ALGORITMA PEMROGRAMAN

Bambang Wahyudi, SKom., MMSI

PROGRAM KOMPUTER

Program komputer adalah rangkaian kata perintah yang telah dimengerti oleh komputer untuk dikerjakannya. Kata-kata perintah tersebut membentuk suatu bahasa yang disebut dengan bahasa pemrograman. Sebagaimana bahasa pada manusia, bahasa pemrograman juga terdiri atas banyak macam bahasa, dan memiliki aturannya masing-masing.

Sulitnya, komputer saat ini belum diberi hak inisiatif, sehingga jika ada sedikit saja kesalahan penulisan perintah oleh pemrogram, ia tidak mau memaklumi-nya atau berusaha memperbaiki sendiri kesalahan tersebut. Serta merta ia "ngambek" dan tidak mau mengerjakan perintah-perintah lainnya. Komputer diciptakan melalui logika manusia, karenanya, ia bekerja secara logis, tanpa campur-tangan "perasaan."

ALGORITMA PEMROGRAMAN

Orang yang telah terbiasa "bergaul" dengan komputer menggunakan satu bahasa pemrograman tertentu (tingkat mahir), biasanya tidak lagi memerlukan kertas coret-coretan untuk membuat suatu program komputer. Namun bagi pemula, pembelajar, atau yang belum mahir, diperlukan kertas coret-coretan tersebut.

Kertas coret-coretan itu akan digunakan untuk menyusun algoritma (langkah-langkah penyelesaian masalah), *flowcharting* (alur logika perintah, yang merupakan aplikasi dari algoritma), maupun menuliskan perintah sesuai dengan kaidah dari bahasa pemrograman yang akan digunakannya.

Sewaktu menyusun algoritma, kita tidak perlu tahu (atau tidak perlu menyesuaikan dengan) bahasa pemrograman yang nanti akan kita gunakan. Hal utama yang kita pikirkan adalah kaidah (hirarki) dari komputer itu sendiri, yaitu input-proses-output.

Input adalah data yang harus ada (sudah ada/ sudah tersedia), yang dapat diproses dengan aturan-aturan tertentu untuk menghasilkan output seperti yang dikehendaki. Data yang ada harus logis (masuk akal) bahwa "ia" dapat diproses untuk menghasilkan output.

PERLUNYA PERINTAH BAHASA PEMROGRAMAN DI DALAM ALGORITMA

Meskipun sudah dikatakan, bahwa sewaktu kita menyusun algoritma kita tidak perlu tahu bahasa pemrograman apa yang akan digunakan kelak, namun, untuk penulisan algoritma yang lebih efisien dan efektif, maka penggunaan sebagian perintah yang ada di dalam bahasa pemrograman perlu dilakukan juga.

Adapun perintah bahasa pemrograman yang paling sering digunakan untuk menyusun algoritma adalah bahasa pemrogrman yang terstrukutur, seperti Pascal, C, SNOBOL, PL/1, dan sebagainya.

Misalkan saja, untuk contoh berikut ini :

Langkah 1 : Beri nilai 10 ke variabel S

Maka, akan lebih mudah jika ditulis sebagai :

Langkah 1 : S := 10;

Belum lagi jika algoritma yang ditulis harus melakukan perulangan langkah ke langkah-langkah sebelumnya (*looping*).

10 Mulai I:= 1;
11 Lakukan perbandingan data ke I dengan data ke I+1
12 Jika data ke I+1 lebih kecil, maka tukar tempat keduanya
13 Tambahkan I dengan 1
14 Lakukan langkah 11 hingga langkah 13 selama nilai I < 10
15 selesai

Tentu akan lebih ringkas jika kita tulis (perintah BASIC) :

```
10 For I= 1 to 10
20 If A(i) > A(I+1) then SWAP A(i), A(j)
30 next
40 end
```

Jadi terlihat, jika algoritma tersebut sederhana, maka penyusunan algoritma akan sama dengan penyusunan sebuah program (karena semua perintahnya sudah sesuai dengan kaidah penulisan di bahasa pemrogramannya).

Apakah semuanya akan demikian ?. Tentu saja tidak, misalkan, kita diminta untuk menentukan bilangan terkecil dari seratus buah bilangan yang akan dimasukkan ke komputer, ini masih dapat langsung dibuatkan programnya.

