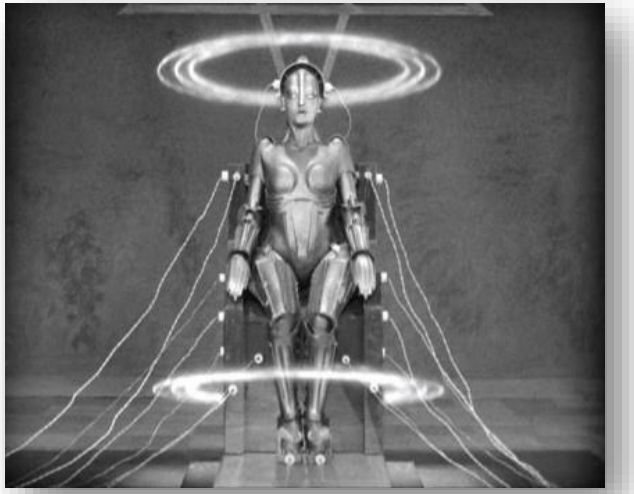
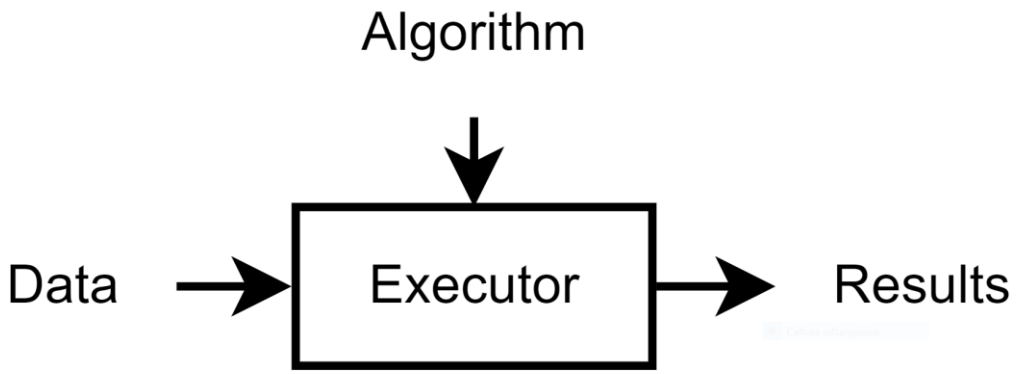
UNIVERSITÀ
DI PARMA

