Fondamenti della Programmazione: Metodi Evoluti

Prof. Enrico Nardelli

Lezione 1: Introduzione

Obiettivi formativi

- Padroneggiare la programmazione in modo professionale.
  - Chiunque può scrivere programmi funzionanti
  - Non ci si sporca, né si suda
  - Scrivere programmi funzionanti in modo robusto ed affidabile, anche e soprattutto man mano che li si modifica ed aggiorna, richiede strumenti e tecniche adeguate.
- Cosa usiamo nel corso:
  - il linguaggio orientato agli oggetti Eiffel
  - la metodologia di progetto Design by Contract
  - l'approccio outside-in allo sviluppo di programmi

Schedule

Lezioni:

- Mercoledì, 11:00 – 13:00, Aula 2
- Venerdì, 14:00 – 16:00, Aula 2

Spiegazioni:

- Al termine delle lezioni
- Su richiesta, previo appuntamento per e-mail

Dove sono

Contatti:

- E-mail: nardelli@mat.uniroma2.it
- Telefono: 06 7259 4204 oppure 335 590 2331
- Studio 0123, piano terra, dente 1, Dip. Matematica

Informazioni sul corso

Pagina web del corso (in italiano):

http://www.mat.uniroma2.it/~nardelli/fondamenti-programmazione-metodi-evoluti.html

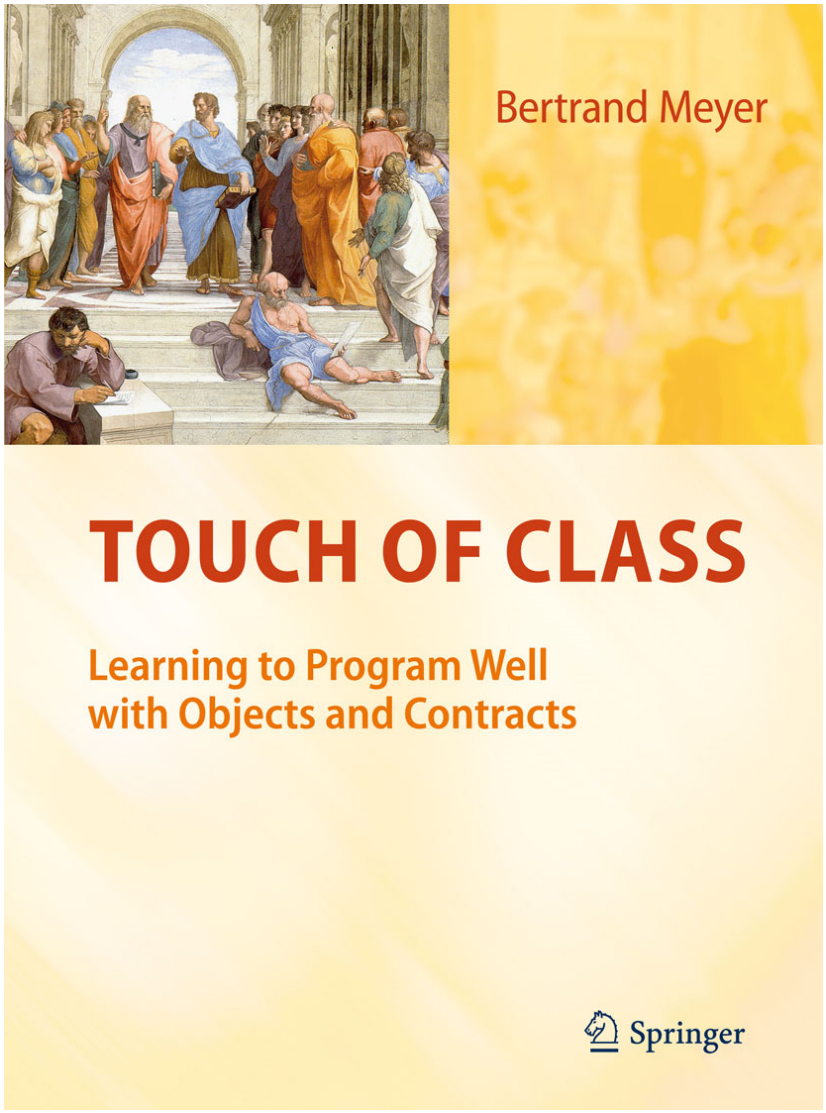
- Informazioni organizzative
- Materiale didattico:
- Informazioni e risultati esami
- Ricevimento studenti

