Gocce di Java

Calcolatori e programmi

(Modulo 1)

Paolo Lollini

DIMAI – Università degli Studi di Firenze

basato sulle slide del Prof. Crescenzi

- Introduzione al metodo informatico
- Rassegna hardware e software di un calcolatore
- Introduzione al concetto di algoritmo
- Dare una visione dei linguaggi di programmazione...
- ...e di come un programma viene tradotto in un linguaggio comprensibile al calcolatore

Buona parte di questi argomenti si applica alla programmazione in generale (non solo a Java).

Cosa è l'informatica? Più facile dire cosa non è

- Poco a vedere con "alfabetizzazione informatica" (saper usare un computer per scrivere un testo o navigare in Internet)
- Non consiste semplicemente nello scrivere programmi
  - anche se è naturale aspettarsi da un informatico la capacità di farlo in modo corretto ed efficace

Cosa è l'informatica?

- Denning et al (1989), ACM (Association of Computing Machinery)
  - *L'informatica è lo studio sistematico dei processi* ***algoritmici*** *che descrivono e trasformano l'informazione: la loro teoria, analisi, progettazione, efficienza, implementazione e applicazione*

Cosa è l'informatica?

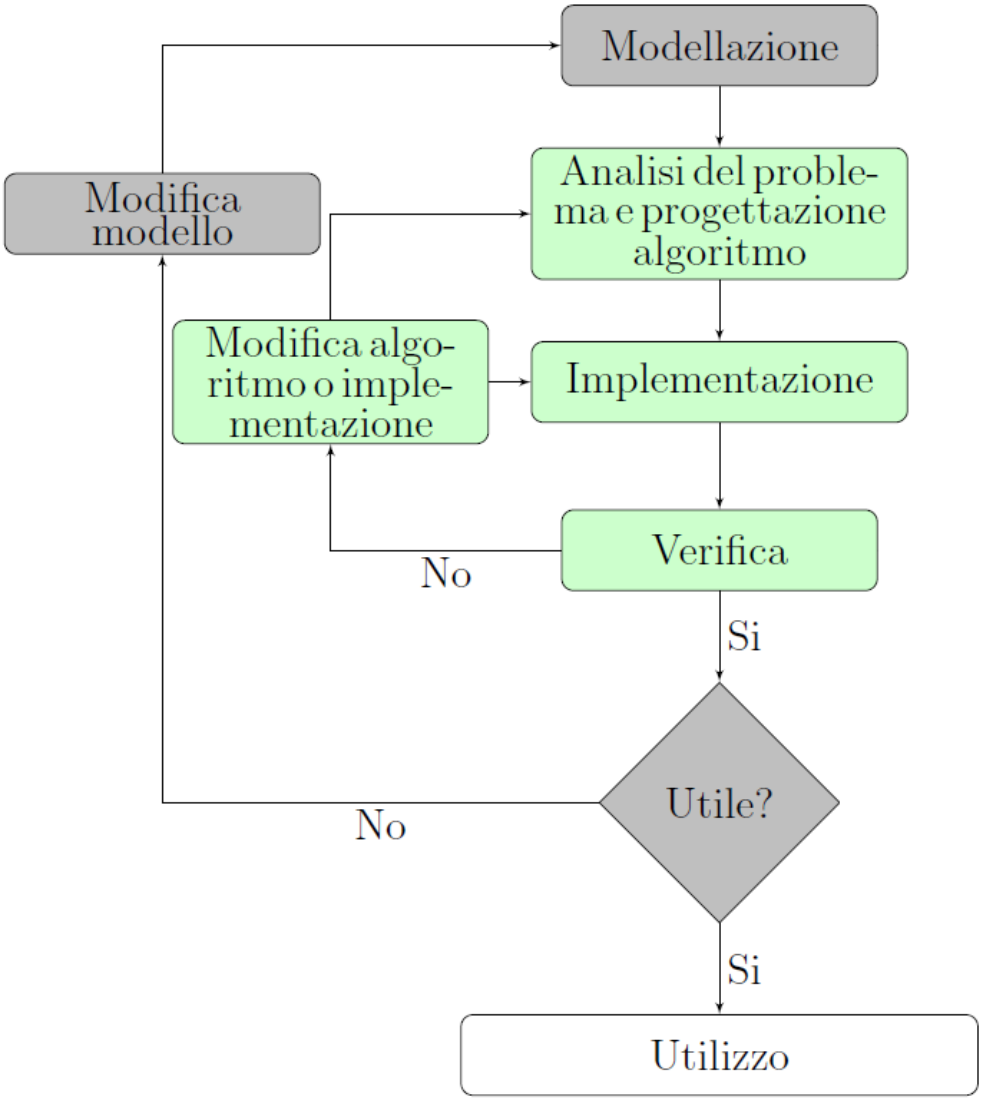
- Concetto di *Algoritmo* definito informalmente come
  - una sequenza precisa di operazioni
  - comprensibili ed eseguibili da uno strumento automatico
- Quindi nel campo dell'informatica la soluzione di uno specifico problema consiste
  - anzitutto nel proporre per il problema stesso un algoritmo risolutivo
  - e successivamente nel codificare l'algoritmo proposto in un programma che possa essere eseguito da un calcolatore