Informatica e Laboratorio di Programmazione

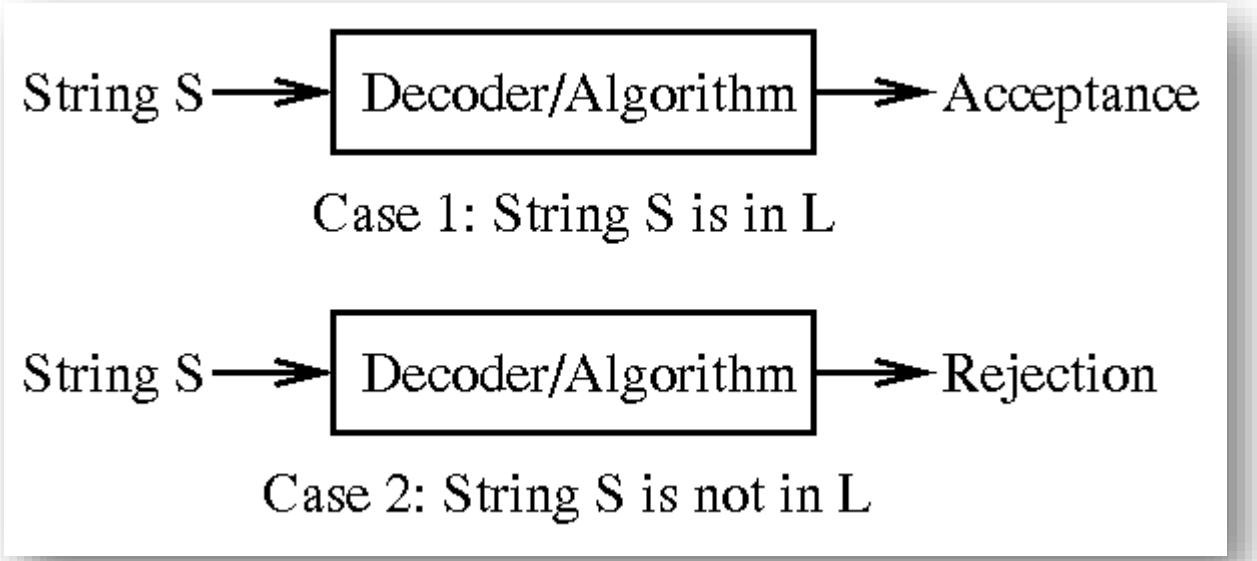
Automi



- ***automa***: macchina astratta
- realizza un certo algoritmo, secondo un modello di calcolo
- algoritmo definito nel *"linguaggio macchina"* dell'automa
- riceve ed elabora dei dati di ingresso



- ***riconoscimento*** di linguaggi
  - problema dell'appartenenza (*membership*)
  - data una stringa x, stabilire se essa appartiene ad L



- linguaggi di ***tipo 3***
  - riconosciuti da ***automi a stati finiti*** (**F**inite **S**tate **M**achine)
  - es.: $\{a^n b : n \geq 0\}$ generato da $S \rightarrow aS \mid b$
- linguaggi di ***tipo 2***
  - ric. da ***automi a pila non deterministici*** (**N**ondeterministic **P**ush**D**own **A**utomata)
  - es.: $\{a^n b^n : n \geq 1\}$ generato da $S \rightarrow aSb \mid ab$
- linguaggi di ***tipo 1***
  - riconosciuti da ***automi limitati linearmente*** (**L**inear **B**ounded **A**utomata)
  - Es.: $\{a^n b^n c^n : n \geq 1\}$
- linguaggi di ***tipo 0***
  - riconosciuti da ***macchine di Turing*** (**T**uring **M**achine)
  - $x \notin L$, semidecidibile: il processo può non terminare!

automi e linguaggi

MACCHINE (AUTOMI) A STATI FINITI

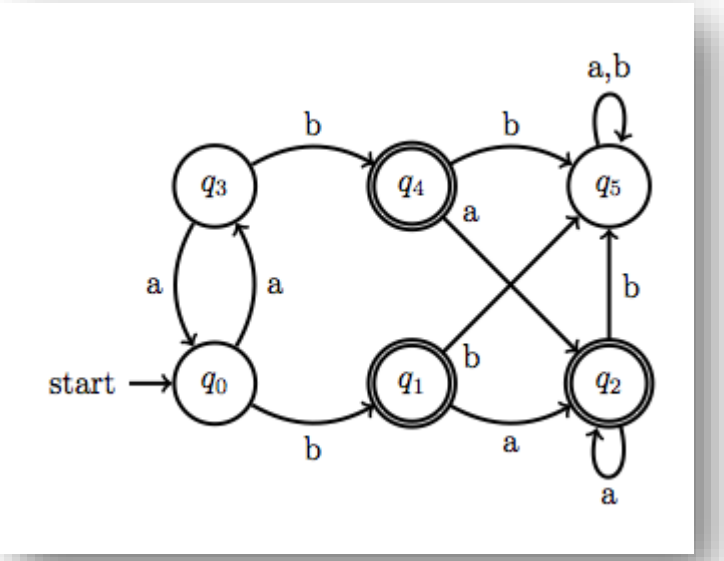
macchina a stati finiti

- $M = <\Sigma, Q, \delta, q_0, F>$
- $\Sigma = \{\sigma_1,...,\sigma_n\}$: ***alfabeto*** di input
- $Q = \{q_0,...,q_n\}$: insieme finito non vuoto di ***stati***
  - $F \subseteq Q$: insieme di stati ***finali***
  - $q_0 \in Q$: stato ***iniziale***
- $\delta: Q \times \Sigma \rightarrow Q$: ***funzione di transizione***
  - in base allo stato e al simbolo di input attuali …
  - determina lo stato successivo

