

PROSEDUR

Definisi :

Prosedur adalah sederetan instruksi algoritmik yang diberi nama, dan akan menghasilkan efek neto yang terdefinisi. Prosedur menyatakan suatu aksi dalam konsep algoritma yang dibicarakan pada cerita “Mengupas kentang”.

Mendefinisikan (membuat spesifikasi) prosedur berarti menentukan nama prosedur serta parameternya (jika ada), dan mendefinisikan **keadaan awal (Initial State, I.S.)** dan **keadaan akhir (Final State, F.S.)** dari prosedur tersebut. Prosedur didefinisikan (dituliskan spesifikasinya) dalam **kamus**. Cara penulisan spesifikasi : prosedur diberi **nama**, dan **parameter formal** (jika ada) yang juga diberi nama dan dijelaskan typenya.

Secara sederhana, dapat diartikan bahwa sebuah prosedur yang terdefinisi “disimpan” di tempat lain, dan ketika “dipanggil” dengan menyebutkan namanya “seakan-akan” teks yang tersimpan di tempat lain itu menggantikan teks pemanggilan. Pada saat itu terjadi asosiasi parameter (jika ada). Dengan konsep ini, maka I.S dan F.S dari prosedurlah yang menjamin bahwa eksekusi program akan menghasilkan efek neto yang diharapkan.

Jadi, setiap prosedur harus :

- Didefinisikan (dibuat spesifikasinya) dan dituliskan kode programnya
- Dipanggil, pada saat eksekusi

Parameter Prosedur

Prosedur tanpa parameter memanfaatkan nilai dari nama-nama yang terdefinisi pada kamus global. Pemakaiannya biasanya harus “hati-hati”, apalagi jika teks program sudah sangat besar dan implementasinya menjadi banyak file.

Prosedur berparameter dirancang, agar sepotong kode yang sama ketika eksekusi dilakukan, dapat dipakai untuk nama parameter yang berbeda-beda.

Nama parameter yang dituliskan pada definisi/spesifikasi prosedur disebut sebagai parameter formal. Sedangkan parameter yang dituliskan pada pemanggilan prosedur disebut sebagai parameter aktual.

Parameter formal adalah nama-nama variabel (list nama) yang dipakai dalam mendefinisikan prosedur, dan membuat prosedur tersebut dapat dieksekusi dengan nama-nama yang berbeda ketika dipanggil. Parameter formal adalah list nama yang akan dipakai pada prosedur, yang nantinya akan diasosiasikan terhadap nama variabel lain pada saat pemanggilan. Sesuai dengan ketentuan nilainya, ada tiga type parameter formal:

- parameter Input, yaitu parameter yang diperlukan prosedur sebagai masukan untuk melakukan aksi yang efektif.
- parameter Output, yaitu parameter yang nilainya **akan** dihasilkan oleh prosedur. Hasil nilai akan disimpan pada nama parameter Output ini.
- parameter Input/Output, yaitu parameter yang nilainya diperlukan prosedur sebagai masukan untuk melakukan aksi, dan pada akhir prosedur akan dihasilkan nilai yang baru.

Pemanggilan Prosedur

Memakai atau "memanggil" prosedur adalah menuliskan nama prosedur yang pernah didefinisikan, dan memberikan harga-harga yang dibutuhkan oleh prosedur itu untuk dapat melaksanakan suatu aksi terdefinisi. Sebuah prosedur juga boleh "memakai" atau memanggil prosedur.