The original page (in english):

http://se.inf.ethz.ch/old/teaching/2009-H/eprog-0021/english_index.html

- Lecture slides (link: *slides and video recordings*)
  - Also audio/video recording of original lectures by B.Meyer
- Exercise material
  - Text and solutions
  - Supplementary slides
- Advanced topics

Il libro di testo



Sulla mia pagina web del corso

- Lucidi di lezioni ed esercitazioni
- Libro di testo
- Risorse didattiche disponibili in rete
- Risorse per l'ambiente di sviluppo usato
- Programma del corso
- Modalità d'esame
- Prove d'esame svolte

Software usato

Ambiente di sviluppo:

EiffelStudio

Scaricabile (versione libera) da

https://sourceforge.net/projects/eiffelstudio/

cliccare "See All Activity" e poi scrollare fino a trovare la prima versione GPL per la vostra piattaforma (Windows, Unix, Linux, ...)

Sito di riferimento http://dev.eiffel.com/

Ulteriori informazioni sulla pagina web del corso

Installarlo per la prossima lezione e portare il PC

Alcuni consigli

- Frequentare le lezioni
- Leggere il materiale prima della lezione
- Seguire con la stampa delle slide e prendere appunti
- Frequentare le esercitazioni
- Fare tutti gli esercizi
- Se non si capisce: chiedere
  - NON CI SONO DOMANDE STUPIDE!
- Non preparare gli esami all'ultimo momento
- Avere sempre un atteggiamento critico e indagatore

Computer dovunque

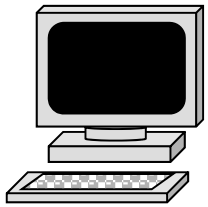
Banche, uffici, ...

Aeroplani, automobile, ...

Lavatrici, elettrodomestici, ...

Cellulari, smartphone, ...

Oggetti personali...



In tutte le dimensioni, colori e varianti

Computer

È una macchina universale (nel senso di Turing-Church: può calcolare qualunque funzione calcolabile)

Esegue il programma che gli è stato dato: l'unico limite è la tua immaginazione.

La buona notizia:

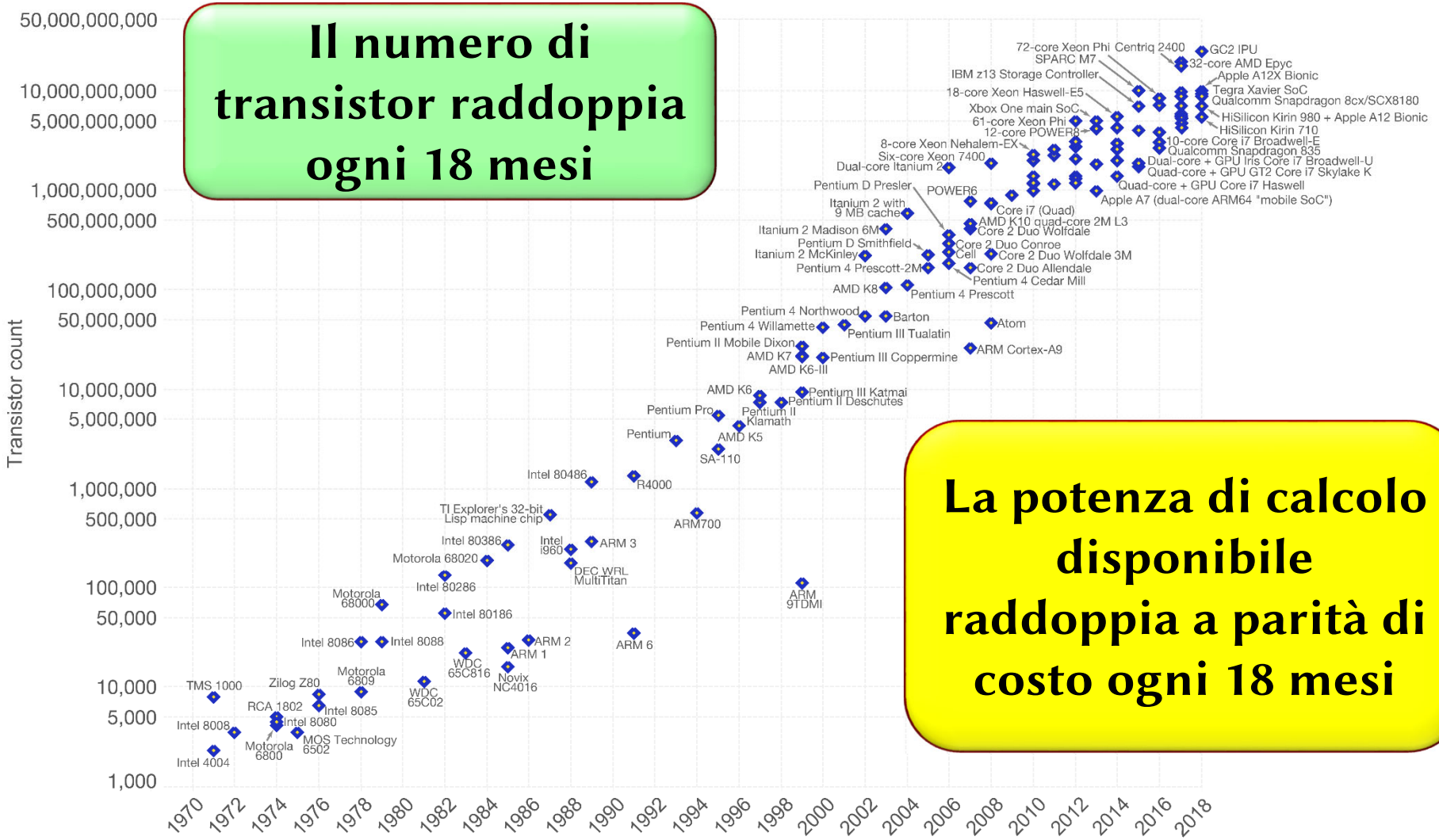
- Il tuo computer farà esattamente ciò che c'è scritto nel tuo programma
- Lo farà molto velocemente.

