FONDAMENTI DI PROGRAMMAZIONE
Corso di Laurea in MATEMATICA
a.a. 2023/2024

Chiara Bodei, Roberta Gori, Damiano Di Francesco Maesa

Dipartimento di Informatica
chiara.bodei@unipi.it, roberta.gori@unipi.it, damiano.difrancesco@unipi.it

Obiettivi

Pensare come un informatico o capire come pensa :)

Pensare come un informatico è molto di più che programmare un computer!

> *Computational thinking will be a fundamental skill used by everyone in the world by the middle of the 21st Century*[1]

- Il "pensiero computazionale": processo mentale che ha a che fare con la risoluzione di problemi (problem solving). Vedi video tratto dall'Apollo 13: `https://www.programmailfuturo.it/progetto/cose-il-pensiero-computazionale`
- Ogni problema si affronta, usando opportunamente l'astrazione e la decomposizione, ricorrendo alle tecniche sviluppate dall'informatica.
- Questo modo di pensare influenza altre discipline: biologia, neuro-scienze, chimica, etc.

[1] J.M. Wing, "Computational Thinking," CACM Viewpoint, March 2006, pp. 33-35.

Pensiero Computazionale in cinque parole-chiave

http://link-and-think.blogspot.it/2016/04/cinque-parole-chiave-pensiero-computazionale.html

Astrazione

- Astrazione: in informatica indica il processo concettuale con il quale si passa da una descrizione di basso livello a una descrizione dello stesso concetto a un livello più alto, che subordina alcuni dettagli ed evidenzia caratteristiche importanti sulle quali si vuole richiamare l'attenzione.
- Astrazione come strumento concettuale del computational thinking

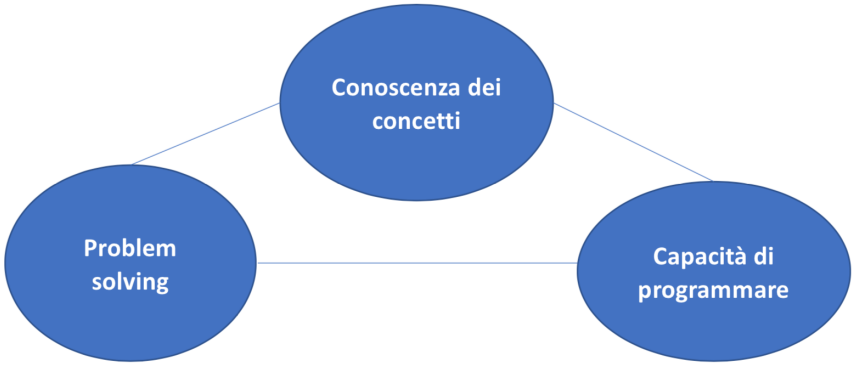
Pensiero Computazionale [competenza trasversale]

I metodi caratteristici che si acquisiscono con lo studio dell'informatica includono:

- analizzare e organizzare i dati del problema in base a criteri logici;
- rappresentare i dati del problema tramite opportune astrazioni;
- formulare il problema in un formato che ci permette di usare un esecutore (in senso ampio) per risolverlo;
- automatizzare la risoluzione del problema definendo una soluzione algoritmica, consistente in una sequenza accuratamente descritta di passi, ognuno dei quali appartenente a un catalogo ben definito di operazioni di base;
- identificare, analizzare, implementare e verificare le possibili soluzioni con una efficace ed efficiente combinazione di passi e risorse (avendo come obiettivo la ricerca della soluzione migliore secondo tali criteri);
- generalizzare il processo di risoluzione del problema per poterlo trasferire ad un ampio spettro di altri problemi.