Parameter aktual adalah nama-nama informasi yang dipakai ketika prosedur itu dipakai ("dipanggil"). Parameter aktual dapat berupa nama atau harga, tetapi harus berupa nama jika parameter tersebut adalah parameter Output (karena hasilnya akan disimpan dalam nama tersebut). Sesuai dengan jenis parameter formal, parameter aktual pada saat pemanggilan :

- parameter Input harus terdefinisi nilainya (karena dibutuhkan oleh prosedur untuk menghasilkan nilai)
- parameter Output tidak perlu terdefinisi nilainya, tetapi justru setelah pemanggilan prosedur akan dimanfaatkan oleh deretan instruksi berikutnya, karena nilainya akan dihasilkan oleh prosedur.
- parameter Input/Output harus terdefinisi nilainya dan nilai baru yang diperoleh karena eksekusi prosedur akan dimanfaatkan oleh deretan instruksi berikutnya.

Pada saat eksekusi, terjadi asosiasi nama parameter formal dengan nama parameter aktual. Pada notasi algoritmik, asosiasi dilakukan dengan cara "by position". urutan nama pada parameter aktual akan diasosiasikan sesuai dengan urutan parameter formal. Karena itu, type harus kompatibel.

Beberapa bahasa pemrograman, dapat dilakukan asosiasi dengan nama dan memperbolehkan adanya *nilai default*. Lihat Catatan Singkat Bahasa Ada [Liem-99c]. Beberapa bahasa pemrograman (Ada, C) juga memperbolehkan nama prosedur yang sama tetapi parameternya berbeda (*overloading subprogram*).

Prosedur dapat mempunyai **kamus lokal**, yaitu pendefinisian nama yang dipakai dan hanya berlaku dalam ruang lingkup prosedur tersebut. Jika nama yang dipakai di dalam prosedur tidak terdefinisi dalam list parameter formal atau dalam kamus lokal, maka nama tersebut harus sudah terdefinisi pada prosedur yang memakainya. Penulisan kamus lokal sama dengan kamus global, yang berbeda adalah lingkup berlakunya nama yang didefinisikan:

- pada kamus "global", nama berlaku untuk program dan semua prosedur/fungsi yang didefinisikan.
- pada kamus lokal, nama berlaku untuk prosedur/fungsi yang bersangkutan dan prosedur / fungsi yang didefinisikan di dalamnya.

Program yang modular adalah program yang dibagi-bagi menjadi modul-modul yang terdefinisi dengan baik dalam bentuk prosedur-prosedur. Setiap prosedur harus jelas definisi dan ruang lingkungnya, supaya dapat dipanggil secara independent. Pembagian program besar dalam prosedur-prosedur akan mempermudah pembagian kerja di antara beberapa pemrogram. Penulisan prosedur juga akan memudahkan program untuk dibaca oleh "manusia" karena kita tidak perlu terpaku pada detil kode prosedur untuk mengerti efek neto yang dihasilkannya. Bahkan dalam beberapa hal, pemrogram tidak perlu tahu sama sekali "isi" atau kode dari prosedur dengan mengetahui spesifikasinya, beberapa bahasa pemrograman bahkan menyediakan prosedur terdefinisi yang sering dipakai dalam pemrogram sehingga pemrogram tidak perlu lagi menuliskan kodenya.

Prosedur berlaku untuk ruang lingkup (*universe*) tertentu, terutama untuk prosedur yang tidak mempunyai parameter. Dalam dua bab berikut, yaitu Mesin Gambar dan Mesin Karakter, akan diberikan gambaran lebih jelas dan lengkap tentang pendefinisian dan pemakaian prosedur karena keduanya adalah mesin abstrak yang tertentu.

Notasi algoritmik untuk prosedur

1. Pendefinisian/Spesifikasi prosedur

procedure NAMAPROSEDUR ([<i>list nama parameter formal: type</i>]) {Spesifikasi , Initial State, Final State}
Kamus lokal : { semua NAMA yang dipakai dalam BADAN PROSEDUR }
Algoritma : { BADAN PROSEDUR } {deretan instruksi pemberian harga, input, output, analisa kasus, pengulangan atau prosedur}

dengan syarat :

- nama prosedur dan prameternya harus disebutkan dalam kamus pemanggil.
- list parameter formal boleh tidak ada (kosong), dalam hal ini di dalam prosedur akan dipakai nama lokal dan nama-nama yang telah terdefinisi dalam kamus "pemakai"nya.
- jika list parameter formal ada (tidak kosong, minimal satu nama), maka harus berupa satu atau beberapa nama INFORMASI beserta typenya.

