harga dari variable lain, atau harga yang dihasilkan dari suatu ekspresi. Sementara ekspresi adalah operasi yang akan menghasilkan harga untuk diberikan pada variabel tersebut yang bisa merupakan ekspresi aritmatis maupun atau ekspresi lojik (Lihat **C.4. Ekspresi Aritmatis & logis** di bagian berikutnya). Aturan dalam penulisan assignment adalah sebagai berikut.

- Aturan penulisannya adalah: nama variabel penerima diikuti tanda assignment ":=" [9] selanjutnya nilai atau ekspresi yang menghasilkan nilai. Misalnya:
  - "sum.x:=0" artinya variabel sum.x diberi nilai literal 0
  - "t0:=(Tbl[I,0]+Tbl[I,1])/2–I*2;" artinya variabel t0 diberi harga hasil ekspresi yang aritmatis di ruas kanan setelah tanda assignment.

[8] Harga literal adalah harga-harga yang pasti misalnya integer 0, 100, -33; bilangan real 29.13, -21.41; string "Indonesia Raya", "Olimpiade sains".

[9] Dalam bahasa C assignment hanya dengan tanda "=" tetapi dalam pascal tanda "=" digunakan pada pembandingan kesamaan harga; dalam pseudopascal ini dibahas di bagian **C.4. Ekspresi**.

C.4. Ekspresi Aritmatis

Suatu ekspresi aritmatis berisikan sederetan operasi aritmatis dan jika yang jika dijalankan akan menghasilkan suatu bilangan (integer atau real). Suatu operasi aritmatis terjadi antara dua harga yang akan dioperasikan (disebut operand) dan sebuah operator kecuali untuk operasi unary yang hanya memerlukan satu operand.

- Notasi-notasi operator arimatis: perkalian (*), pembagian (/), penambahan (+), pengurangan (-), sisa pembagian (**mod**), pembagian dengan pembulatan ke bawah (**div**).
- Jika ekspresi berisi beberapa operasi maka operasi dengan orde tertinggi yang akan dilakukan terlebih dahulu.
- Urutan orde untuk operasi aritmatis mulai dari yang tertinggi hingga terendah adalah: perkalian/pembagian/modulo/divisi, kemudian penambahan/pengurangan (tanda / menyatakan berorde sama).
- Untuk yang berorde sama, urutan operasinya dilakukan dari kiri ke kanan dalam penulisan ekspresi namun jika berpotensi membingungkan kadang-kadang urutan bisa dipertegas dengan bantuan tanda kurung dan penggunaan tanda kurung memungkinkan bagian operasi yang ada di dalam tanda kurung dilakukan terlebih dahulu. Misalnya:
  - "(Tbl[I,0]+Tbl[I,1])/2–I*2" akan dilakukan penjumlahan Tbl[I,0] + Tbl[I,1] terlebih dahulu kemudian hasilnya akan dibagi 2 dan dikurangi oleh hasil perkalian antara I and 2.
- Operand-operand bisa variabel, literal, atau hasil dari pemanggilan fungsi. Misalnya:
  - dalam "sum.x+kubik(Tbl[I,1]–t0)" yang terlibat dalam operasi adalah sum.x, dan hasil dari pemanggilan fungsi kubik(Tbl[I,1]–t0) (Penjelasan pemanggilan fungsi ada di bagian fungsi).
- Operasi unary adalah operasi yang hanya berlaku sebagai operasi terkiri dalam ekspresi dengan operator minus (sama dengan pengurangan) dengan cara penulisan: operator mendahului operand. Efek dari operasi ini adalah memperhalikan harga operand di sampingnya itu dengan -1. Jika operand berharga positif akan menjadi negatif dan sebaliknya jika berharga negatif menjadi positif..
- Operasi tidak mengubah isi variabel-variabel yang menjadi operand tersebut kecuali jika ia adalah variabel yang akan menerima hasil operasi tersebut di dalam operasi assignment hasil dari ekspresi (lihat bagian **C.3. Assignment & Ekspresi** sebelumnya).