Obiettivi (cont)

- Capire come formulare i problemi in modo da poterli risolvere in modo computazionale
- Essere capaci di scrivere programmi di piccola/media dimensione
- Studiare i modelli astratti di calcolo per capire come funzionano i computer e come sono in grado di risolvere i problemi



Informazioni utili

- Docenti: Prof. Chiara Bodei, Roberta Gori, Damiano Di Francesco Maesa
- Orario Lezioni: LUN 9-11 , MER 9-11
- Orario Laboratorio: (GIO 16-18)
- Ricevimento studenti Bodei: per ora su appuntamento
- E-mail: chiara.bodei@unipi.it, roberta.gori@unipi.it, damiano.difrancesco@unipi.it
- Canale Teams del corso: 017AA 23/24 - FONDAMENTI DI PROGRAMMAZIONE CON LABORATORIO

  Pagina web del corso:
  www.di.unipi.it/∼chiara/CORSO_FP_23/FP/index.html

  Testi consigliati per la consultazione:
  - J. Hopcroft-R. Motwani-J. Ullman. Automi, linguaggi e calcolabilità. Pearson.
  - B.W. Kerninghan, D.M. Ritchie. Linguaggio C. Pearson.
  - Ceri-Mandrioli-Sbattella. Informatica: programmazione. McGraw-Hill

Programma di massima del corso

- Concetti di base della programmazione
- La programmazione nel linguaggio C
- Cenni di teoria degli automi e dei linguaggi

Informatica: cosa è e cosa non è

- **Non** è la scienza e la tecnica dei calcolatori.
- **Non** è lo studio degli utilizzi e delle applicazioni dei calcolatori e del software.
- **Non** è lo studio di come scrivere i programmi per i calcolatori.
- È la scienza che studia i procedimenti di calcolo effettivi

È lo studio sistematico degli algoritmi che descrivono e trasformano l'informazione: la loro teoria, analisi, progetto, efficienza, realizzazione e applicazione
Association for Computing Machinery.

Algoritmi

- Utilizziamo algoritmi nella vita quotidiana tutte le volte che, ad es., seguiamo le istruzioni per il montaggio di una apparecchiatura, per impostare il ciclo di lavaggio di una lavastoviglie, per prelevare contante da uno sportello Bancomat, ecc.

> Un algoritmo è una sequenza di passi che, se intrapresa da un esecutore, permette di ottenere i risultati attesi a partire dai dati forniti.

- Una volta in grado di specificare un algoritmo per risolvere un problema, siamo anche in grado di automatizzare il procedimento descritto dall'algoritmo.

Il termine algoritmo deriva dal nome del matematico persiano Muhammad ibn Musa al-Khwarizmi (Corasmia 780 circa - 850 circa). Esercitò la professione nella città di Baghdad, dove insegnava, e introdusse nel mondo arabo i numeri indiani. La sua opera "Il calcolo degli indiani" venne successivamente tradotta in latino da un monaco europeo, con il titolo Liber algarismi - (Il libro di al-Khwarizmi).

Algoritmi

- Il concetto di algoritmo ha origini molto lontane: l'uomo ha utilizzato spesso algoritmi per risolvere problemi di varia natura.
- È una tecnica trasmissibile che consente di passare dal risolvere i problemi al far risolvere i problemi. Le capacità intellettive necessarie per risolvere un problema sono "codificate" nell'algoritmo.
- Solo in era moderna ci si è posti il problema di caratterizzare problemi e classi di problemi per i quali è possibile individuare una soluzione algoritmica e solo nel secolo scorso è stato dimostrato che esistono problemi per i quali non è possibile individuare una soluzione algoritmica