2. Pemanggilan prosedur

Program POKOKPERSOALAN {Spesifikasi , Input, Proses, Output}
Kamus : { semua NAMA yang dipakai dalam algoritma } procedure NAMAPROSEDUR (Input/Output : < <i>list-nama parameter formal</i> >) {Spesifikasi : Initial State, Final State}
Algoritma : { deretan instruksi pemberian harga, input, output, analisa kasus, pengulangan} NAMAPROSEDUR (< <i>list parameter aktual</i> >)

dengan syarat :

- pada waktu pemanggilan terjadilah asosiasi antara parameter formal dengan parameter aktual sesuai dengan urutan penulisan dalam list-nama parameter formal.
- list parameter aktual harus sama jumlah, urutan dan typenya dengan list parameter formal.
- list parameter aktual yang berupa **Input** dapat berupa **nama informasi** atau nama **konstanta** yang telah terdefinisi dalam kamus atau konstanta; dapat

juga berupa harga konstanta, atau harga yang dihasilkan oleh suatu ekspresi atau fungsi.

- list parameter aktual yang berupa **Output** harus berupa **nama informasi**. Jika didefinisikan sebagai Input, walaupun pernah diubah dalam badan prosedur, isi dari nama yang dipakai pada parameter aktual tidak pernah berubah. Jika didefinisikan sebagai parameter Output dan parameter aktual yang diasosiasikan terhadapnya pernah diubah harganya dalam badan prosedur, isinya akan berubah.

Contoh 1-Prosedur: VOLTAGE

Tuliskanlah program yang membaca tahanan (Ohm) dan arus (Ampere), kemudian menghitung tegangan yang dihasilkan dan menuliskan hasilnya. Perhitungan tegangan harus dituliskan menjadi suatu prosedur bernama PROSES, supaya struktur program jelas : Input - Proses - Output.

Input : R : integer, tahanan (Ohm) dan A : integer, arus (Ampere)

Proses : menghitung $V = R * A$

Output : V : integer, tegangan (Volt)

Pelajarilah dua buah solusi yang diberikan berikut ini, dan berikan komentar anda.

Solusi 1 : Prosedur tanpa parameter

Program VOLTAGE1	
{Program yang membaca tahanan dan arus, menghitung Voltage dan mencetak hasil perhitungan}	
Kamus : R : <u>integer</u> { tahanan dalam ohm} A : <u>integer</u> { arus dalam ohm} V : <u>integer</u> { tegangan dalam ohm} procedure PROSES1 { Prosedur untuk "memproses" tahanan dan arus menjadi tegangan }	
Algoritma : <u>input</u> (R,A) PROSES1 <u>output</u> (V)	

procedure PROSES1	0.1
{ I.S : diberikan harga R dan A yang telah terdefinisi }	
{ FS : memproses R dan A sehingga dihasilkan V yaitu tegangan dengan rumus : $V = R * A$ }	
Kamus lokal :	
Algoritma : $V \leftarrow R * A$	

Solusi 2 : Prosedur dengan parameter

Program VOLTAGE2 {Program yang membaca tahanan dan arus, menghitung Voltage dan mencetak hasil perhitungan}	0.0
Kamus : R : <u>integer</u> { tahanan dalam ohm} A : <u>integer</u> { arus dalam ohm} V : <u>integer</u> { ttegangan dalam ohm} procedure PROSES2 (<u>Input</u> : R,A : <u>integer</u> ; <u>Output</u> V: <u>integer</u>) { Prosedur untuk "memproses" tahanan R dan arus A menjadi tegangan V}	
Algoritma : <u>input</u> (R,A) PROSES2 (R,A,V) <u>output</u> (V)	

procedure PROSES2 (<u>Input</u> : R,A : <u>integer</u> ; <u>Output</u> V: <u>integer</u>) { I.S : diberikan harga R dan A yang telah terdefinisi} { FS : memproses R dan A sehingga dihasilkan V yaitu tegangan dengan rumus : $V = R * A$ }	0.1
Kamus lokal :	
Algoritma : $V \leftarrow R * A$	