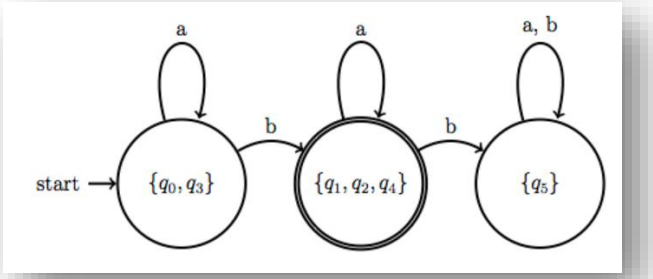
FSM riconoscono tutti e soli i linguaggi regolari

esempio

- M = <Σ, Q, δ, q_0, F>
- Σ = {a,b}: alfabeto di input
- Q = {q_1,q_2,q_3,q_4,q_5}: insieme degli stati
  - {q_1,q_2,q_4} ⊆ Q: insieme di stati finali
  - q_0 ∈ Q: stato iniziale
- δ: Q x Σ → Q: funzione di transizione
  - in base allo stato e al simbolo di input attuali
  - determina lo stato successivo

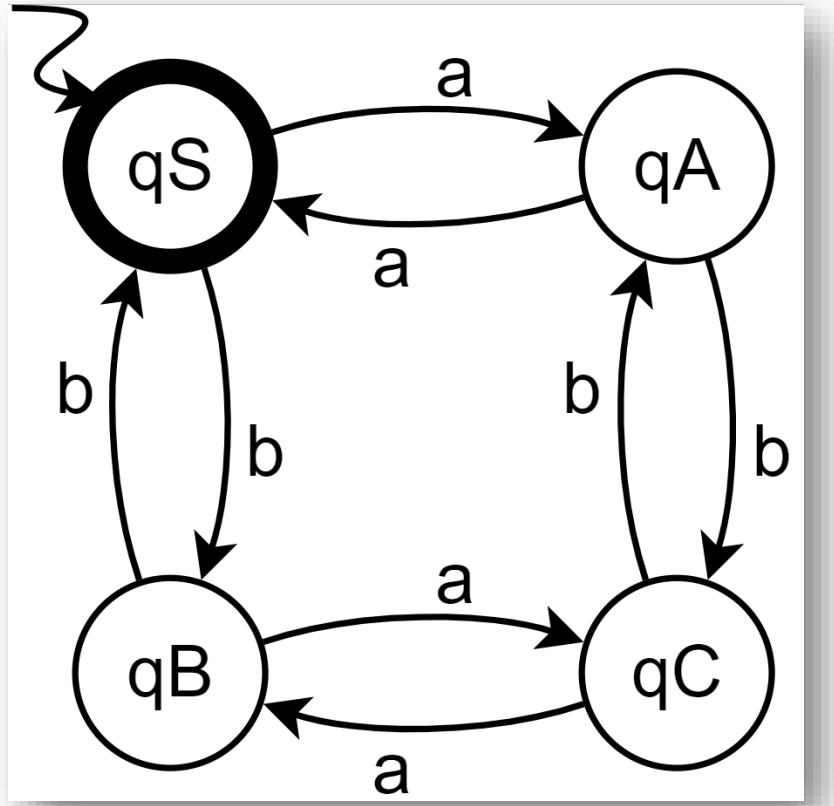


δ	a	b
→ q_0	q_3	q_1
*q_1	q_2	q_5
*q_2	q_2	q_5
q_3	q_0	q_4
*q_4	q_2	q_5
q_5	q_5	q_5



M = <{a, b}, {qS, qA, qB, qC}, δ, qS, {qS}>

δ	a	b
q_S	q_A	q_B
q_A	q_S	q_C
q_B	q_C	q_S
q_C	q_B	q_A



grammatica equivalente:
S → aA | bB | ε
A → aS | bC
B → aC | bS
C → aB | bA

macchina a stati finiti non deterministica

- **N**ondeterministic **F**inite **A**utomaton
- $M = <\Sigma, Q, \delta_N, q_0, F>$
- $\Sigma = \{\sigma_1, ..., \sigma_n\}$: alfabeto di input
- $Q = \{q_0, ..., q_m\}$: insieme finito non vuoto di stati
- $F \subseteq Q$: insieme di stati finali
- $q_0 \in Q$: stato iniziale
- $\delta_N: Q \times \Sigma \rightarrow P(Q)$: funzione di transizione
  - determina ***insieme di stati successivi***
  - $P(Q)$ è l'insieme delle parti di Q, ossia l'insieme di tutti i possibili sottoinsiemi di Q