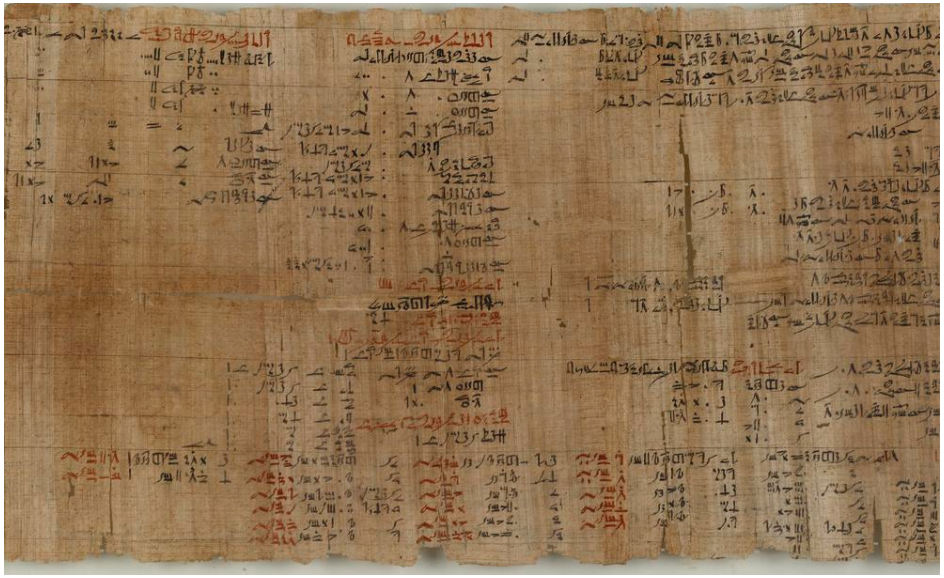
L'arte del problem solving

Un'idea geniale risolve spesso un grande problema, ma nella risoluzione di tutti i problemi interviene un pizzico di genialità.

Polya G., Come risolvere i problemi di matematica. Logica ed euristica nel metodo matematico. Feltrinelli, Milano, 1967

Il primo algoritmo documentato

Papiro di Ahmes o di Rhind
British Museum
ca 1650 a.C Scritto in ieratico



La moltiplicazione egizia

- Gli antichi non sapevano fare le operazioni in colonna (non c'era ancora la tecnologia giusta)
- Fare calcoli come le moltiplicazioni in generale era per loro complesso
- Fare calcoli più semplici come raddoppiamenti e dimezzamenti era invece più facile
- Pizzico di genialità: per calcolare A x B, raddoppio A e dimezzo B. Vado avanti semplificando così fino ad ottenere quanto mi serve.
- Uso la tecnica del *divide et impera*

Calcolare il MCD con l'algoritmo di Euclide

Per calcolare il Massimo Comun Divisore di x e y posso:

- Seguire la definizione: calcolare i divisori di x e di y, cercare quelli che sono divisori di entrambi e quindi identificare il massimo tra di loro; oppure
- sfruttare la proprietà che mi dice che:
  - MCD(x,x) = x
  - MCD(x,y) = MCD(x-y,y) se x >y
  - MCD(x,y) = MCD(x,y-x) se x >y

  e procedere con sottrazioni successive, riconducendo così un problema complesso a un problema più semplice e posso farlo più volte fino a trovare la soluzione

