IF1210 Algoritma dan Pemrograman 1

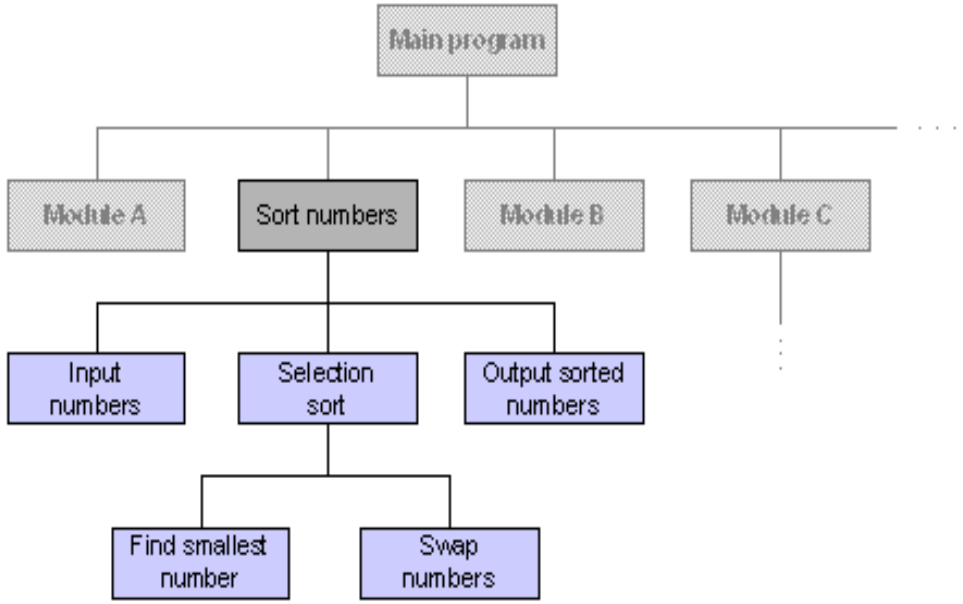
Source Code Standard/ Convention

Tim Pengajar IF1210

Sekolah Teknik Elektro dan Informatika

Contoh (1)

- Pembuatan software untuk *spreadsheet* atau basis data
- Salah satu modul yang diperlukan:
  - Melakukan pengurutan (*sorting*)



Sumber: http://courses.cs.vt.edu/csonline/SE/Lessons/Procedural/index.html

Contoh (2)

- Pembuatan software untuk *spreadsheet* atau basis data
- Tiap modul dibuat oleh orang yang berbeda, masing-masing memiliki 'style'
- Harus ter'integrasi'kan dengan baik → perlu **standar**