C.4. Ekspresi Lojik

Suatu ekspresi lojik bisa merupakan salah satu dari berikut ini: (1) satu harga literal lojik, (2) suatu variabel lojik, (3) suatu ekspresi pembandingan harga, (4) operasi lojik dari dua ekspresi lojik atau (5) negasi suatu ekspresi lojik. Keempat dan kelima memungkinkan ekspresi kombinasi yang kompleks.

- Ekspresi pembandingan terdiri atas dua antitas yang akan dibandingkan dan kriteria pembandingannya; penulisan berurutan entitas kiri kriteria pembandingan dan entitas kanan.
- Kriteria pembanding yang dikenali adalah "=" (sama[10]), "<" (lebih kecil), ">" (lebih besar), "<=" (lebih kecil atau sama dengan), ">=" (lebih besar atau sama dengan), "<>" (tidak sama[11]).
- Entitas yang akan dibandingkan bisa berupa variabel biasa atau ekspresi aritmatis (ekspresi aritmatis harus dituliskan di dalam tanda kurung!) atau suatu harga literal asalkan kedua entitas yang dibandingkan berjenis harga sama.
- Operasi lojik terjadi antara dua variabel lojik dan atau harga lojik (bisa dari hasil ekspresi pembandingan atau ekspresi lojik lain) dengan suatu operator kecuali jika operasi negasi dianggap sebagai operasi lojik maka ia hanya menggunakan satu operand.
- Operator-operator lojik yang memerlukan dua operand adalah: **and**, **or,** dan **xor** sementara operator negasi adalah **not.**
- Operator-operator tsb dijalankan dengan urutan sesuai dengan ordenya: dari yaang tertinggi ke yang terendah, jika sama dari kiri ke kanan dalam ekspresi.
- Urutan orde dari operator-operator tersebut (dari tertinggi ke terendah): not, and, or/xor (tanda / menyatakan berorde sama).
- Seperti halnya dalam ekspresi aritmatis, tanda kurung dapat digunakan untuk mengatur urutan operasinya. Untuk menambah kejelasan algoritma kadang-kadang tanda kurung disertakan.
- Hasil ekspresi bisa diassign ke suatu variabel boolean atau bisa juga digunakan dalam perintah kendali aliran (Dibahas di **C.5. Struktur Kendali Aliran**).
    - Misalnya: dalam "status[I] := (Tbl[I,0] = Tb l[I,1]) or (sum.x <> sum.y)" di ruas kanan terdapat operasi lojik "or" antara dua hasil pembandingan. Pembadingan pertama adalah memeriksa kesamaan dan pembandingan kedua adalah memeriksa ketidaksamaan. Hasil ekspresi ini diberikan ke dalam variabel status[I] (pada posisi berindeks I dari array status).

C.5. Struktur kendali aliran

Struktur kendali aliran adalah suatu bentuk/struktur yang memiliki peranan khusus untuk mengatur aliran urutan pengerjaan operasi atau beberapa operasi tertentu. Terdapat sejumlah bentuk kendali aliran (atau dikenal juga dengan istilah struktur) sebagai berikut (penjelasannya masing-masing tedapat pada bagian D.1. s.d. D.7):

- **begin-end**
    - struktur untuk menjadikan sejumlah perintah atau elemen lain sebagai satu kesatuan
- **if-then**

[10] Dalam bahasa C tanda pembandingan kesamaan ini dituliskan ganda ("==") untuk membedakannya dengan assignment.

[11] Dalam bahasa C tanda pembandingan ketidaksamaan ini dituliskan dengan "!=".

- struktur untuk menjalankan perintah (atau perintah-perintah jika disatukan dalam struktur **begin-end**) setelah notasi "**then**" jika ekspresi yang diperiksa (antara notasi "**if**" dan notasi "**then**") berharga boolean "true".