Legge di Moore (da Wikipedia)

Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.

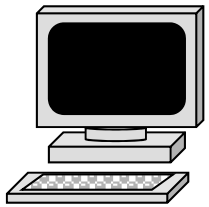
Our World in Data



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Computer

Attraverso il programma che gli forniamo il computer diventa una macchina specializzata



\+

Programma

=

Macchina specializzata

Macchina universale

Software

Software dovunque

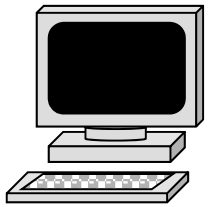
Banche, uffici, ...

Aeroplani, automobile, ...

Lavatrici, elettrodomestici, ...

Cellulari, smartphone, ...

Oggetti personali...



Dati e informazioni

I dati sono come l'informazione è rappresentata (p.es. in un computer come un file MP3 audio), sono i simboli che costituiscono la rappresentazione

L'informazione è ciò che ha valore per te (immagini, musica, testo, ...), interpretazione umana dei dati

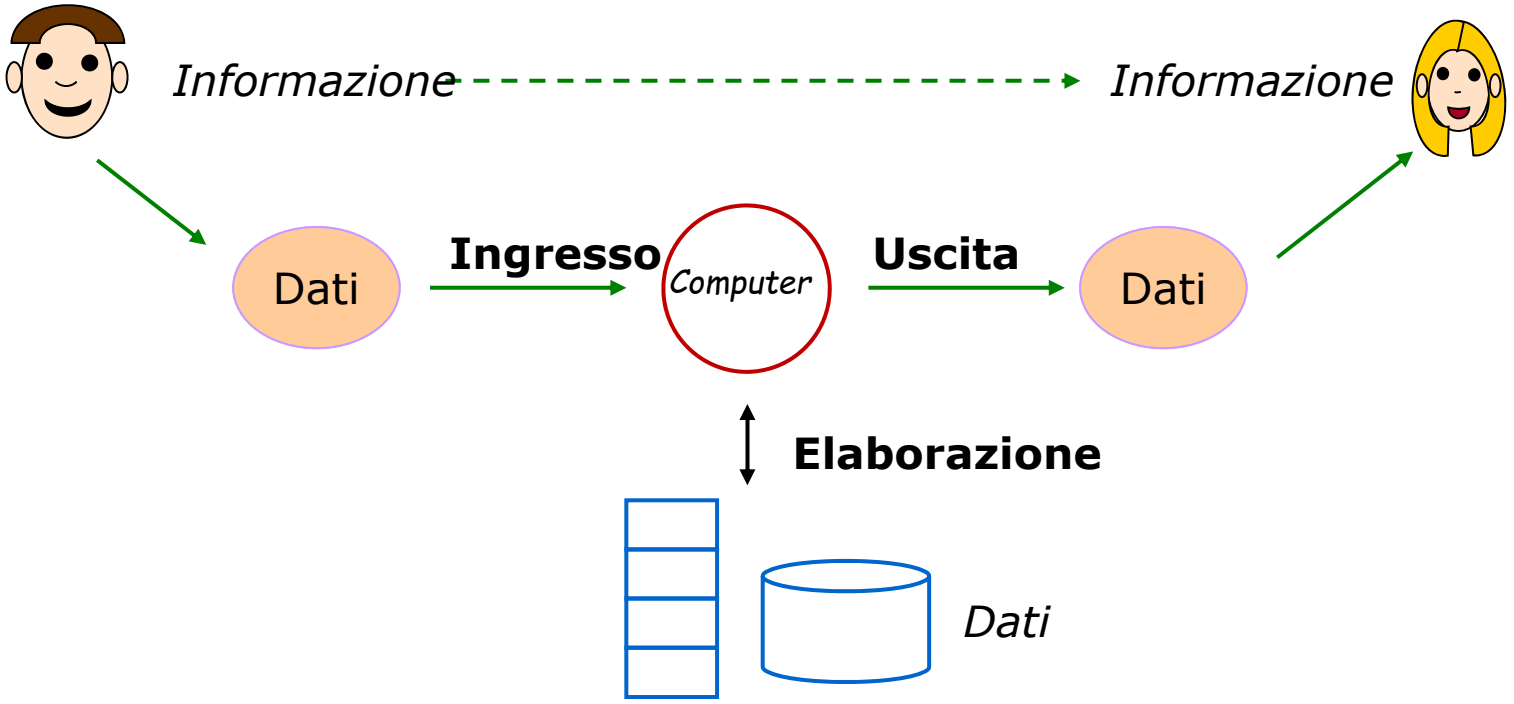
Teoria matematica dell'informazione (C.Shannon): riduzione dell'incertezza del ricevente

Informazione e elaborazione dei dati

Dispositivi di ingresso acquisiscono dati da informazione fornita dalle persone

Dati elaborate dal computer e memorizzati nella memoria

Dispositivi di uscita restituiscono dati recepiti come informazione dalle persone



Miti e scuse comuni

"Computer sono intelligenti"

> *Fatto: I computer non sono intelligenti o stupidi. Eseguono programmi definiti dalle persone. I programmi riflettono l'intelligenza dei loro autori.*
>
> *Le operazioni di base di un computer sono estremamente elementari (memorizza un valore, somma due numeri, ...).*

"Il computer non me lo permette"

"Il computer ha perso i dati"

"Il computer non ha capito"

I computer non fanno errori *....

- Neanche i programmi fanno errori
- Programmatori FANNO errori

*In realtà, l'hardware può avere dei guasti, ma è un evento estremamente più raro di un errore di programmazione

Computer

Il computer è una macchina universale.

Esegue il programma che gli è stato dato: l'unico limite è la tua immaginazione e la tua attenzione

La buona notizia:

- Il tuo computer farà esattamente ciò che c'è scritto nel tuo programma
- Lo farà molto velocemente.

La cattiva notizia:

- Il tuo computer farà ESATTAMENTE ciò che c'è scritto nel tuo programma
- Lo farà molto velocemente