Algoritma (program)nya bisa kita susun sebagai berikut :

```
1 DIM A(100)
2 FOR M = 1 TO 100
3 INPUT A(M) : NEXT : KECIL = A(1)
4 FOR M = 2 TO 100
5 IF KECIL > A(M) THEN X = KECIL: KECIL = A(M) : A(M) = X
6 NEXT : PRINT KECIL : END
```

Tetapi, misalkan jika kita diminta untuk mengalihkan notasi *infix* menjadi *postfix* melalui *stack*, hal itu sulit untuk dilakukan.

Algoritmanya bisa menggunakan gabungan kalimat dengan bahasa pemrograman, berikut contoh penggalannya.

Contoh :

1. Asumsi : deretan notasi *infix* dimasukkan ke dalam sebuah variabel *array* bernilai *string*, nama variabelnya D
2. S adalah variabel *string* untuk menyimpan susunan data di dalam *stack*
3. H adalah variabel *string* untuk menyimpan hasil
4. P = banyaknya elemen *array*
5. For I = 1 to p

```
        If top(s) = empty then          {top(s) adalah posisi atas stack)
          if D(i) = operand then
            H = D(i)
             Else
             S = S + D(i)
             Top(s) = D(i)
          Endif
          Else
          If D(i) = operator then
            If derajat D(i) > derajat Top(s) then
            . . .
        . . .
    . . .
 . . .
```

Jadi, terdapat beberapa kata yang tidak dapat dijabarkan langsung ke dalam bahasa pemrograman. Misalkan, kata Top(s), empty, operand, operator, dan derajat.

PERLUNYA PROSEDUR

Toh akhirnya, kita tidak akan mungkin hanya membuat algoritmanya saja melainkan dilanjutkan ke pembuatan programnya. Karenanya, algoritma sebaiknya dibuat sedemikian rupa agar setiap perintah yang ada di dalamnya dapat diaplikasikan langsung ke dalam bahasa pemrograman.

Itulah perlunya prosedur. Misalkan kata "operand" di algoritma di atas yang tidak dapat langsung diaplikasikan di dalam bahasa pemrogramannya, kita buat saja prosedur dari algoritma tersebut yang mendefinisikan apa itu "operand."

Misalkan :

```
1 Procedure OPERAND
2 IF ASC(D(I)) > 64 AND ASC(D(I)) < 91 THEN OP = .T. ELSE OP = .F.
3 RETURN
```

Sehingga, di algoritma utamanya bisa diubah dari :

```
If top(s) = empty then          {top(s) adalah posisi atas stack)
   if D(i) = operand then
      H = D(i)
```

menjadi :

```
If top(s) = empty then          {top(s) adalah posisi atas stack)
   Do Procedure OPERAND
   IF op = .t.
      H = D(i)
```

PERLUNYA STANDAR PENGGUNAAN PERINTAH BAHASA PEMROGRAMAN

Sulit memang membuat standardisasi penggunaan perintah bahasa pemrograman di sebuah algoritma. Sulit karena ada yang hanya memahami satu bahasa pemrogrman saja sehingga ia tak mau menggunakan perintah di bahasa pemrograman lain.

Namun, itu sebatas cara penulisan saja, misalkan di BASIC A = 10, di Pascal berlaku A := 10, namun untuk perintah *looping*, umumnya memiliki alur logika yang sama, yaitu dalam penggunaan FOR-NEXT, REPEAT-UNTIL, DO WHILE-ENDDO, WHILE-WEND, dan sebagainya.

Jadi, meskipun tidak ada standar yang pasti, paling-paling hanya berbeda cara penulisannya saja, namun sama dalam alur logikanya. Jadi, ternyata, standardisasi semacam ini tidak diperlukan.

LATIHAN SOAL ALGORITMA

1. Alihkan notasi *infix* menjadi *prefix*
2. Buat suatu pohon biner, tentukan algoritma perjalanan *preorder*;
3. Alihkan notasi *infix* menjadi *postfix* melalui operasi *stack*;
4. Jadikan seratus angka yang sudah dimasukkan ke komputer secara acak menjadi urut (*sort*) dari kecil ke besar (*ascending*)
5. Cari nilai tengah dari seratus angka yang sudah dimasukkan ke komputer secara acak.