RISOLVERE
I TEMI D'ESAME

UN PARAGONE…

SVILUPPO DI UN TEMA

- **Leggere bene la traccia**
  (evitare di andare "fuori tema")
- **Pensare a cosa si sa e a cosa si può scrivere,**
  quali tematiche toccare, ecc.
- **Pensare (meglio ancora scrivere!) uno schema:**
  come organizzare le informazioni?
- **SOLO DOPO, stendere il tema**
  (è possibile modificare lo schema)

ERRORE TIPICO: buttarsi a scrivere tutto "di getto"

SVILUPPO DI UN PROGRAMMA C

- **Capire il problema**
  (eventualmente: provare a risolvere qualche istanza!)
- **Pensare a come risolverlo (come faremmo noi?)**
- **Arrivare ad un algoritmo per passi successivi, partendo da una descrizione più astratta e poi scomponendola in passi sempre più elementari…**
  [NB: è possibile tornare indietro e modificare lo schema di partenza]
- **… fino ad arrivare al linguaggio C**

ERRORE TIPICO: buttarsi a scrivere il codice "di getto"

I PASSI PRINCIPALI

- Capire il problema e, eventualmente, ***risolvere delle istanze "a mano"***
- ***Scomporre il problema in sottoproblemi***, affrontandoli uno alla volta (metodo dei "raffinamenti successivi")
  - se un [sotto]problema è complesso, individuare "informalmente" un metodo risolutivo, poi procedere in modo più preciso
  - usare "dove serve" lo ***pseudocodice***
  - ***affrontare prima il "caso generale"***, i sottocasi specifici dopo [ERRORE TIPICO: cominciare il programma con una sequela di if]
  - nella ***stesura di un ciclo***, occorre:
    - > identificare le variabili "chiave" necessarie
    - > attribuire il significato preciso che devono assumere all'inizio di ogni iterazione (quando viene controllata la condizione) e attenersi a questo significato (p.es. nella specifica della condizione di permanenza)

LE COSE PIU' DIFFICILI DA IMPARARE:

- Applicare il metodo dei raffinamenti successivi "al giusto livello di astrazione": l'astrazione deve semplificare il processo risolutivo permettendoci di concentrarci sul cuore del problema stesso.

  ESEMPIO

  Acquisisci dati
  Elabora dati
  Stampa risultati

  } Si adatta praticamente a tutti gli esercizi, ma non serve a nulla!

- Imparare a "tornare indietro" fin dalle prime fasi del processo di scomposizione per raffinamenti successivi, per giungere ad una stesura più semplice o più ordinata del codice

COME IMPARARLE

- Purtroppo o per fortuna, non esistono regolette da applicare

- Come in molti sport:
  - bisogna imparare una ***tecnica*** che ha una sua teoria ma…
  - bisogna fare ***esercizio*** ma…
  - bisogna abituarsi ad applicare gli accorgimenti tecnici già quando si affrontano esercizi semplici: solo così la tecnica verrà poi applicata "spontaneamente" negli esercizi più complicati

  ⇒ Studiare solo "sulla carta" non porta lontano

  ⇒ Anche "bruciare le tappe" non porta lontano: bisogna sforzarsi di applicare la tecnica fin da subito

AVVISI

- Vedremo una serie di esercizi tratti dai temi d'esame:
  in un singolo esercizio, i diversi "passi" possono avere un ***ruolo più o meno accentuato*** [per alcuni è più complesso il processo di raffinamento successivo, per altri si arriva direttamente al codice ma la chiave è dare un significato preciso alle variabili, ecc. ecc.]

- In aula, vedremo ***pochi esercizi significativi***: tanti esercizi sarebbero controproducenti, perché quello che è importante è
  - ***esaminare approfonditamente*** pochi esercizi significativi, per ***imparare "in teoria" la tecnica***
  - ***provare a rifarli*** da soli (NB: non esiste una sola soluzione!) per ***imparare "in pratica" la tecnica***
  - ***provare poi da soli*** (con l'aiuto del compilatore) a farne altri, per ***far diventare la tecnica "spontanea"***

QUINDI:

• Faremo in aula esercizi tratti dai temi d'esame e verranno poi proposti **esercizi d'esame individuali senza soluzione** perché:

- è possibile risolverli studiando bene gli esempi già visti: se non avete idea di come risolverli, significa che non avete capito bene la tecnica "in teoria" e non bisogna guardare la soluzione, ma **riesaminare gli esempi fatti in aula e rifletterci su**
- per imparare la tecnica "in pratica" dovete **arrivare in fondo da soli**, senza guardare la soluzione (altrimenti non imparate)
- è assolutamente indispensabile commettere errori ma abituarsi a scoprirli – e correggerli – da soli (usando il compilatore!) per poi non commetterli più: guardando la soluzione uno non sbaglia e **se non si sbaglia non si impara**