



# Fondamenti d'Informatica: Introduzione

Barbara Re, Phd

# Che cosa si può calcolare?

---

## **Teoria della Computabilità - quale strano mostro! 😊**

Diverse sono le risposte possibili ...

- ▶ Classiche – basate su Macchine di Turing, grammatiche e funzioni ricorsive
- ▶ Moderne – legate all'evoluzione dei linguaggi di programmazione

# Che cosa si può calcolare?

---

**Il tema della computabilità** richiama l'idea di **computer e calcolatore**

Computare o calcolare significa far di conto con i numeri (es. presi gli interi  $0, +/- 1, +/- 2, \dots$  si vuole sommarli, moltiplicarli, calcolare limiti e derivate, ...)

Tutti i **problemi** che si possono formalizzare tramite **modelli matematici** possono essere **computati**

# Applicazioni computazionali

---

## René Descartes (Cartesio) – inizio del seicento

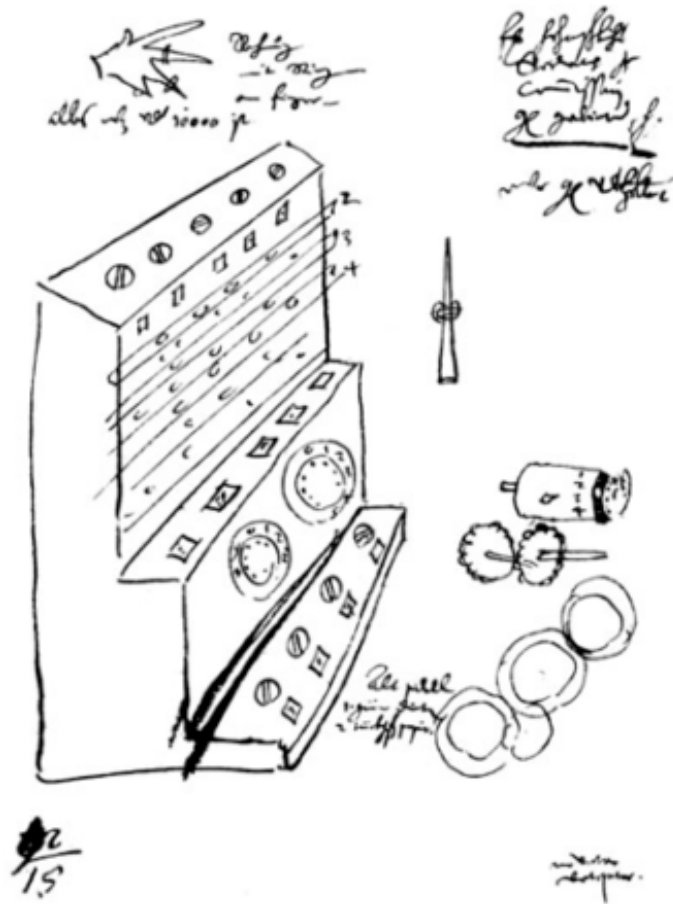
- ▶ **Vagheggiava** sulla possibilità di tradurre in termini matematici ogni problema e risolverlo tramite equazioni
- ▶ Applicava l'intuizione **al campo geometrico**

## Gottfried Wilhelm Leibniz – 1666

- ▶ Invitava a usare computazioni per risolvere oggettivamente i conflitti umani
- ▶ Mirava a trasferire le controversie dai ragionamenti complicati ai calcoli semplici

Alle applicazioni computazionali si accompagna un tentativo di costruire **macchine calcolatrici** capaci di aiutare a simulare la mente umana nel suo lavoro (tradizionali operazioni di somma, prodotto, sottrazione e divisione)

# Macchine calcolatrici: Wilhelm Schickard (1623)



La macchina di Schickard **sommava e sottraeva** numeri fino a **sei** cifre, e indicava il superamento della sua capacità facendo suonare una campanella