Catatan :

1. Prosedur dengan parameter lebih menjamin modularitas program. Sedaapat mungkin semua prosedur diparametrisasi dengan baik.
2. Prosedur tanpa parameter bekerja dengan nama global. Hanya boleh dipakai untuk kasus yang sangat khusus, yaitu jika nama global merupakan “*universe*” (dunia, lingkungan) dari program, misalnya pada contoh mesin abstrak.
3. Prosedur tanpa parameter tidak boleh dipakai jika alasannya hanya karena pemrogram malas menuliskan parameter !

Contoh 2-Prosedur: TUKAR

Prosedur untuk menukar dua harga yang disimpan dalam dua nama a dan b

I.S. : diberikan a=A dan b=B

F.S. : a=B dan b=A

Program TUKAR {Program yang membaca dua buah harga x dan y, menyimpannya dan kemudian menukarnya}
Kamus : x,y : <u>integer</u> procedure PROCTUKAR (<u>Input/Output</u> : a,b : <u>integer</u>) { Prosedur untuk menukar dua buah harga yang tersimpan dalam dua nama integer} { I.S : diberikan a=A dan b=B, } { F.S : a=B dan b=A}
Algoritma : <u>input</u> (x.,y) PROCTUKAR (x,y) <u>output</u> (x,y)

procedure PROCTUKAR (<u>Input/Output</u> : a,b : integer)	0.1
{ I.S : diberikan a=A dan b=B, }	
{ FS : a=B dan b=A }	
Kamus lokal :	
Temp : integer	
Algoritma :	
Temp ← a { Temp = a; a = a; b = b }	
a ← b { Temp = a; a = b; b = b }	
b ← Temp { Temp = a; a =b; b = a }	

Contoh3-Prosedur : PUTAR3BIL

Contoh pemanfaatn prosedur yang pernah dipakai

Gunakan prosedur TUKAR untuk menulis prosedur yang "memutar" 3 buah nama integer

Contoh : jika a berisi 1, b berisi 2 dan c berisi 3, maka hasilnya :
a berisi 3, b berisi 1 dan c berisi 2.

procedure PUTAR3BIL (<u>Input/Output</u> a,b,c : integer)
{ I.S : a=A dan b=B, dan c=C }
{ F.S : a=C, b=A, c=B }
Kamus lokal :
Algoritma :
<div style="text-align: right;">{ a= A, b=B, c=C }</div> <div>PROCTUKAR (a, c) { a = C; b= B, c=A }</div> <div>PROCTUKAR (b, c) { a = C; b= A, c=B }</div>

Contoh di atas adalah contoh sebuah prosedur yang memakai sebuah prosedur lain.

Catatan:

Contoh-contoh di atas tidak terlalu mewakili daya guna penulisan prosedur dengan nyata, pada bagian-bagian berikutnya akan terlihat lebih nyata manfaat penulisan prosedur, yang membuat :

- program mudah dibaca
- prosedur yang sama dipakai berkali-kali sehingga mengurangi penulisan yang berulang
- pengkodean yang independen

Latihan soal:

1. Kerjakanlah kembali soal-soal pada bagian sebelumnya, dengan mendefinisikan prosedur, jika memang sesuai.
2. Apa beda prosedur dan fungsi ?
3. Pada bahasa C, hanya ada "fungsi". Prosedur direalisasi dengan menuliskannya sebagai fungsi yang tidak menghasilkan harga. Apa pendapat Anda ?