Sumber gambar:
http://www.oncoursesystems.com/school/webpage/11191090/1181281

Coding Standard

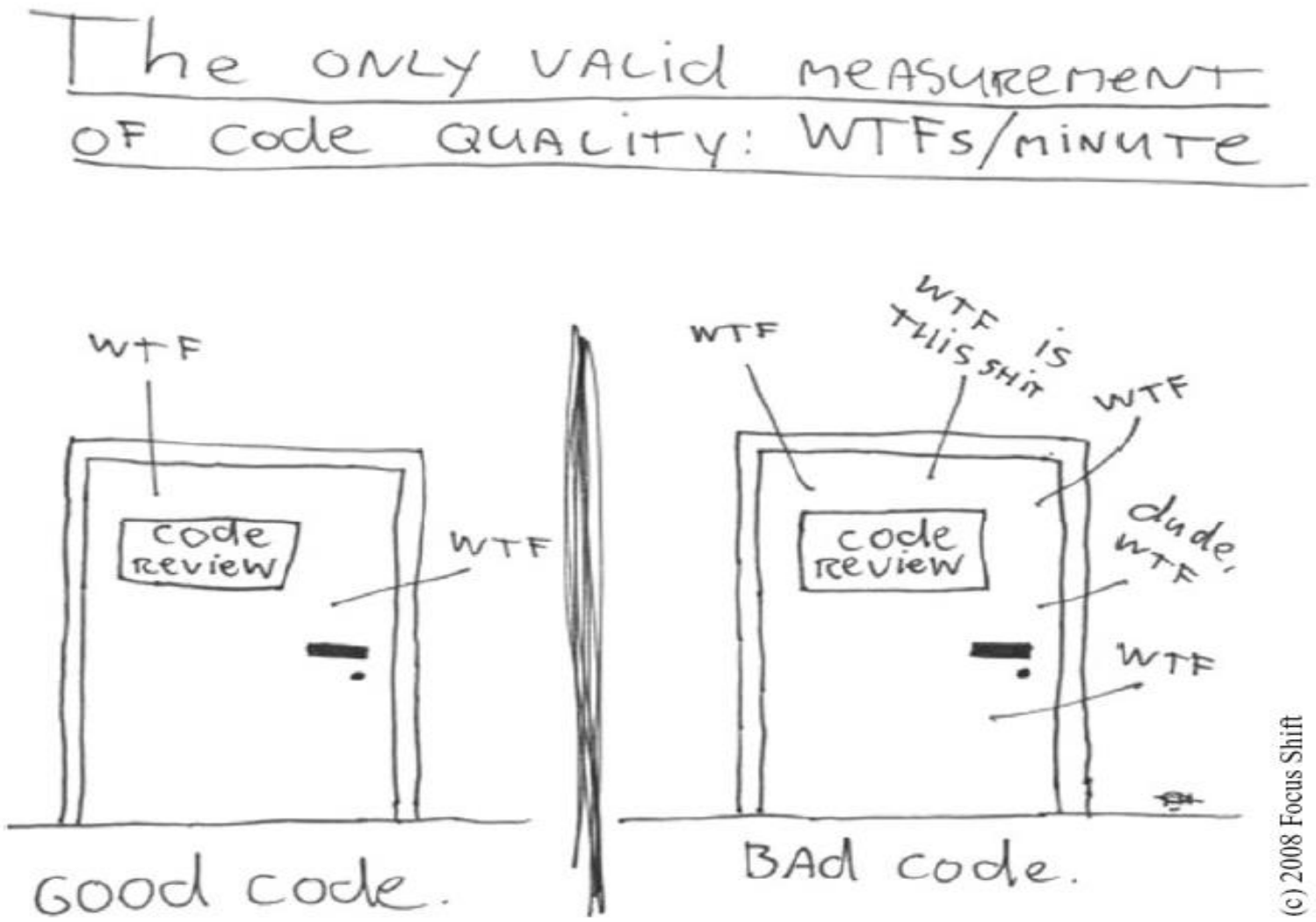
"The best applications are coded properly. This sounds like an obvious statement, but by 'properly', I mean that the code not only **does its job well**, but is also **easy to add to, maintain and debug**."

David Mytton, founder ServerDensity,

http://www.sitepoint.com/coding-standards/

Video: http://www.youtube.com/watch?v=HjpjpJ-xuvQ (example of bad coding practice)

Intermezzo: Sebaik apakah kode sumber anda?



Reproduced with the kind permission of Thom Holwerda.
http://www.osnews.com/story/19266/WTFs_m

Intermezzo: Kode anda harus 'BERSIH'

Sumber:
Robert C. Martin, Clean Code, Prentice Hall

Kode yang ‘BERSIH’

Kode yang:

1. Elegan & Efisien
2. Simple & Direct
3. Mempunyai *meaningful names*
4. Mempunyai *minimal dependencies*
5. Lulus pada semua tes
6. Tidak ada duplikasi
7. Mempunyai jumlah *class, function, method*, dll sesedikit mungkin (*tiny abstraction*)
8. Mudah dibaca oleh orang lain

The Need of Good Coding Practices (1)

- Readability
  - Program harus dapat dibaca dan dipahami dengan cepat dan baik oleh diri sendiri maupun orang lain
- Maintainability
  - Program dengan readability yang baik akan lebih mudah dipelihara
- Menghindari **code bad smells**
  - Tidak ada hubungannya dengan eksekusi, tetapi lebih pada kebaikan source code

The Need of Good Coding Practices (2)

- Dibutuhkan standar yang sama dalam penulisan kode
  - *Practice* yang umum dilakukan, setiap perusahaan bisa menetapkan *coding standard* sendiri
- Beberapa *practice* bila dilakukan dengan baik bahkan bisa mengarahkan pada program yang benar

Konvensi Penting IF1210

- Standar blok program
- Indentasi
- Pemakaian komentar dengan wajar dan baik
- Nama-nama yang baik dan *meaningful* untuk type, variabel, konstanta, fungsi/prosedur, dll.
- Hanya menulis kode yang memang dipakai
- Pemakaian *construct* dasar program dengan sebaik-baiknya
- Penggunaan *standard schema*