# Macchine Calcolatrici: Blaise Pascal (1643)

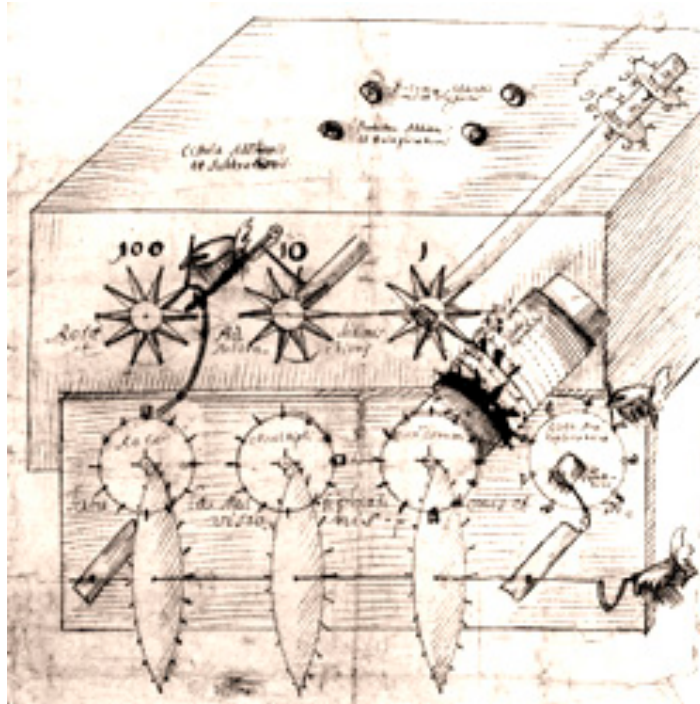
---



La macchina di Pascal consente di **addizionare e sottrarre** numeri fino a **otto** cifre, tenendo però conto del riporto

# Macchine Calcolatrici: Gottfried Wilhelm von Leibniz (1672)

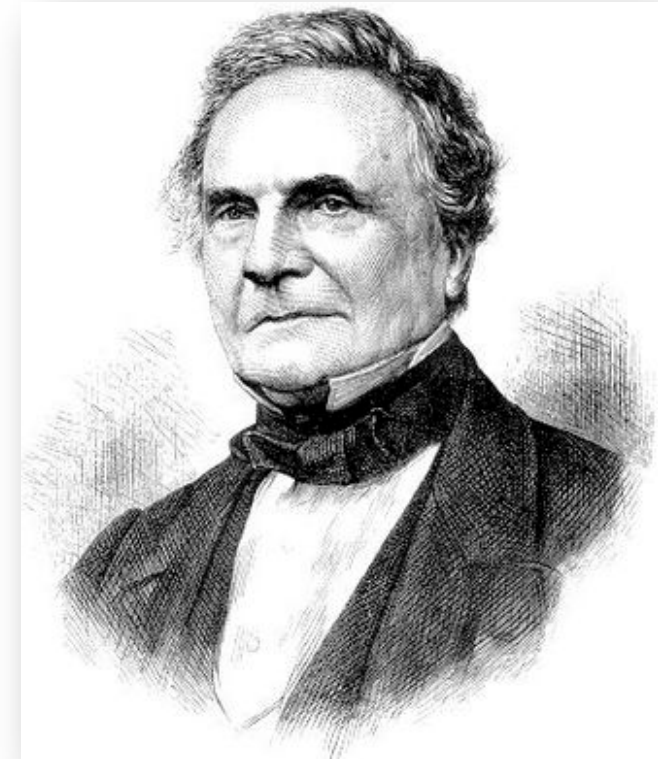
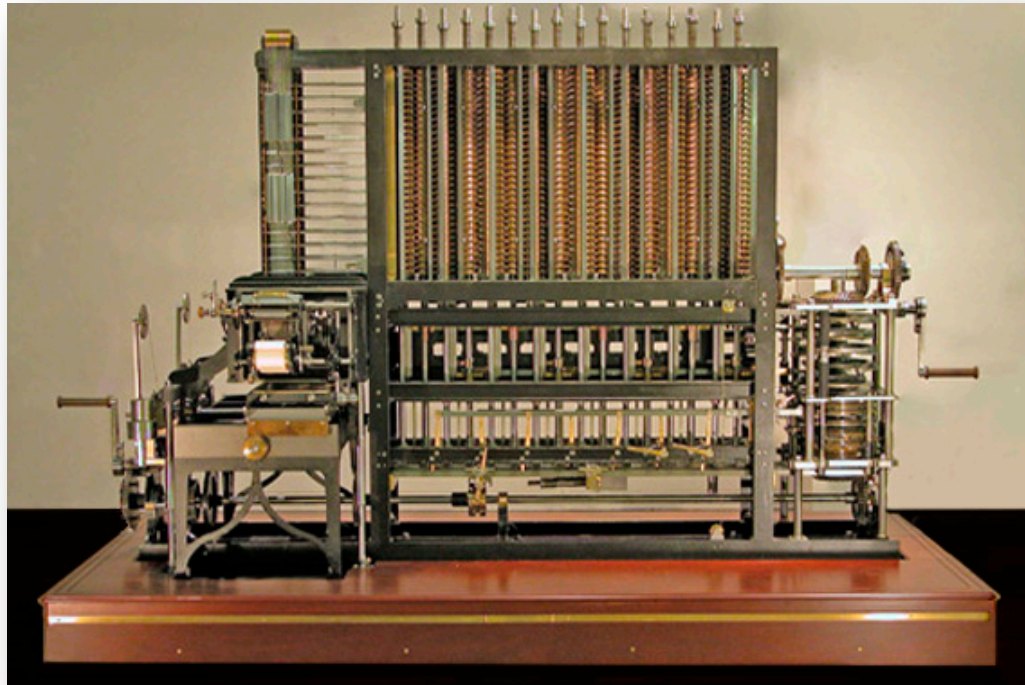
---