- **if-then-else**
  - struktur untuk menjalankan perintah (atau perintah-perintah jika disatukan dalam struktur **begin-end**) setelah notasi "**then**" jika ekspresi yang diperiksa (antara notasi "**if**" dan notasi "then") berharga boolean "**true**" dan sebaliknya jika "**false**" menjalankan perintah (atau perintah-perintah jika disatukan dalam struktur **begin-end**) setelah notasi "**false**".
- **for-do**
  - pengulangan perintah (atau perintah-perintah jika disatukan dalam struktur begin-end) setelah notasi "**do**" dengan jumlah pengulangan sejalan dengan perubahan harga iterator yang dinyatakan di antara notasi "**for**" dan notasi "**do**".
- **while-do**
  - pengulangan (iterasi) perintah (atau perintah-perintah jika disatukan dalam struktur **begin-end**) setelah notasi "**do**" selama ekspresi yang diperiksa (antara notasi "**while**" dan notasi "**do**") berharga boolean "true". Jika pertama kali diperiksa langsung berharga "false" maka iterasi sama sekali tidak dilakukan
- **repeat-until**
  - pengulangan (interasi) perintah atau perintah-perintah antara notasi "repeat" dan notasi "until" hingga ekspresi yang diperiksa (setelah notasi "until") berharga boolean "true". Jadi minimal satu kali dijalankan.
- **case-of-end**
  - jika if-then-else adalah pencabangan aliran dengan dua cabang maka struktur ini bisa memiliki pencabangan yang sangat banyak.

- Bentuk-bentuk kendali tersebut bisa bersarang yaitu suatu bentuk di dalam bentuk yang lain.
  - Misalnya: di dalam **for** I := 0 to 99 **do** ada **begin-end**; di dalam begin-end tsb ada **if-then** dan kemudian di dalamnya ada lagi **begin-end** dst.
- Untuk if-then, if-then-else, while-do, repeat-until, kriteria pengendalian aliran adalah suatu ekspresi.
  - Misalnya dalam "if (status[I] or status[99-I]) then ..." sebagai ekspresi yang diperiksa adalah hasil operasi lojik antara status[I] dengan status[99-I] dengan operator lojik "or".[12]

[12] Dalam Pascal ekspresi tidak harus dibatasi tanda kurung sementara dalam C harus. Dalam contoh ini ekspresi tsb dibatasi dengan tanda kurung. Walaupun dalam Pseudopascal tidak diharuskan, untuk situasi tertentu disarankan adanya tanda kurun tsb untuk membantu memperjelas penulisan algoritma.