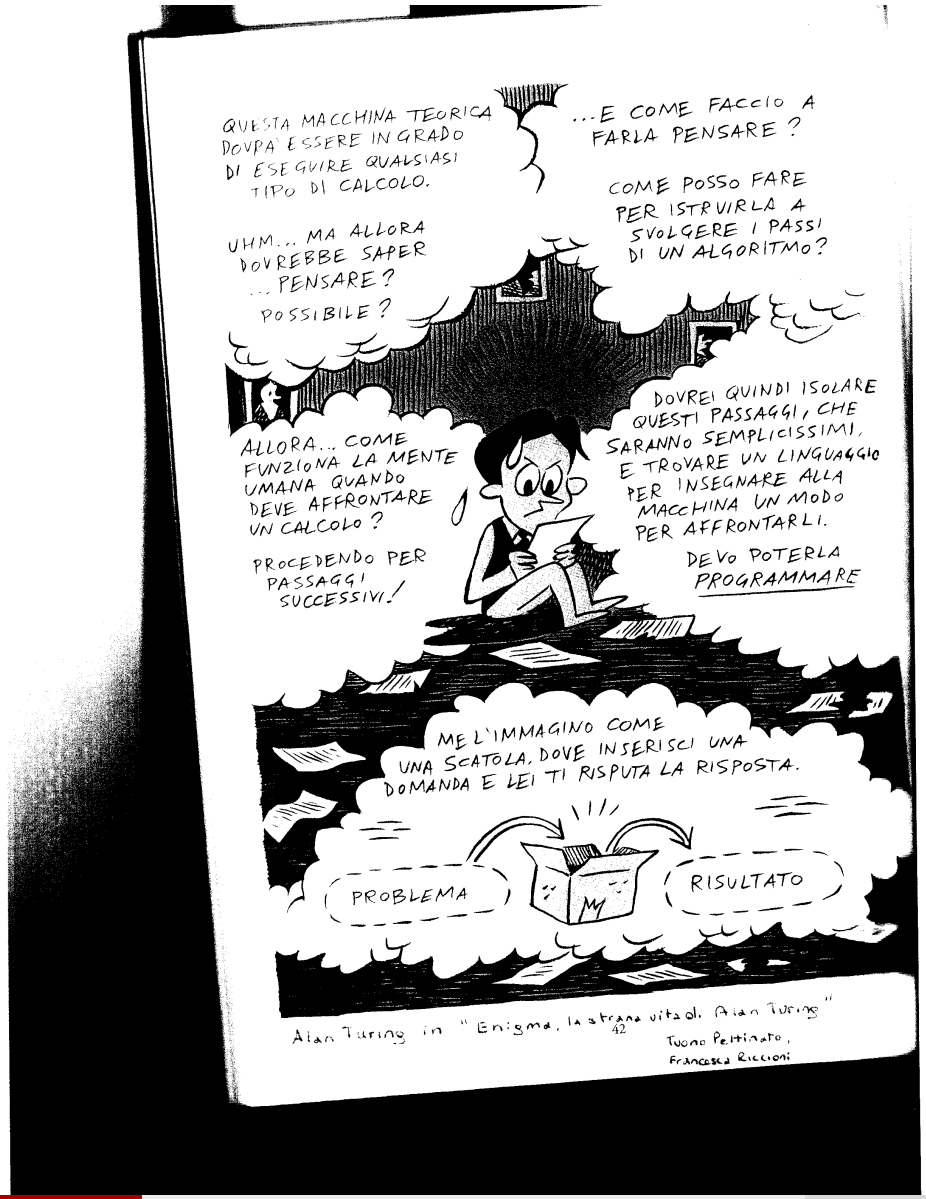
Astrazione di nuovo

- In informatica si separa il **cosa** (astrazione) dal **come** (implementazione), le proprietà funzionali da quelle strutturali
- Rispetto ai due esempi precedenti, possiamo separare il **cosa**, ovvero ottenere il risultato di una moltiplicazione in un caso e del Massimo Comun Divisore nell'altro dal **come**, ovvero farlo con un procedimento di calcolo oppure con un altro.

Domande fondamentali

- È sempre possibile trovare una soluzione algoritmica a un problema?
- Esiste un esecutore automatico in grado di eseguire un algoritmo e se sì come è fatto?

Il computer: una macchina per calcolare



Come nasce l'informatica? Facciamo un passo indietro

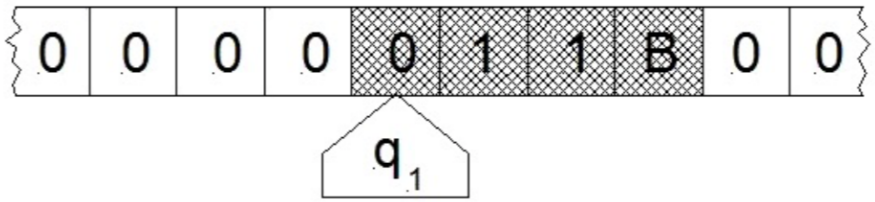
- È la scienza che studia i procedimenti di calcolo effettivi e nasce prima dei computer, studiando i processi di calcolo umano
- Turing vede il calcolo come manipolazione combinatoria di insiemi finiti e discreti di simboli secondo un insieme finito di regole