La macchina di Leibniz affrontò il problema della **moltiplicazione**

# Macchine calcolatrici: Charles Babbage (Ottocento)

---



Concepisce un meccanismo universale chiamato macchina analitica capace di adattarsi a vari contesti (numeri, mosse scacchi, ... )



<http://www.computerhistory.org/babbage/>



# Macchine calcolatrici: Ada Lovelace Byron (Ottocento)

---

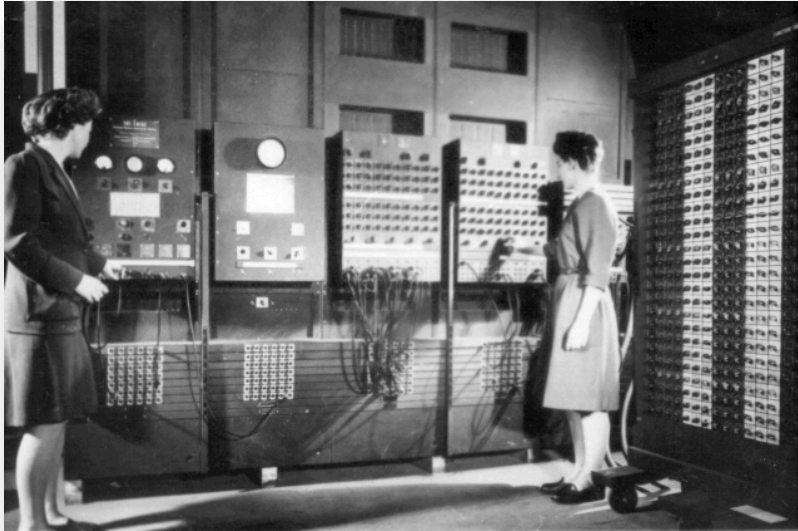
Incontra Charles Babbage e iniziò a studiare i **metodi di calcolo** realizzabili con la **macchina analitica**