C.6. Algoritma Utama/Fungsi/prosedur

- Algoritma utama adalah badan utama dari algoritma sebagai analogi program utama dalam program. Namun berbeda dengan program, algoritma bisa jadi hanya merupakan bagian tertentu dari suatu program bahkan bagian dari suatu fungsi/prosedur dari program tsb..
- Fungsi/prosedur adalah sederetan perintah/operasi seperti halnya algoritma itu sendiri yang dipisahkan dari algoritma utamanya guna menjadikannya sebagai subfokus dalam algoritma. Dalam suatu algoritma jika apa yang dikerjakan dalam fungsi/prosedur itu dapat dianggap jelas maka fungsi/prosedur tersebut tidak dituliskan atau digantikan dengan kalimat bahasa sehari-hari..
- Pascal menyediakan sejumlah fungsi/prosedur yang siap pakai (tanpa pemrogram menuliskan lagi fungsi/prosedur itu), dalam Pseudopascal ini beberapa fungsi/prosedur Pascal tertentu yang sangat umum juga digunakan.
    - Misalnya read (untuk membaca masukan), readln (membaca masukan termasuk karakter "end of line"), write (untuk mencetak keluaran), writeln (mencetak keluaran dan suatu "end of line"), eof (untuk memeriksan apakah masukan sudah mencapai "end of file").
- Mengikuti definisi Pascal, fungsi dibedakan dari prosedur dari adanya harga yang dihasilkan oleh fungsi sebagai akibat dari pemanggilan/penggunaan fungsi tersebut.
- Suatu fungsi/prosedur bisa didefinisikan dengan sejumlah argumen (istilah lain: parameter). Pada saat pemanggilan fungsi/prosedur yang bersangkutan, argumen ini jika ada dituliskan di dalam tanda kurung setelah nama fungsi/prosedur itu.
- Ada dua jenis argumen: argumen bersifat "call by value" dan argumen "call by reference". Untuk "call by value" dalam pemanggilannya yang dikirimkan ke fungsi/prosedur adalah harganya sementara "call by reference" yang dikirimkan adalah variabel itu sendiri. Perbedaan kedua argumen ini dinyatakan di daftar argumen di bagian definisi fungsi dengan mengawali nama variabel ybs dengan notasi "var".
    - Misalnya: fungsi kubik di atas memerlukan satu argumen 'call by value" dan prosedur sw memerlukan dua argumen "call by reference".
- Untuk "call by value" selain suatu harga literal, harga suatu variabel, bisa juga hasil dari suatu operasi aritmatis atau ekspresi lojik.
    - Misalnya "kubik(Tbl[I,1] – t0)" memanggil fungsi kubik dengan argumen adalah harga yang dihasilkan dari operasi "Tbl[I,1] – t0".
- Khususnya pada pemanggilan fungsi yang akan menghasilkan harga setelah pemanggilan itu dilaksanakan, harga hasilnya itu kemudian menggantikannya di dalam operasi/ekspresi yang bersangkutan.
    - Misalnya: dalam "sum.x := sum.x + kubik(Tbl[I,1] – t0)" setelah pemanggilan "kubik(Tbl[I,1] – t0)", harga yang dihasilkannya beserta harga dalam sum.x dijumlahkan dan hasilnya diassign kembali ke sum.x

- Untuk “call by reference” argumen di bagian yang melakukan pemanggilan hanya dapat berupa variabel saja dan ke variabel itu di bagian fungsi/prosedur jika mengalami perubahan isi (walau dengan nama variabel yang lain), saat kembali ke pemanggil perubahan itu tetap berlaku.
    - Misalnya: dalam pemanggilan “sw(sum.x, sum.y)” variabel sw.x dan sw.y saat berada di prosedur sw dengan nama baru a dan b akan dieprtukarkan, sehingga saat kembali isi sw.x dan sw.y sudah bertukaran.

D. Aturan-aturan Penulisan Struktur Kendali Aliran

Secara umum aturan-aturan penulisan struktur kendali aliran ini mengikuti aturan bahasa Pascal. Seperti hanya Pascal (dan bahasa-bahasa lain, penulisan notas-notasi, perintah-perintah atau ekspresi-ekspresi tidak harus dibaris tersendiri/tertentu; bisa saja beberapa elemen imi dituliskan bergabung dengan baris yang sama. Namun, demi kejelasan dalam pembacaan algoritma beberapa kebiasaan dilakukan (tapi tidak harus).

- Perintah yang ditulis dalam bahasa sehari-hari dituliskan tidak dicampur dalam satu baris dengan elemen algoritma yang lain.
- Indentasi yang sama untuk blok struktur yang sama sangat disarankan..

D.1 Struktur begin-end

- Perintah-perintah dalam satu deret yang menjadi satu kesatuan struktural (satu blok) dituliskan di antara notasi **begin** dan notasi **end**. Perintah yang ditulis dalam notasi bahasa Pascal dituliskan dengan tanda separator “**;**” di akhir setiap perintah sementara perintah dalam bahasa sehari-hari dituliskan tanpa tanda separator tsb.
- Jika terdapat struktur lain di antara satu **begin**-**end**, maka struktur itu diperlakukan sebagaimana perintah yang ditulis dalam bahasa pascal (diakhiri separator “**;**”).