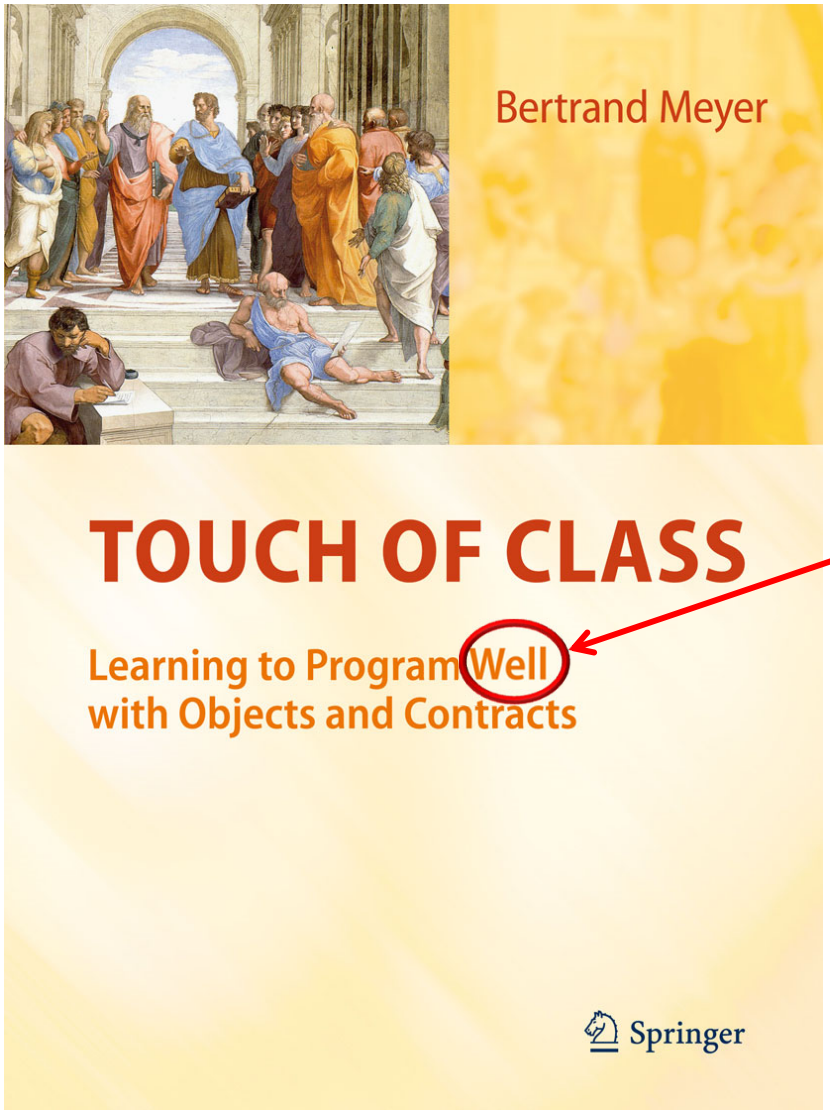
"Sbagliare è umano, ma per complicare davvero le cose serve un computer"

Scrivere "BENE" il software è difficile

- Programmi terminano per errore (*crash*)
- Programmi possono non funzionare anche senza terminare per errori
- Programmi scorretti uccidono le persone (p.es.: nei dispositivi medici)

- Programmatori sono responsabili del corretto funzionamento dei loro programmi
- Programmatori sono responsabili degli aspetti etici del loro lavoro
- Lo scopo del corso non è tanto insegnare la programmazione ma la BUONA programmazione

Imparare a programmare *BENE*



Un errore software un po' costoso...

Ecco le conseguenze di un errore di programmazione:

https://www.youtube.com/watch?v=gp_D8r-2hwk

Razzo Ariane 5, 1996: 370 milioni US$ persi per un semplice errore di programmazione (7 miliardi US$ è il costo totale dello sviluppo del razzo)

- Jean-Marc Jézéquel & Bertrand Meyer: *Design by Contract: The Lessons of Ariane*, in *Computer* (IEEE), vol. 30, no. 1, January 1997, pag 129-130, archive.eiffel.com/doc/manuals/technology/contract/ariane/.

I peggiori fallimenti del software

10 peggiori errori software (fino al 2005):

http://www.wired.com/software/coolapps/news/2005/11/69355?currentPage=all

Un resoconto più recente

http://www.computerworld.com/s/article/9183580/Epic_failures_11_infamous_software_bugs

Basta cercare "*worst software failures*" oppure "*worst software bugs*"

Ingegneria del software

Come l'ingegneria tradizionale ha lo scopo di arrivare a scrivere software con le seguenti qualità

- Correttezza
  Fa quello che dovrebbe fare
- Estendibilità
  Facile da modificare
- Leggibilità
  da parte delle persone
- Riusabilità
  per non reinventare sempre le stesse cose
- Robustezza
  Reagire bene a malfunzionamenti
- Sicurezza
  Resistere agli attacchi

Scrivere “BENE” il software è difficile...

È difficile ottenere programmi con tutte le qualità elencate

Non si può procedere per tentativi, servono metodi appropriati (*object-oriented* è uno di questi)

... ma scrivere software è bello e gratificante

Progetti e costruisci le macchine che vuoi

Eserciti creatività e immaginazione

Puoi migliorare la vita delle persone e rendere il mondo un posto migliore

L'esperienza di vedere che un programma che hai pensato e realizzato funziona perfettamente è molto soddisfacente!