Correla la macchina analitica di quello che ora chiameremo **software**

**E' considerata da molti come la prima programmatrice di computer di tutti i tempi**



# Primo computer elettronico (1946)



**ENIAC** (Electronic Numerical Integrator And Computer) fu progettato e costruito alla Moore School of Electrical Engineering (una ex scuola universitaria dell'Università della Pennsylvania) per il Ballistic Research Laboratory (un ex centro di ricerca dell'esercito degli Stati Uniti d'America) da Prosper Eckert e John Mauchly



**EDVAC** (Electronic Discrete Variables Automatic Computer) è la prima macchina digitale programmabile tramite un software basata su quella che sarà poi definita l'architettura di von Neumann

# Macchine, Linguaggi e Regole

---

La teoria della computabilità tratta della definizione formale del concetto di calcolo meccanico

- ▶ Le macchine calcolatrici per attuare le computazioni sono di fatto macchine “**calcolatrici meccaniche**”
- ▶ Si sviluppa sino alle moderne **soluzioni elettroniche**

Fondamentale è la definizione di linguaggi e regole con cui favorire la collaborazione delle macchine

- ▶ Identificare un **linguaggio astratto** appropriato, con **simboli** opportuni per cui formulare problemi da risolvere, per poterli poi proporre alla **macchina che li deve computare**
- ▶ Identificare le **leggi** che la macchina deve seguire per svolgere i suoi **calcoli**

# Cosa, Chi, e Come?

---

## Cosa è computabile?

- ▶ Tutti i problemi definibili per tramite di un alfabeto appropriato

## Chi computa?

- ▶ L'algoritmo che affronta il problema e la macchina che lo svolge

## Come si computa?

- ▶ Le regole che algoritmo e macchina calcolatrice devono rispettare nel loro lavoro

# L'alfabeto

---

- ▶ L'alfabeto è una sequenza di simboli finiti che descrivono i termini del problema
- ▶ Esempi di alfabeto: Lingua Italiana, Lingua inglese, polinomi e coefficienti interi, numeri reali, ...



Concordato **l'alfabeto** si può formulare il **problema** in modo adeguato e trasmetterlo **all'algoritmo** e alla **macchina calcolatrice** per la loro computazione

# L'algoritmo

- ▶ Algoritmo deriva dal nome del **matematico arabo Al Khuwarizmi**, vissuto nel IX secolo d.C.
- ▶ Un algoritmo fornisce la **soluzione** ad una classe di problemi
- ▶ Un algoritmo è una **successione di istruzioni** o passi che definiscono le operazioni da eseguire sui dati per **ottenere i risultati**
- ▶ Lo schema di esecuzione di un algoritmo specifica che **i passi devono essere eseguiti in sequenza**, salvo diversa indicazione
- ▶ Ogni algoritmo è concepito per interagire con l'ambiente esterno per **acquisire dati e comunicare messaggi o risultati**; i dati su cui opera un'istruzione sono forniti dall'esterno o sono frutto di istruzioni eseguite in precedenza



# Esempi di algoritmi

---

- ▶ Controllo dei voli aerei
- ▶ Regolazione reattori nucleari
- ▶ Reperimento d'informazioni da archivi
- ▶ Smistamento di comunicazioni telefoniche
- ▶ Gioco degli scacchi
- ▶ Controllo della produzione di una catena di montaggio
- ▶ Sistemi di puntamento di missili
- ▶ ...

# Correttezza e Efficienza

---

- ▶ Ovunque si impieghi un calcolatore elettronico occorrono algoritmi **corretti** e **efficienti** che ne utilizzino al massimo le possibilità
- ▶ Un algoritmo si dice **corretto** se, per ogni istanza di input, si ferma con l'output corretto
- ▶ Un algoritmo **corretto** risolve il problema computazionale dato
- ▶ Un algoritmo **efficiente** risolve il problema computazionale il più velocemente possibile e organizza i dati in modo da usare il minor spazio di memoria possibile