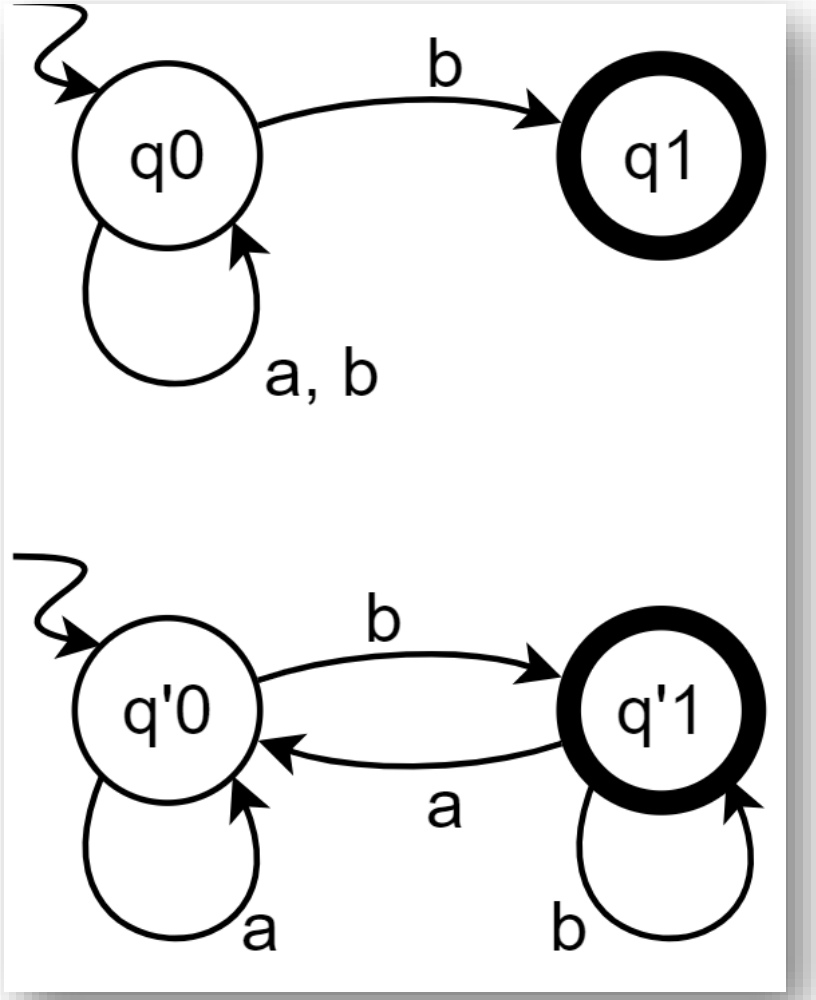
$M = <\{a, b\}, \{q_0, q_1\}, \delta, q_0, \{q_1\}>$

δ	a	b
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{\}$	$\{\}$

$M' = <\{a, b\}, \{q'_0, q'_1\}, \delta', q'_0, \{q'_1\}>$

δ'	a	b
q'_0	q'_0	q'_1
q'_1	q'_0	q'_1

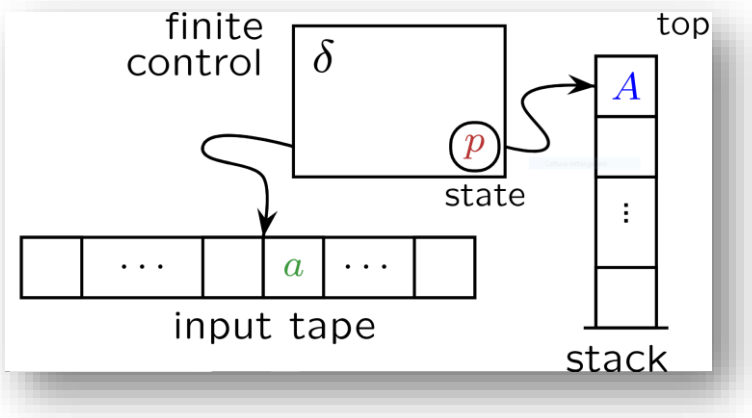
Per ogni automa a stati finiti non deterministico è possibile costruire un automa a stati finiti deterministico in grado di riconoscere lo stesso linguaggio



automi e linguaggi

automi a pila

automi a pila (PDA)



