

Kelemahan Programmer Muda



Salah satu kesalahan yang banyak ditemui terhadap programmer baru yang masih muda adalah kurangnya wawasan tentang teori dan apa-apa yang sudah dikerjakan oleh orang lain. Seringkali ketika diberikan sebuah tugas, sang programmer ini dengan gagahnya langsung cepat-cepat melakukan pengodean (*coding*). Dia tidak mau mencari informasi tentang berbagai solusi yang mungkin telah ditemukan orang untuk mengatasi masalah atau tugas yang diberikan kepadanya.

Saya ambil sebuah contoh, pencarian (*searching*) sebuah data tertentu dalam sebuah kumpulan data. Sebagian besar programmer muda ini langsung membuat *loop* “*for*” atau “*while*” yang menelusuri kumpulan data tersebut. Dia tidak peduli tentang struktur data dari kumpulan data tersebut (yang bisa jadi berupa *array*, *tree*, *graph*). Strategi yang dipilihnya pun asal-asalan. Dia tidak mau memikirkan apakah lebih baik menggunakan *Depth First Search* (DFS) atau *Breadth First Search* (BFS). Padahal di luar sana sudah banyak literatur yang membahas tentang *search* beserta aplikasi-aplikasinya. Selain masalah pencarian, masalah lain yang mirip adalah pengurutan (*sorting*).

Akibat dari ketidaktahuan ini seringkali sang programmer membutuhkan waktu yang lama untuk menemukan solusinya. Jika sudah berhasil membuat solusinya pun, ternyata solusi yang dibuat oleh sang programmer ini tidak efisien. Programnya bisa jalan untuk data yang jumlahnya sedikit. Begitu jumlah data dinaikkan, program menjadi tidak jalan, atau membutuhkan waktu yang lama untuk menyelesaikan tugasnya. Bayangkan, ada aplikasi yang membutuhkan waktu berjam-jam hanya untuk melakukan *sorting*. Hal ini sering disebabkan kesalahan dalam pemilihan struktur data beserta algoritma yang digunakan untuk memproses data tersebut. Solusinya, sering-sering baca buku-buku, jurnal, dan teori-teori pemrograman untuk menemukan solusi yang lebih baik.

Jika kesalahan di atas disebabkan karena kurang pemahaman tentang teori, maka kesalahan lain yang sering dilakukan oleh programmer muda adalah kurang seringnya praktik membaca atau melihat *source code* orang lain. Ketika dia diminta untuk membuat program dalam bahasa C, maka programnya seperti di bawah ini:

```
#include <stdio.h>
main()
{
printf("Hello world\n");
}
```

Contoh di atas membuat mata saya sakit ketika membacanya. Apa salahnya? Ada banyak. Peletakan kata kunci “*main*” yang masuk ke dalam tanpa sebab, peletakan tanda kurung kurawal pembuka dan penutup yang seenaknya (tidak lurus di baris sendiri atau mengikuti baris sebelumnya) merupakan hal yang menyebalkan bagi saya. Apakah Anda tidak gemas melihat *source code* semacam itu? Contoh pemrograman dalam bahasa lain juga hampir mirip kesalahannya.

Apa yang dapat kita pelajari dari contoh ini? *Style* dari penulisan program ternyata harus juga dikuasai oleh seorang programmer. Program sebaiknya

tidak hanya sekadar jalan, akan tetapi harus juga mudah dan indah jika dibaca. (Ingat artikel lalu tentang seni dan pemrograman?) Ada beberapa aliran *style* dalam penulisan program. Misalnya, ada yang menggunakan “*tab*” untuk *indentation*, tapi ada yang menggunakan spasi (*spacebar*). Dua-duanya dapat Anda lakukan, tapi Anda harus konsisten. Jika Anda menggunakan *tab* atau spasi, berapa karakter kosong yang Anda gunakan?

Kemampuan pemrograman ini sama seperti kemampuan kita memainkan alat musik, gitar misalnya. Seorang pemain gitar yang andal perlu tahu teori-teori (*chord*, *scale*, *picking*), sering berlatih secara rutin, dan juga perlu melihat contoh *style* cara bermain gitaris kawakan. Seringkali ada hal-hal yang lebih cepat dipahami setelah melihat teori tersebut dipraktikkan.

Sering-sering melihat atau membaca *source code* dari programmer kawakan akan memperkaya pengetahuan Anda dalam bidang pemrograman. Pendekatan *open source*, di mana kita bisa melihat *source code* dari sebuah program yang kita kagumi, membuat kita lebih mudah dalam meningkatkan kemampuan pemrograman kita. *Nah*, mari kita giatkan untuk membaca jurnal (teori) dan melihat *source code* (praktik). ☺

Bayangkan, ada aplikasi yang membutuhkan waktu berjam-jam hanya untuk melakukan *sorting*.