# Condizioni di un sistema di computazione

1. Un algoritmo è di lunghezza finita
2. Esiste un agente di calcolo (la macchina calcolatrice) che sviluppa la computazione eseguendo le istruzioni dell'algoritmo
3. L'agente di calcolo ha a disposizione una memoria, dove vengono immagazzinati i risultati intermedi del calcolo
4. Il calcolo avviene per passi discreti
5. Il calcolo non è probabilistico
6. Non deve esserci alcun limite finito alla lunghezza dei dati in ingresso
7. Non deve esserci alcun limite finito alla quantità di memoria disponibile
8. Deve esserci un limite finito alla complessità delle istruzioni eseguibili dal dispositivo
9. Il numero di passi della computazione è finito ma non limitato
10. Sono ammesse computazioni senza fine

# Condizioni di un sistema di computazione

---

1. Un algoritmo è di lunghezza finita
2. Esiste un agente di calcolo (la macchina calcolatrice) che sviluppa la computazione eseguendo le istruzioni dell'algoritmo
3. L'agente di calcolo ha a disposizione una memoria, dove vengono immagazzinati i risultati intermedi del calcolo
4. **Il calcolo avviene per passi discreti**
5. Il calcolo non è probabilistico
6. Non deve esserci alcun limite finito alla lunghezza
7. Non deve esserci alcun limite finito alla quantità di memoria disponibile
8. Deve esserci un limite finito alla complessità delle istruzioni eseguibili dal dispositivo
9. Il numero di passi della computazione è finito ma non limitato
10. Sono ammesse computazioni senza fine

Il calcolo non avviene attraverso dispositivi analogici, ma si svolge ordinatamente, un passo dietro l'altro

# Condizioni di un sistema di computazione

---

1. Un algoritmo è di lunghezza finita
2. Esiste un agente di calcolo (la macchina calcolatrice) che sviluppa la computazione eseguendo le istruzioni dell'algoritmo
3. L'agente di calcolo ha a disposizione una memoria, dove vengono immagazzinati i risultati intermedi del calcolo
4. Il calcolo avviene per passi discreti
5. **Il calcolo non è probabilistico**
6. Non deve esserci alcun limite finito alla lunghezza
7. Non deve esserci alcun limite finito alla quantità di memoria disponibile
8. Deve esserci un limite finito alla complessità delle istruzioni eseguibili dal dispositivo
9. Il numero di passi della computazione è finito ma non limitato
10. Sono ammesse computazioni senza fine

Il calcolo non è soggetto a fattori casuali, ma si svolge in modo assolutamente preciso e rigoroso (anche detto deterministico)

# Condizioni di un sistema di computazione

---

1. Un algoritmo è di lunghezza finita
2. Esiste un agente di calcolo (la macchina calcolatrice) che sviluppa la computazione eseguendo le istruzioni dell'algoritmo
3. L'agente di calcolo ha a disposizione una memoria, dove vengono immagazzinati i risultati intermedi del calcolo
4. Il calcolo avviene per passi discreti
5. Il calcolo non è probabilistico
6. **Non deve esserci alcun limite finito alla lunghezza dei dati in ingresso**
7. Non deve esserci alcun limite alla lunghezza dei dati in ingresso
8. Deve esserci un limite alla lunghezza dei dati in ingresso  
L'input del problema può essere arbitrariamente lungo (ad esempio, un algoritmo di somma tra gli interi si deve applicare ad ogni possibile addendo, quindi ad ogni numero intero, comunque grande)
9. Il numero di passi di calcolo è finito
10. Sono ammesse come operazioni solo quelle che possono essere eseguite da una macchina calcolatrice

## 7. Non deve esserci alcun limite finito alla quantità di memoria disponibile

---

- ▶ Si asserisce alla possibilità di accedere ad una memoria potenzialmente illimitata
- ▶ Sarà mica vero?
  - ▶ Proviamo a pensare che cosa accade in caso contrario, se ammettiamo di limitare preliminarmente ad un livello uniforme prefissato le potenzialità della memoria
  - ▶ In queste condizione può capitare che algoritmi anche elementari, costruiti per eseguire semplici computazioni si trovino nell'impossibilità di completarle
  - ▶ Ad esempio, la funzione che ad ogni intero associa il suo quadrato  $f(x) = x^2$  non è più calcolabile perché lo spazio di memoria necessario per computare il quadrato di  $x$  dipende ovviamente da  $x$  e dunque, per  $x$  molto grande, si trova ad infrangere i limiti eventualmente prefissi
- ▶ Tutto ciò per dire che l'assunzione fatta è del tutto plausibile!