- **Push**down **A**utomata
- simili a FSM
  - dotato di ***memoria infinita***, organizzata a ***pila***
  - si può accedere solo alla ***cima*** della pila
  - ***lettura*** del simbolo in cima
  - ***sostituzione*** simbolo in cima con nuova stringa (anche ε)
- in forma non-deterministica, permette di riconoscere i *linguaggi non contestuali*
- in forma deterministica, riconosce solo i *linguaggi non contestuali deterministici* (sottoclasse)
  - base dei comuni linguaggi di programmazione

definizione PDA

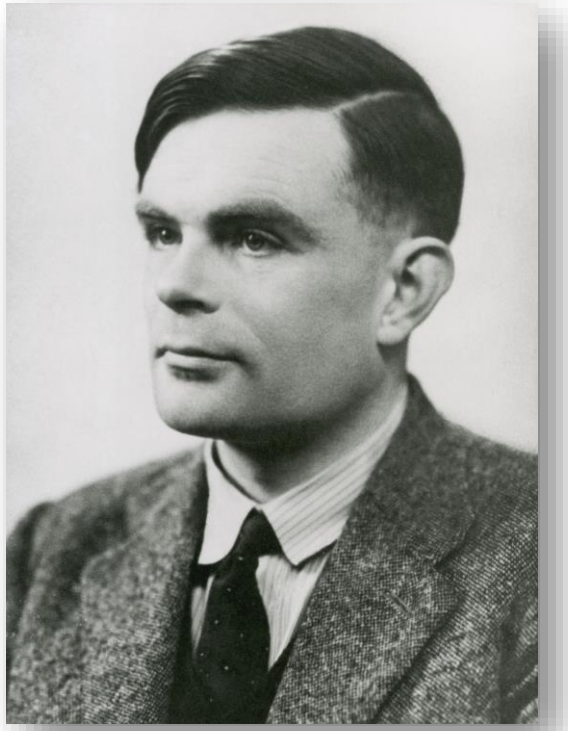
- **$M = <\Sigma, \Gamma, z_0, Q, q_0, F, \delta>$**
- $\Sigma = \{\sigma_1, ..., \sigma_n\}$: ***alfabeto*** di input
- $\Gamma = \{z_0, ..., z_m\}$: ***simboli*** della ***pila***
- $z_0 \in \Gamma$: *simbolo* di *pila* ***iniziale***
- $Q = \{q_0, ..., q_k\}$: insieme finito non vuoto di ***stati***
- $q_0 \in Q$: stato ***iniziale***; $F \subseteq Q$: insieme di ***stati finali***
- $\delta: Q \times \Sigma \times \Gamma \rightarrow Q \times \Gamma^*$: ***funzione di transizione***
  - in base a ***stato***, simbolo di ***input***, simbolo in cima a ***pila*** …
  - determina ***stato successivo*** e ***simboli*** inseriti nella ***pila***
  - per rimuovere il simbolo in cima alla pila, si scrive ε

esempio PDA

- automa a pila M che riconosce $L = \{a^n b^n, n \geq 1\}$
- $M = <\{a, b\}, \{Z_0, A_0, A\}, Z_0, \{q_0, q_1, q_2\}, q_0, \{q_2\}, \delta>$
  - A: si impilano sopra A_0 per contare le a
  - q_0: si memorizzano le a
  - q_1: si confrontano le b con quanto memorizzato
  - q_2: stato finale

riconoscitore $a^n b^n$

δ	Z_0, a	Z_0, b	A_0, a	A_0, b	A, a	A, b
q_0	A_0, q_0		AA_0, q_0	ε, q_2	AA, q_0	ε, q_1
q_1				ε, q_2		ε, q_1
q_2						