In questo corso vedremo principalmente la fase di trasformazione di un algoritmo in un programma.

Cosa è l'informatica?

- Metodo algoritmico (o informatico)
  - Formulare algoritmi che risolvano un problema
  - Trasformare questi algoritmi in programmi
  - Verificare la correttezza e l'efficacia di tali programmi analizzandoli ed eseguendoli

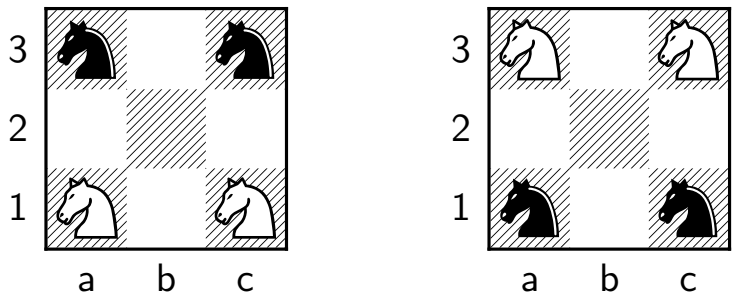
Il metodo algoritmico



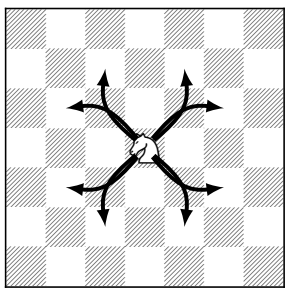
Il metodo algoritmico: trovare il giusto modello

Il rompicapo di Guarini

- Qual è la sequenza di mosse più breve che consente ai cavalli di passare dalla configurazione a sinistra a quella a destra?
- senza mai posizionare due cavalli sulla stessa casella?



- Possibili mosse del cavallo



Una prima soluzione...

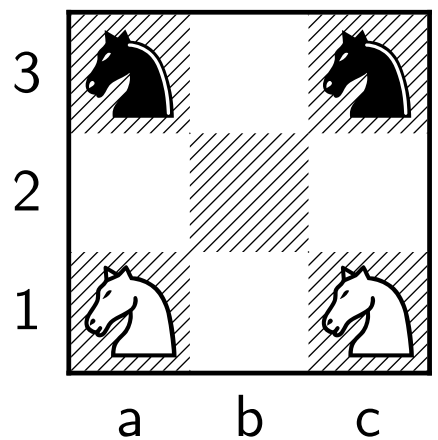
- Provare tutte le possibili sequenze di mosse dei quattro cavalli
- e selezionare quella più breve che soddisfi i requisiti del rompicapo

