

# Menguji Software



Beberapa waktu lalu, saya menguji seorang mahasiswa yang membuat sebuah rangkaian *hardware* untuk tugas akhirnya. Dengan bangganya dia menceritakan fitur-fitur rangkaian yang dibuatnya. Saya kemudian menanyakan bagaimana dia yakin bahwa rangkaianannya memang bekerja seperti yang diharapkan. Pada prinsipnya saya bertanya bagaimana cara dia menguji karyanya.

*"Begini pak, saya sudah tes rangkaian saya dengan dua tes. Yang pertama menggunakan pola semua nol dan yang kedua dengan menggunakan pola semua satu."*

*"Sebentar. Anda mengatakan bahwa Anda menggunakan dua pola?"* potong saya.

*"Ya, pak. Sudah saya tes kesemuannya."* Masih dengan gaya yang stil yakin.

*"Hah? Maksud Anda, pola tesnya hanya dua???"* Saya terbelalak.

*"Eh... Ya, pak."* Mahasiswa mulai tampak grogi dan gemeteran.

Saya kemudian mulai nyerocos menceritakan pentingnya pengujian.

Kejadian seperti di atas saya alami beberapa kali ketika menguji mahasiswa yang membuat *hardware* dan *software*. Bahkan di sisi *software* biasanya lebih sembrono lagi. Tampaknya, mahasiswa yang saya uji ini tidak mengerti pentingnya pengujian. Lebih mengerikan lagi, ternyata saya mendengar bahwa pengujian di kalangan pembuat *software* di luar kampus juga belum mendapat perhatian.

Pengujian dilakukan untuk meyakinkan bahwa *software* memang sudah benar sesuai dengan yang diinginkan dan tidak memiliki bugs. Sayangnya, hal ini tidak mudah. Biasanya pengujian dilakukan dengan menggunakan beberapa pola (*test pattern*) yang dianggap mewakili kemungkinan *input*. Pendekatan ini menggunakan basis statistik, yang mengatakan bahwa semakin banyak sampel (pola) pengujian yang digunakan semakin baik tingkat kepercayaan kita kepada sistem yang diuji tersebut. Tentunya, pengujian dengan dua data saja tidak bisa dikatakan sudah cukup secara statistik.

Namun, ada masalah lagi. Untuk meyakinkan tidak ada kesalahan, semua kombinasi input harus diuji. Padahal seringkali tidak mungkin sistem diuji dengan segala kombinasi. Pakar komputer Edsger W. Dijkstra mengatakan, *"Non-exhaustive testing can be used to show the presence of bugs, but never to show their absence."* Artinya,

pengujian yang tidak menggunakan semua kombinasi tidak dapat digunakan untuk membuktikan bahwa *software* tersebut tidak memiliki bugs.

Saya ambil sebuah contoh cerita lagi. Pada suatu ketika ada seorang dosen yang memberikan tugas pemrograman kepada siswanya, yaitu membuat sebuah program yang dapat menampilkan bilangan prima di atas bilangan dua. Ada seorang siswa dengan cepat membuat programnya dan kemudian dia uji. Ketika dijalankan hasilnya keluar angka "3" dan "5". Puas dia. Dia serahkan kepada dosennya. Sang dosen kemudian menguji program buatan siswa ini. Dia jalankan dan tampil angka "3", dan "5". Sang dosen belum puas, diteruskannya program tersebut, keluar angka "7". Sang mahasiswa sudah mulai senang. Dosen masih belum puas. Diteruskannya program tersebut, dan ternyata keluar angka "9".

Sang mahasiswa kaget, kenapa demikian. Kemudian sang dosen merunut kode sang mahasiswa, dan ternyata sang mahasiswa membuat program pen ghasil bilangan ganjil, bukan penghasil bilangan prima.

Sambil bersungut-sungut, sang mahasiswa mengubah programnya dengan logika kira-kira sebagai berikut:

```
if (hasil sama dengan 9) do not print
```

Dia jalankan kembali programnya dan dia puas dengan hasilnya. Tampilannya adalah "3, 5, 7, 11". Sudah benar pikirnya. Dia serahkan kembali hasilnya ke dosennya. Karena sudah pernah gagal sebelumnya, sang dosen kali ini lebih berhati-hati dalam menguji program tersebut. Dia jalankan program tersebut dengan lebih lama. Hasilnya, "3" ... (mahasiswa berdoa) "5" ... (mahasiswa mulai tersenyum) "7" ... (mahasiswa mulai tenang) "11" ... "13" (mahasiswa mulai bersorak di dalam hati), dan kemudian ... "15". Aduh! Sang mahasiswa pingsan.

Lantas bagaimana cara kita meyakinkan bahwa *software* kita sudah benar? Sebetulnya ada satu bidang yang disebut *"formal verification"*. Pembuktian secara matematis digunakan untuk membuktikan kebenaran sebuah program. Namun tampaknya, ini masih terlalu sulit sehingga belum dapat digunakan secara praktis. Kita terpaksa masih menggunakan pengujian secara konvensional. Apa boleh buat. Tapi, lain kali kalau membuat sebuah program, jangan diuji dengan dua data saja ya. ☺

**Pengujian dilakukan untuk meyakinkan bahwa software memang sudah benar...**