

## **PROGRAMMAZIONE II(canale A-D)a.a.2004/2005**

**Prof.ssa EMANUELA FACHINI,  
STUDIO: D.to Informatica, Via Salaria,113  
TEL. 0649918314**

**E-MAIL: fachini@di.uniroma1.it**

**ORARIO DI RICEVIMENTO: venerdì ore 14 - 16 o anche in coda alle lezioni, quando possibile.**

pagina del corso:

<http://twiki.dsi.uniroma1.it/twiki/view/Programmazione2ad/WebHome>

**Esercitazioni: dott.ssa Chiara Petrioli**

**TESTO CONSIGLIATO:**

[Al Kelley, I. Pohl C](#), didattica e programmazione, Addison-Wesley, traduzione italiana a cura del Prof. G. Pighizzini, Univ. Milano

# Programma di Programmazione II

## a.a. 2006/2007

**Ricorsione.** Richiami di programmazione ricorsiva: i fondamenti della ricorsione; lo stack delle chiamate ricorsive; esempi di programmi ricorsivi. La "tail recursion". Richiami su strutture dinamiche ricorsive: le liste.

**Cenni di complessità di spazio e di tempo per gli algoritmi e i programmi.** Valutazione asintotica e miglioramento delle costanti moltiplicative. Confronto tra soluzioni iterative e ricorsive.

**Tecniche per la scrittura di programmi corretti.** L'importanza della specificazione precisa di un programma. Precondizioni e postcondizioni delle funzioni. Uso di assert per la loro verifica. Strutturazione dei programmi: sottoprogrammi e moduli. Tipi di dato astratti. L'information hiding. Pile e code: definizione e implementazione mediante liste e vettori. Invarianti come strumenti di progettazione di cicli. Applicazione del metodo al problema dell'ordinamento.

**Alberi binari.** definizione e implementazione. Proprietà degli alberi binari: altezza, foglie, relazione tra numero dei nodi e altezza in un albero binario qualunque, alberi pieni, completi, criteri di bilanciamento. Gli alberi binari di ricerca.

**Tecniche di testing.** La scelta dei dati di test negli approcci Black-box e glass-box.

**Nel passato sono stati prodotti moltissimi programmi che**

- **terminano imprevedibilmente (crash),**
- **non producono risposte corrette (pieni di bugs),**
- **sono costosi da correggere o modificare, e**
- **contengono errori nascosti (timebombs) che si manifesteranno prima o poi**

# PROGRAMMAZIONE II

## Requisiti dei programmi

- **correttezza**
- **efficienza**
- **facilità di lettura, e quindi comprensione**
- **facilità di manutenzione**
- **robustezza**
- **portabilità**
- **riusabilità**

**proprietà oggettiva**  
**proprietà oggettiva**  
**non quantificabile**

“

“

“

“

# L'obiettivo più difficile (!) è scrivere programmi corretti

## Cosa fare :

**Progettare accuratamente il programma scegliendo le strutture dati più convenienti, suddividendolo in moduli coerenti per obiettivo (per esempio tutte le funzioni di I/O,...)**  
**Per ogni compito che deve essere eseguito scrivere la relativa funzione in pseudocodice e verificarne la correttezza**

## Cosa **non** fare:

**Scrivere il programma senza preoccuparsi dello stile o della strutturazione e affidare ogni verifica della correttezza al testing**