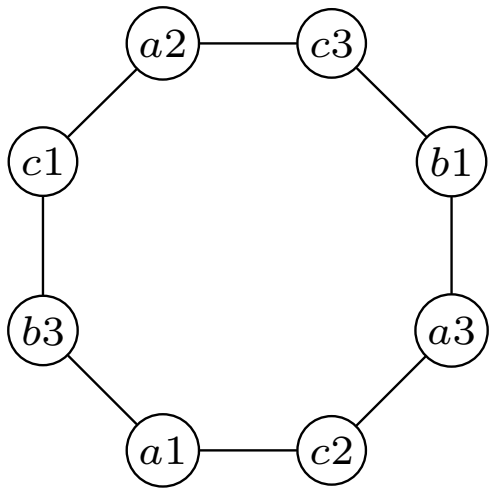
Il numero di possibili sequenze è molto elevato, rendendo tale soluzione del tutto inutilizzabile dal punto di vista pratico.

Il modello

La soluzione può essere ottenuta agevolmente se il problema viene posto in termini diversi ma equivalenti:

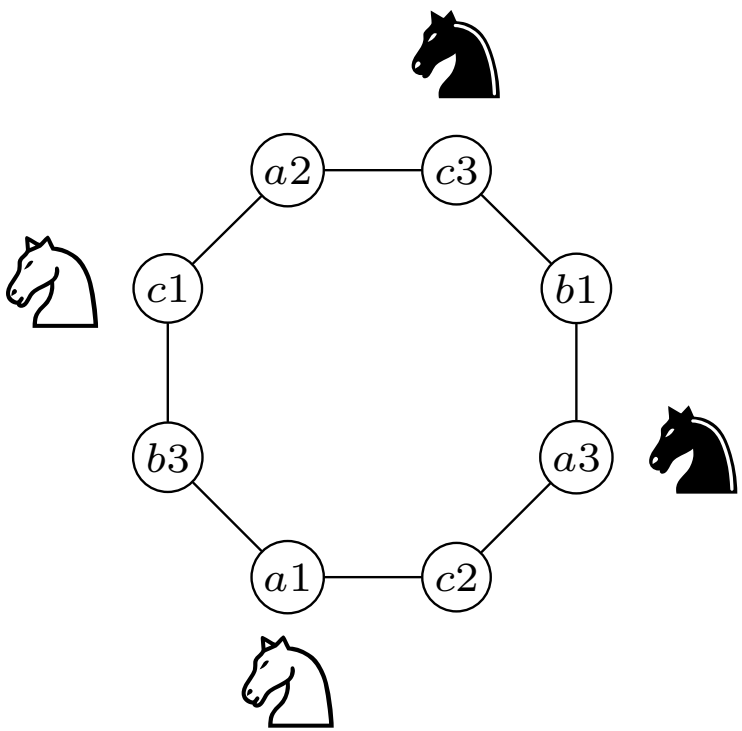
- Da una casella della scacchiera un cavallo può raggiungere solo due caselle
- Questa relazione fra caselle può essere descritta graficamente
- Rappresentare il problema mediante una relazione di raggiungibilità