D.2 Strktur if-then

- Kondisi yang akan diperiksa dalam struktur ini adalah ekspresi lojik yang diletakan di antara notasi **if** dan notasi **then**. Perintah (atau perintah-perintah) yang akan dilakukan jika ekspresi lojik itu berharga boolean **true** diletakkan setelah notasi **then**.
- Jika lebih dari satu perintah maka perintah-perintah itu dijadikan satu kesatuan struktur dengan tambahan notasi **begin**-**end** yang mengapitnya.

D.3 if-then-else

- Seperti pada **if-then** Kondisi yang akan diperiksa dalam struktur ini adalah ekspresi lojik yang diletakan di antara notasi **if** dan notasi **then**. Perintah (atau perintah-perintah) yang akan dilakukan jika ekspresi lojik itu berharga boolean **true** diletakkan setelah notasi **then** dan perintah (atau perintah-perintah) yang akan dijalankan jika berharga **false** diletakkan setelah notasi **else**.
- Baik untuk **then** maupun untuk **else**, jika lebih dari satu perintah maka masing-masing perintah-perintah itu dijadikan kesatuan-kesatuan struktur dengan tambahan notasi **begin**-**end** yang mengapitnya (masing-masing).

D.4 for-do

- Di antara notasi **for** dan notasi **do** ekspresi iteratif (ekspresi yang menyatakan bagaimana iterasi itu harus terjadi) dituliskan.
  - Penulisan ekspresi iteratif berisi
  - nama variabel iterator (variabel integer yang harganya akan berubah-ubah sejalan dengan iterasi yang terjadi),
  - setelah tanda assignment dituliskan harga awal iterator (yang bisa merupakan suatu harga literal atau ekspresi aritmatis),
  - notasi **to** atau **downto** yang menyatakan arah harga iterator menaik atau menurun, harga terakhir interasi itu masih akan dilakukan (yang bisa merupakan suatu harga literal atau suatu ekspresi aritmatis), dan
  - jika kenaikan/penurunan bukan bilangan 1, harga perubahan yang terjadi pada iterator setiap satu iterasi dilalui dinyatakan dengan notasi **step** dan diikuti harga perubahannya (misalnya "step 2").
- Setelah notasi **do** perintah (atau perintah-perintah) yang akan dilakukan pada setiap kali iterasi dituliskan. Jika lebih dari satu perintah maka perintah-perintah itu dijadikan satu kesatuan struktur dengan tambahan notasi **begin-end** yang mengapitnya.

D.5 while-do

- Kondisi yang akan diperiksa adalah ekspresi lojik yang diltakan di antara notasi **while** dan notasi **do**. Perintah (atau perintah-perintah) yang akan dilakukan secara berulang-ulang selama ekspresi lojik itu berharga boolean **true** diletakkan setelah notasi **do**.
- Jika lebih dari satu perintah maka perintah-perintah itu dijadikan satu kesatuan struktur dengan tambahan notasi **begin-end** yang mengapitnya.

D.6 repeat-until

- Kondisi yang akan diperiksa adalah ekspresi lojik yang diltakan setelah notasi until. Perintah (atau perintah-perintah) yang akan dilakukan dan kemudian diulang-ulang selama ekspresi lojik itu berharga boolean **false** diletakkan antara notasi **repeat** dan notasi **until**.

