Manipolare Codice

Franco Maria Nardini

Manipolare Codice

- Quando si legge codice:
  - Prima capire COSA fa…
  - Poi PERCHE' lo fa…
  - e successivamente COME lo fa…

Manipolare Codice

- Attività importanti:
  - confronto tra sorgenti e documentazione: consistenti?
  - confronto tra realtà e documentazione: consistenti?
  - usare pagine **`man`**: casi di fallimento e successo non considerati?
  - domande
    - si poteva far meglio?
    - …

Importante! I

"One of my most productive days was throwing away 1,000 lines of code."

Ken Thompson

- "Più" codice NON implica "miglior" codice!

Importante! II

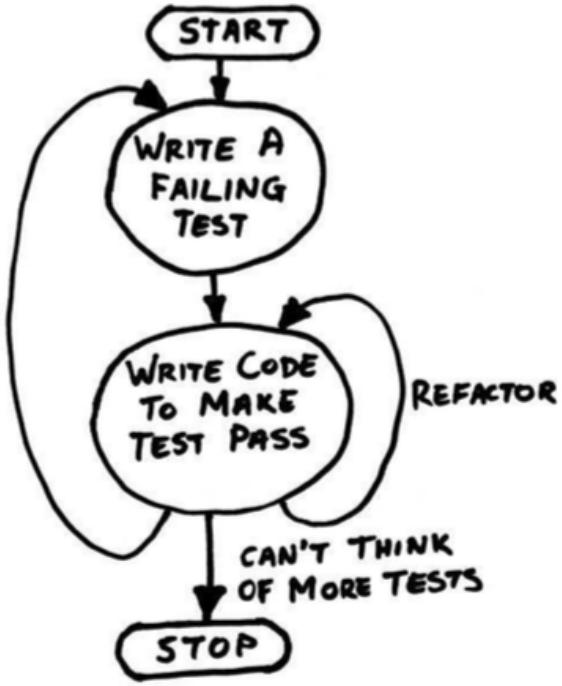
- “Più” documentazione NON implica “più facile” da capire
- Alcuni consigli:
  - nomi di variabili e funzioni auto-esplicativi
  - tenere commenti e documentazione aggiornati
- Un buon commento descrive il “perchè” non “cosa”

Importante! III

```
/* check input size */
if (argc != 3) {

/* open files to be read */
if ((fd1=open(argv[1], O_RDONLY)) == -1) {

/*main function*/
int main(int argc, char *argv[])

/* Close file descriptors */
close(sourcefile);
```

- Evitare commenti inutili!

Importante! VI



- Scrivere con test in mente!

Importante! V

- Utilizzare correttamente **`stdout`** e, soprattutto **`stderr`**
  - **`strerror(3)`**
  - **`perror(3)`**
- Evitare la generazione di output inutile
  - **`printf(“open input success”);`**

Importante! VI

- Spell-checking dei commenti, output, error!
- Uso degli spazi o dei tab: a scelta ma siate consistenti!
- Controllo dei return codes!
- Evitare l'overflow dei buffer!
  - `strcat(3)` pericolosa: usare `strlcat(3)`, `strlcpy(3)`

Importante! VII

- Uso di “regole di stile” per codice
  - "Advanced Programming in the UNIX Environment" source code style guide.
  - Aiutano a tenere il codice “ben scritto”.

```
$ wget http://hpc.isti.cnr.it/~nardini/siselab/style
```