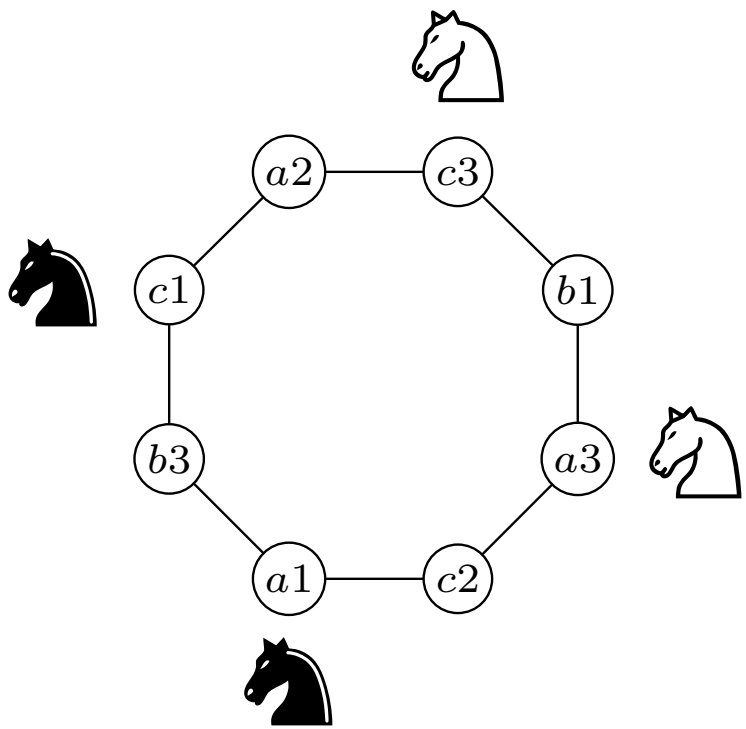




Il modello

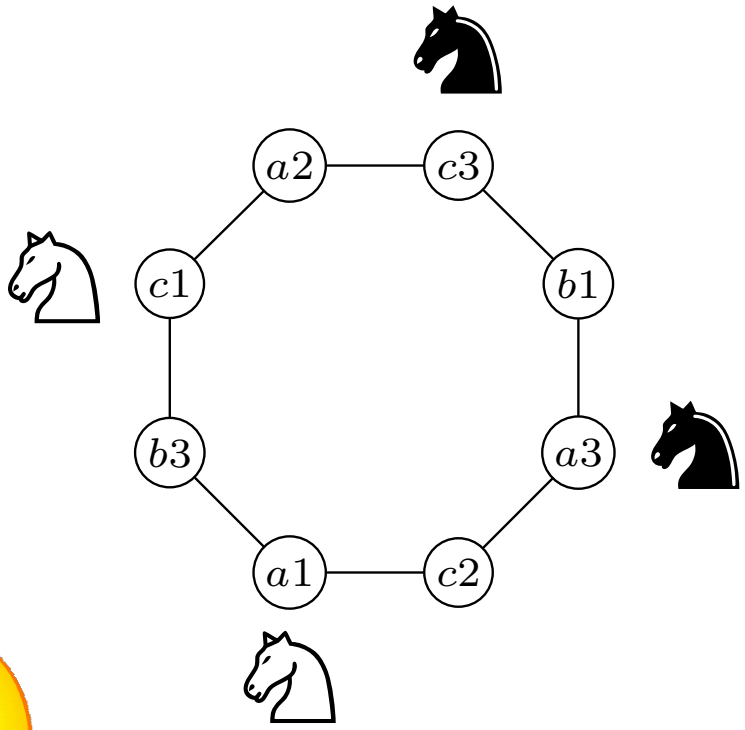
- Rappresentare il problema mediante una relazione di raggiungibilità