D.7 case-of-end

- Hal yang akan diperiksa adalah variabel atau ekspresi arinmatis yang memiliki harga-harga tertentu. Variabel atau ekspresi aritmatis itu diletakkan di antara notasi **case** dan notasi **of**.
- Di antara notasi of dan notasi end terdaftar sejumlah kemungkinan harga/kasus dan perintah (atau perintah-perintah) yang akan dijalankan untuk kemungkinan harga tersebut.
- Satu harga literal[13] yang mungkin[14] diletakkan di sebelah kiri, diikuti notasi "**:**" (titik dua) lalu diikuti oleh perintah yang akan dijalankan selanjutnya sebuah "**;**" (titik koma)[15].
- Jika ada beberapa perintah yang akan dijalankan maka perintah-perintah itu dijadikan satu kesatuan struktur dengan tambahan notasi **begin-end** yang mengapitnya.

E. Aturan-aturan Penulisan Prosedur dan Fungsi

Secara umum aturan kerangka penulisannya mengikuti aturan dalam bahasa Pascal kecuali beberapa hal termasuk bahwa badan algoritma di dalam fungsi atau prosedur sesuai dengan pembahasan algoritma di atas (memungkinkan baris perintah dalam bahasa sehari-hari, perintah-perintah rinci yang bukan fokus diperlihatkan sebagai baris "…."). Seperti halnya nama variabel nama-nama prosedur maupun function bersifat *case insentitive* namun demi untuk memberikan kejelasan dalam pembacaan, penamaan prosedur/function maupun variabel direkomendasikan untuk konsisten di setiap bagian algoritma.

Dalam bahasa Pascal, suatu prosedur/function dapat memiliki prosedur/function lokalnya sendiri. Dalam pseudocode ini demi kompatibilitas dengan bahasa lain, hal itu ditiadakan.

E.1 Prosedur

Mengawali suatu prosedur notasi **procedure** dituliskan dan diikuti nama prosedur. Jika ada argumen, mana argumen-argumen didaftarkan di dalam **tanda kurung** dan setiap argumen tersebut dituliskan dengan dipisahkan dengan tanda koma ("**,**"). Dalam bahasa Pascal jenis variabel harus dispesifikasikan, namun dalam Pseudopascal ini seandainya cukup jelas maka demi mempersingkat, jenis variabel bisa dihilangkan. Untuk menunjukkan suatu argumen bersifat pass-by-reference maka di depan nama argumen itu dituliskan notasi **var** (seperti pada bahasa Pascal).

---

[13] Dalam Bahasa C, sebelum harga literal tsb, ada notasi "case" mendahuluinya.

[14] Dalam Pascal harga yang mungkin ini bisa dinyatakan sebagai range harga yang mungkin misalnya bilangan-bilangan antara 1 sampai dengan 10 dengan menulisnya sebagai "1..10" atau beberapa kemungkinan harga dengan dipisahkan koma misalnya "2, 5, 9". Hal ini tidak ada di bahasa C sehingga dalam Pseudopascal ini hal itu dihindari.

[15] Dalam bahasa C, harus diikuti notasi "break;" karena tanpa itu maka alirannya jadi sangat berbeda.

Berikutnya jika ada variabel lokal yang perlu ditonjolkan didaftarkan (beserta jenis variabel tersebut) setelah notasi **var**. Badan algoritma prosedur dituliskan di antara **begin** dan **end** dan diikuti tanda "**;**" (titik koma).

E.2 Fungsi

Mirip seperti pada prosedur kecuali tiga hal berikut ini. Pertama, yang mengawali fungsi adalah notasi **function**. Kedua, setelah nama (serta argumen-argumennya yang dituliskan di dalam tanda kurung), suatu notasi ":" (titik dua) dan jenis harga yang akan dihasilkan fungsi dituliskan. Ketiga dalam badan algoritma nama fungsi digunakan seolah suatu variabel penerima assignment yang apabila algoritma dalam fungsi selesai dijalankan maka harga terakhir yang diberikan padanya adalah harga yang akan dikembalikan ke algoritma pemanggil fungsi ini. Jika nama fungsi berada sebagai operand dari suatu operasi maka yang terjadi adalah pemanggilan fungsi terhadap disinya sendiri yang dikenal dengan istilah rekursif.