## 8. Deve esserci un limite finito alla complessità delle istruzioni eseguibili dal dispositivo

---

- ▶ Si ribadisce quanto evidenziato dal punto I “Un algoritmo è di lunghezza finita” relativamente al numero delle istruzioni
- ▶ Si conferma che anche per la complessità delle singole istruzioni ci deve essere un limite
- ▶ Solo una porzione finita della memoria dell’agente di calcolo deve essere occupata da queste istruzioni iniziali al momento in cui la computazione su un dato input si avvia
- ▶ Non contraddice la condizione 7 -> che riguarda i successivi passi di calcolo e afferma la possibilità di accogliere senza limite le informazioni nella restante parte della memoria

*Anche la mente umana è intrinsecamente limitata ma ha delle potenzialità illimitate di crescita*

# Condizioni di un sistema di computazione

---

1. Un algoritmo è di lunghezza finita
2. Esiste un agente di calcolo (la macchina calcolatrice) che sviluppa la computazione eseguendo le istruzioni dell'algoritmo
3. L'agente di calcolo ha a disposizione una memoria, dove vengono immagazzinati i risultati intermedi del calcolo
4. Il calcolo avviene per passi
5. Il calcolo non è proiettivo  
Non è possibile stabilire a priori un limite massimo per il numero dei passi richiesti per eseguire un generico algoritmo su un certo input
6. Non deve esserci a priori un limite alla complessità delle istruzioni eseguibili dal dispositivo
7. Non deve esserci a priori un limite alla complessità delle istruzioni eseguibili dal dispositivo
8. Deve esserci un limite finito alla complessità delle istruzioni eseguibili dal dispositivo
9. **Il numero di passi della computazione è finito ma non limitato**
10. Sono ammesse computazioni senza fine

# Condizioni di un sistema di computazione

---

1. Un algoritmo è di lunghezza finita
2. Esiste un agente di calcolo (la macchina calcolatrice) che sviluppa la computazione eseguendo le istruzioni dell'algoritmo
3. L'agente di calcolo ha a disposizione una memoria, dove vengono immagazzinati i risultati intermedi del calcolo
4. Il calcolo avviene per passi discreti
5. Il calcolo non è probabilistico
6. Non deve esserci a priori una soluzione
7. Non deve esserci a priori una soluzione, in tal caso la computazione è destinata a prolungarsi indefinitivamente senza produrre risposte soddisfacenti
8. Deve esserci un limite al numero di passi del calcolo
9. Il numero di passi della computazione è finito ma non limitato
10. **Sono ammesse computazioni senza fine**





# Formalismi

---

- ▶ Alcuni importanti formalismi che hanno storicamente condizionato la teoria e l'applicazione dei calcolatori:
  - ▶ Le macchine di Turing
  - ▶ Gli schemi ricorsivi di Kleene e McCarthy
  - ▶ ... altri più moderni legati ai linguaggi di programmazione

# Risolvere un problema

---

- ▶ Uno dei principali risultati della teoria della computabilità è la completa identità delle classi di problemi risolubili nei diversi formalismi
  - ▶ Ci sono problemi che non possono avere risposta ☹️
  - ▶ Ci sono problemi che possono essere risolti 😊
  - ▶ Ci sono problemi che richiedono risorse così ingenti da scoraggiare ogni tentativo di soluzione pratica 😊 ☹️
- ▶ Molte proprietà relative, ad esempio, alle macchine di Turing si estendono immediatamente ad un qualsiasi linguaggio di programmazione