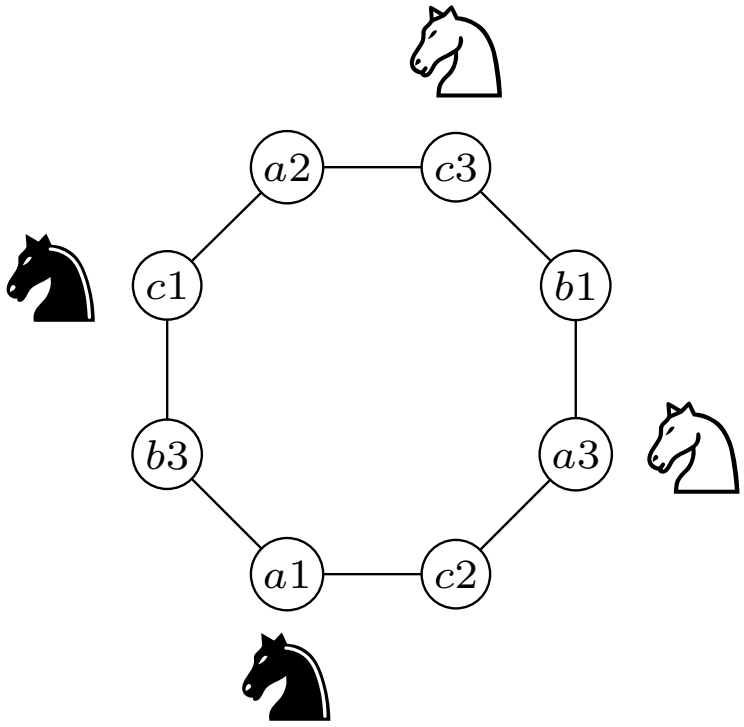




La soluzione: algoritmo

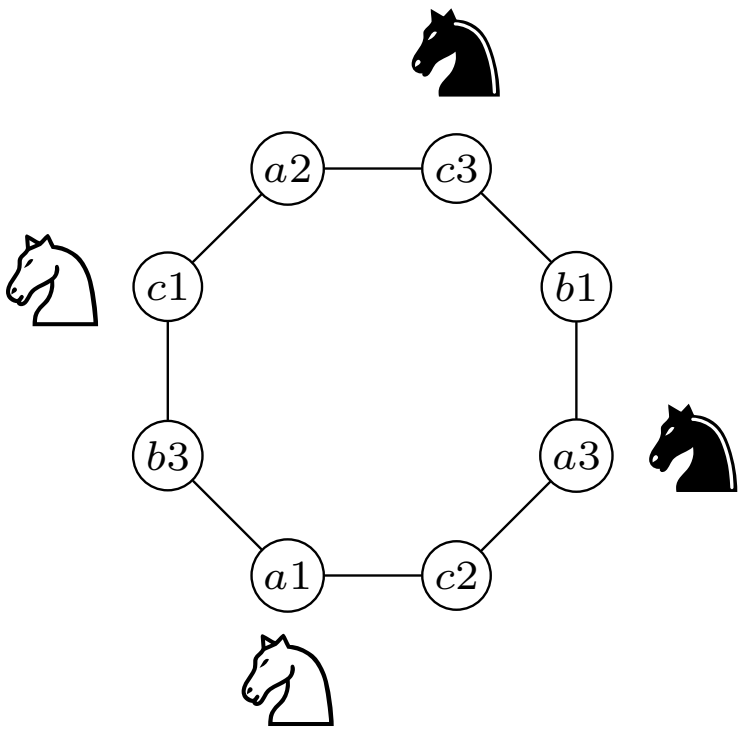
- Trovare il minimo numero di mosse per andare dalla configurazione a sinistra a quella a destra