Domande fondamentali più dettagliate

- **Universalità: esiste una nozione astratta di macchina per calcolare?**
- Calcolabilità: esistono problemi che non possono essere risolti da nessuna macchina per il calcolo?
- Esistono dei limiti alla potenza espressiva delle macchine per calcolare?
- Esistono procedimenti più efficienti di altri?

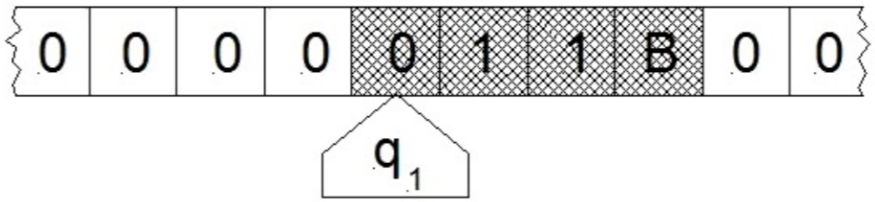
La Macchina di Turing

- È una macchina ideale (non fisica) di calcolo che manipola i dati contenuti su un nastro di lunghezza potenzialmente infinita, secondo un insieme di regole ben definite
- È il modello astratto di una macchina in grado di eseguire algoritmi



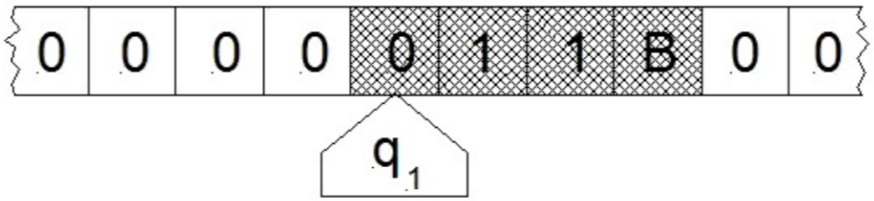
La Macchina di Turing (cont.)

- Se abbiamo una descrizione del funzionamento della macchina possiamo simularne il comportamento con carta e matita
- Questo processo è esso stesso un procedimento di calcolo
- Tra tutte le macchine di Turing ce ne è una che è in grado di simulare tutte le altre: è la Macchina Universale
- Con una descrizione e i dati la Macchina Universale eseguirà il calcolo per noi: è questo un modello teorico di computer



La Macchina di Turing (cont.)

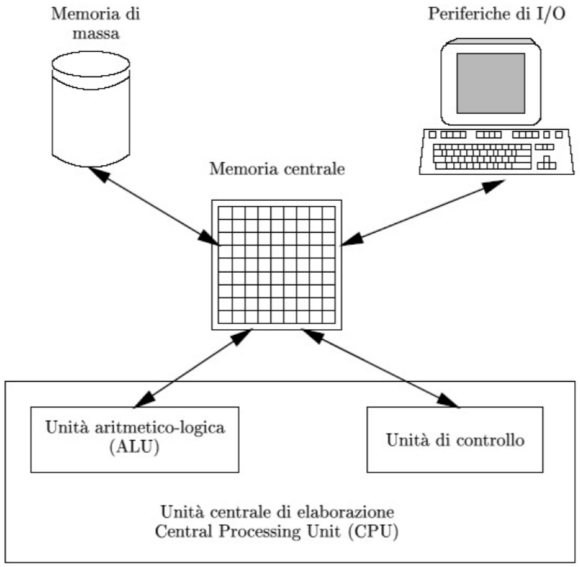
- Un problema è calcolabile se è calcolabile secondo Turing, ovvero se esiste una macchina di Turing in grado di risolverlo
- Le macchine di Turing possono eseguire tutte le computazioni che eseguono i moderni computer
- Modello universale di calcolo!