Standar Blok Program

- Blok program prosedural terdiri atas:
  - Header program: judul program dan spesifikasi
  - Kamus
  - Algoritma

- **Penting**: Jika Anda menulis program utuh (misalnya di praktikum) selalu tuliskan:
  - Identitas pemrogram (NIM/Nama)
  - Tanggal penulisan program → menunjukkan versi

Standar Blok Program dalam Notasi Algoritmik

Program <JudulProgram> { Spesifikasi Program }
KAMUS { Deklarasi type, variabel, konstanta, fungsi, prosedur }
ALGORITMA { Deretan langkah algoritmik untuk penyelesaian persoalan } { Memanfaatkan **notasi algoritmik** }

Standar Blok Program C

```
/* Identitas dan tanggal */
/* Program <NamaProgram> */
/* Spesifikasi Program */

int main() {
   /* KAMUS */
   /* Daftar variable */
   …

   /* Definisi, Spesifikasi, dan Realisasi Prosedur
      Fungsi jika ada */
   …

   /* ALGORITMA */
   /* langkah-langkah dalam program utama */
   …
   return 0;
}
```

Tuliskan identitas coder dan tanggal penulisan kode

Tuliskan spesifikasi program

Daftar konstanta dan variabel global

Definisi, Spesifikasi, dan Realisasi fungsi dan prosedur

Program utama

Indentasi (1)

- Gunakan indentasi yang semakin menjorok ke dalam untuk menandai setiap *inner block*
- Khusus untuk program yang diketik:
  - Gunakan spasi dengan jumlah yang sama untuk setiap indentasi baru
  - Tab dan spasi tidak bisa dicampur dengan baik → Gunakan tab dan spasi dengan konsisten

Indentasi (2)

- **Indentasi sangat penting di notasi algoritmik** karena menandai blok program
  - Contoh: 2 potongan kode ini berbeda eksekusinya

```
...
x ← -1
if (x >= 0) then
    output(“Mungkin positif”)
    output(“Mungkin nol”)
...
```

```
...
x ← -1
if (x >= 0) then
    output(“Mungkin positif”)
output(“Mungkin nol”)
...
```

Komentar

- Komentar wajib dalam program (konvensi):
  - identitas + tanggal, spesifikasi program
  - Spesifikasi fungsi dan prosedur
- Komentar untuk hal-hal yang penting
  - Tidak berlebihan sehingga teks penuh komentar

Nama-nama yang baik dan *meaningful* (1)

- Gunakan 1 nama untuk **1 buah keperluan** saja
  - Hindari menggunakan variabel/subprogram dengan nama sama untuk keperluan yang berbeda
- Gunakan nama yang memang menggambarkan hal yang direpresentasikan (*meaningful*)
  - hindari menggunakan nama yang membingungkan atau tidak merepresentasikan apa pun

```
procedure Analisis ( … )
nnnmsspd : integer
akucintakamu : character
```

Nama-nama yang baik dan *meaningful* (2) - Variabel

- Variabel:
  - Perhatikan nama-nama yang memiliki arti dan penggunaan yang dikenal umum: e.g. flag, found, count, sum, idx, min, max, … → hindari menggunakannya untuk keperluan lain
  - i, j, k hanya digunakan dalam *control loop*

Hanya menulis kode yang memang dipakai

- Bagian kode yang berlebihan tidak akan menimbulkan compile-error
  - Beberapa compiler memberikan warning

- Hindari menuliskan bagian kode yang tidak dipakai sama sekali
  - Khususnya dalam menuliskan program utuh (di praktikum dan tugas besar)

Pemakaian *construct* dasar program dengan sebaik-baiknya

- Perhatikan kembali *construct* dasar program prosedural dan gunakan sesuai dengan tujuannya dengan sebaik-baiknya
- Contoh:
  - Analisis kasus:
    - harus disjoint dan complete (walaupun mungkin tidak semua kasus dituliskan aksi yang eksplisit)
    - Kapan menggunakan if-then-else, depend-on, dll.
  - Loop: gunakan jenis loop yang benar→ akan diulas kembali dalam skema pengulangan

Penggunaan *standard schema*

- Menggunakan skema ‘standar’ yang disepakati
- Dalam kuliah IF1210 akan dibahas lebih lanjut beberapa skema dasar dan standar:
  - Skema proses validasi
  - Skema pengulangan/pemrosesan sekuensial
  - Skema pemrosesan array
  - Skema pemrosesan file I/O