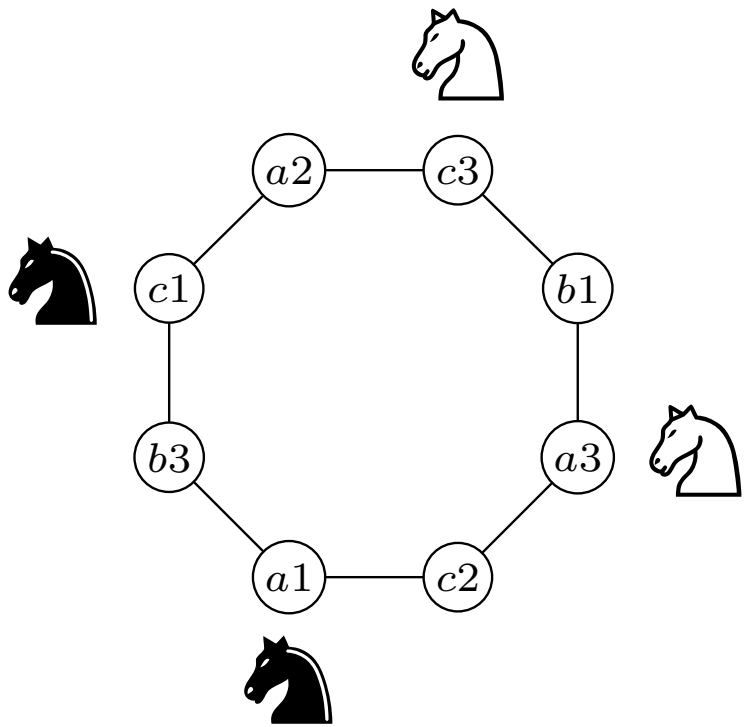




La soluzione: algoritmo

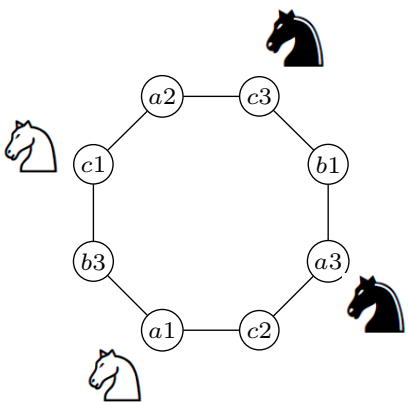
- Trovare il minimo numero di mosse per andare dalla configurazione a sinistra a quella a destra

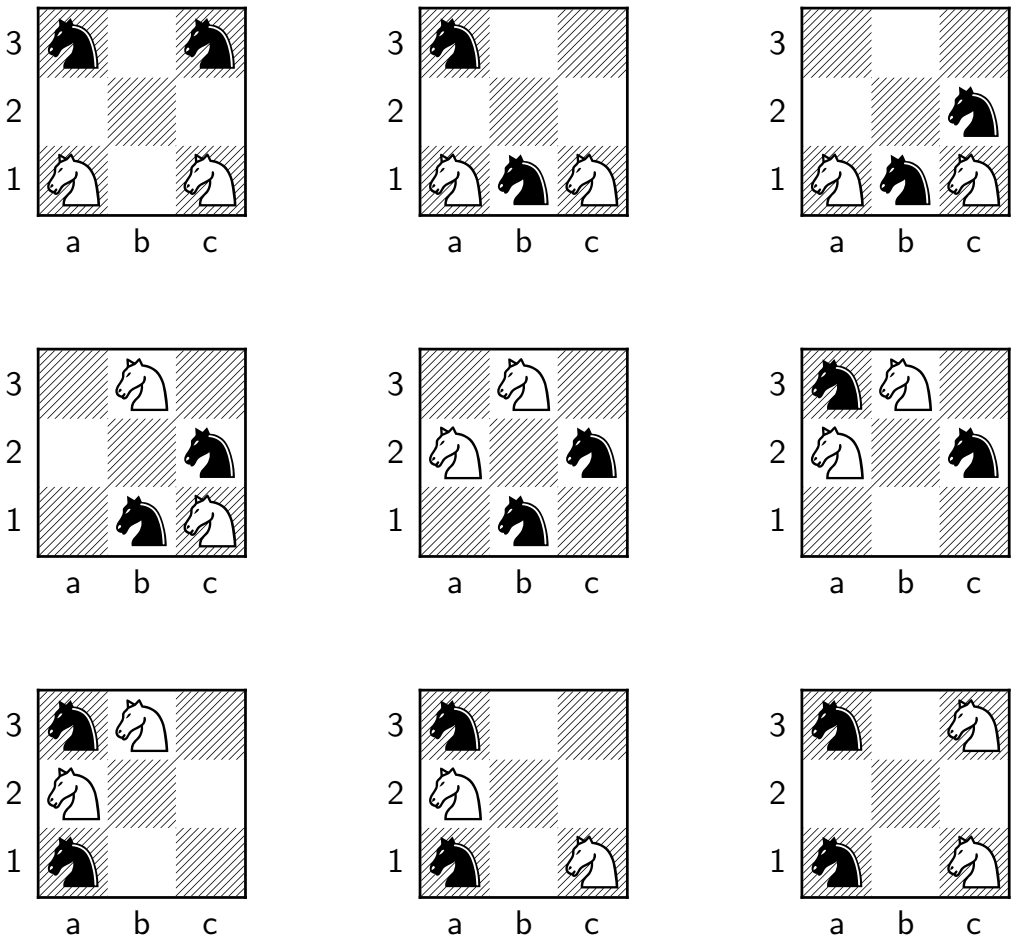




- Ruotare i cavalli di quattro posizioni in senso orario (o antiorario). Totale: 16 mosse.