Il modello di Von Neumann

Modello concreto

- Equivalente alla Macchina di Turing
- Architettura della macchina
- Design alla base dei calcolatori attuali
- Esegue i calcoli predefiniti e quelli definiti dal programmatore
- La generalità è data dal programma memorizzato



Domande fondamentali più dettagliate

- Universalità: esiste una nozione astratta di macchina per calcolare?
- **Calcolabilità: esistono problemi che non possono essere risolti da nessuna macchina per il calcolo?**
- **Esistono dei limiti alla potenza espressiva delle macchine per calcolare?**
- Esistono procedimenti più efficienti di altri?

Calcolabilità

- Esistono problemi che non possono essere risolti in modo algoritmico
- Non è vero che possono risolvere tutti i nostri problemi

Domande fondamentali più dettagliate

- Universalità: esiste una nozione astratta di macchina per calcolare?
- Calcolabilità: esistono problemi che non possono essere risolti da nessuna macchina per il calcolo?
- Esistono dei limiti alla potenza espressiva delle macchine per calcolare?
- **Esistono procedimenti più efficienti di altri?**

Complessità

- Per uno stesso problema possono esistere soluzioni/algoritmi diversi
- È importante saperli confrontare e scegliere quelli più efficienti
- Ci sono problemi “facili”: sono quelli che richiedono un tempo ragionevole
- Ci sono problemi “difficili” che richiedono invece tempi così lunghi da essere praticamente irrisolvibili

Come si parla alla macchina?

- Il linguaggio (di programmazione) è lo strumento che permette di descrivere gli algoritmi in modo che il calcolatore possa comprenderli ed eseguirli
- Esiste una Babele di linguaggi di programmazione
- Il concetto di "cosa è calcolabile" non cambia: hanno la stessa potenza espressiva

Quando l'esecutore è il calcolatore

- I calcolatori sono macchine in grado di eseguire velocemente e con precisione sequenze di operazioni elementari.
- Un programma è la descrizione di un algoritmo, espressa in un linguaggio di programmazione che il calcolatore è in grado di comprendere ed eseguire.
- Il calcolatore riceve in ingresso un programma e un insieme di dati iniziali e produce in uscita i risultati dell'esecuzione del programma.
- A differenza di altre macchine automatiche (lavatrici, calcolatrici tascabili, ecc.) i calcolatori sono programmabili: la funzione svolta dipende dal particolare programma che indica alla macchina quali azioni compiere. La macchina non cambia al variare della funzione.

Linguaggi di programmazione

- Esistono moltissimi linguaggi di programmazione che possono essere classificati a seconda del paradigma di programmazione che seguono
- Un paradigma di programmazione indica lo stile e l'insieme di strumenti concettuali da seguire per scrivere programmi, riflettendo un un modo di pensare il processo di computazione.
- Adottando il C seguiremo il paradigma imperativo, per il quale un programma viene come una sequenza di comandi che agiscono sui dati o sull'ordine di esecuzione delle istruzioni [Altri esempi sono: Assembly, FORTRAN, COBOL, Pascal]
- Esistono altri paradigmi di programmazioni, tra i quali: funzionale [ML, Haskell, Scheme], orientato a oggetti [Java, Smalltalk, Eiffel], logico [PROLOG] e concorrente [CCS, CSP].

Cosa intendiamo per programmazione

- Il procedimento che porta alla definizione dei programmi adatti a risolvere problemi è detto programmazione.
- I concetti che stanno alla base della programmazione si possono spiegare e comprendere senza far riferimento al calcolatore.
- La programmazione è tuttavia divenuta una vera e propria disciplina solo con l'avvento dei moderni calcolatori elettronici.