automi e linguaggi

macchina di Turing

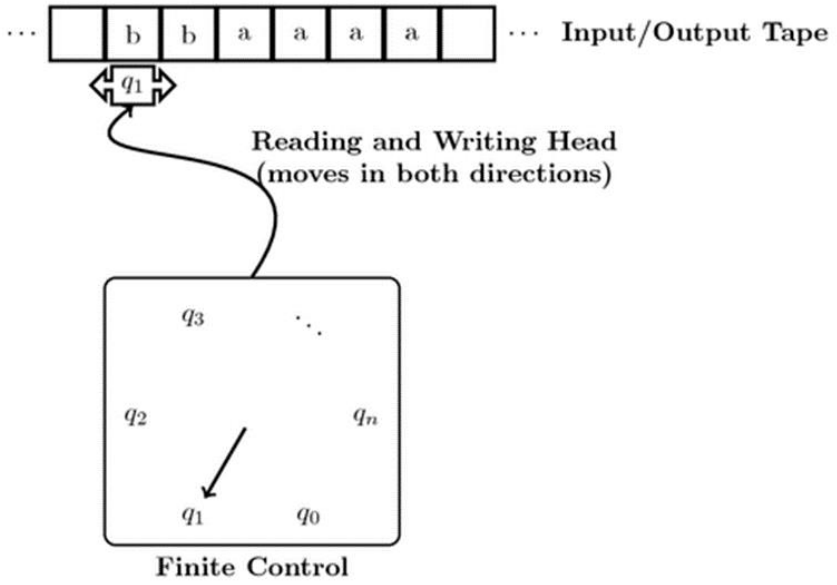
A. Turing (1937). On Computable Numbers, with an Application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, s2-42 (1): 230–265.

macchina di Turing

- ***modello di calcolo*** proposto dal logico matematico inglese ***Alan Turing*** in un suo famoso articolo del 1937
- macchina ***ideale*** (modello ***astratto***) in grado di leggere e scrivere dati contenuti su un nastro di lunghezza potenzialmente infinita, secondo un insieme prefissato di regole
- ha potere computazionale ***massimo***
  - è in grado di effettuare tutte le elaborazioni effettuabili dagli altri modelli di calcolo
- per ogni problema calcolabile esiste una MdT in grado di risolverlo
  - per ogni funzione calcolabile esista una macchina di Turing equivalente

macchina di Turing

- automa con testina di scrittura/lettura su nastro bidirezionale "*illimitato*"
- ad ogni passo:
  - si trova in un certo stato
  - legge un simbolo dal nastro
  - in base alla funzione di transizione di transizione (deterministica):
    - scrive un simbolo sul nastro
    - sposta la testina di una posizione
    - cambia lo stato



https://www.google.com/doodles/alan-turings-100th-birthday

Turing machine deterministica

- $M = <\Sigma, Q, q_0, F, \delta>$
- $\Sigma = \{\sigma_1, ..., \sigma_n, \text{b}\}$: ***alfabeto*** del nastro (con blank)
- $Q = \{q_1, ..., q_m\}$: insieme finito di ***stati***
- $q_0 \in Q$: stato ***iniziale***
- $F \subseteq Q$: insieme di stati ***finali***
- $\delta: Q \times \Sigma \rightarrow Q \times \Sigma \times \{L, R, N\}$: ***funzione di transizione***
  - determina la ***configurazione successiva***:
    - ***stato***
    - ***simbolo*** scritto su ***nastro***
    - ***spostamento*** della testina (left, right, null)

funzionamento

- ***inizialmente*** il nastro contiene la stringa di ***input*** preceduta e seguita da una serie infinita di simboli vuoti
- la ***testina*** è posizionata sul ***primo*** simbolo della stringa di input e la macchina si trova in uno stato speciale (stato ***iniziale***)
- sulla base dello stato in cui si trova e del simbolo la macchina applica la relativa ***funzione di transizione***
  - modificare il ***simbolo*** sotto la testina
  - cambia lo ***stato*** interno
  - ***sposta*** la testina a destra o a sinistra
- la macchina prosegue nell'esecuzione fino a quando non viene a trovarsi in uno stato ***finale***, oppure non è presente nessuna funzione di transizione dato la situazione attuale
  - se la macchina si ***arresta*** in uno stato finale il risultato è dato dalla stringa presente sul nastro

macchina di Turing universale

- è possibile definire una macchina di Turing capace di ***simulare*** il comportamento di ogni macchina di Turing
- la ***rappresentazione*** degli stati e della funzione di transizione sono presenti, oltre alla stringa di input, sul ***nastro***
- per ogni problema calcolabile esiste una macchina di Turing quindi …
  - la ***macchina di Turing universale*** è in grado di risolvere ogni problema calcolabile
- un modello di calcolo che ha lo stesso potere computazionale di una macchina di Turing si dice ***Turing completo***
- ***un linguaggio di programmazione è considerato a tutti gli effetti tale se è Turing completo***