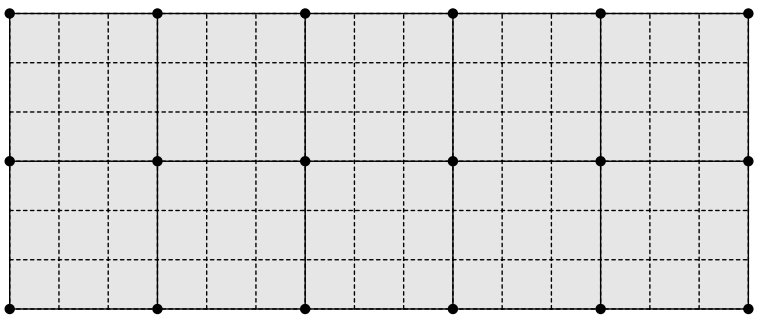
La soluzione: le prime 8 mosse





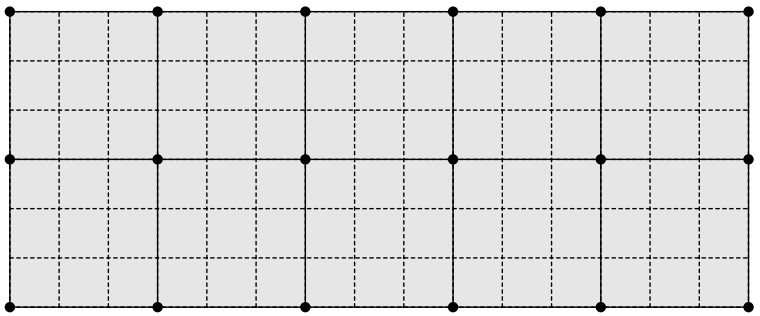
Il metodo algoritmico: trovare il giusto algoritmo

- Problema
  - Piastrellare una stanza rettangolare di dimensione $n \times m$ con il minor numero possibile di mattonelle quadrate di uguale dimensione
    - Esempio: $n = 6$ e $m = 15$



Il metodo algoritmico: trovare il giusto algoritmo

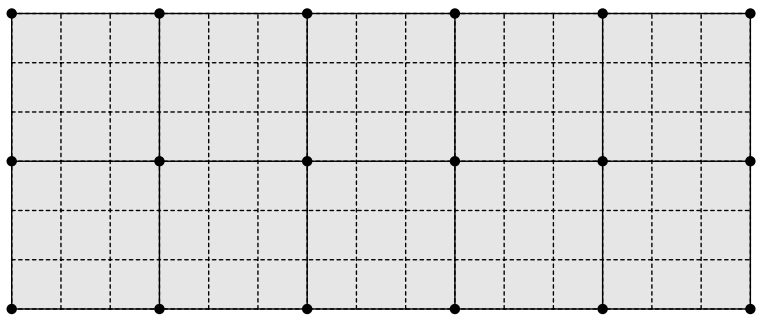
- Problema
  - Piastrellare una stanza rettangolare di dimensione $n \times m$ con il minor numero possibile di mattonelle quadrate di uguale dimensione
    - Esempio: $n = 6$ e $m = 15$



- Modello

Il metodo algoritmico: trovare il giusto algoritmo

- Problema
  - Piastrellare una stanza rettangolare di dimensione $n \times m$ con il minor numero possibile di mattonelle quadrate di uguale dimensione
    - Esempio: $n = 6$ e $m = 15$

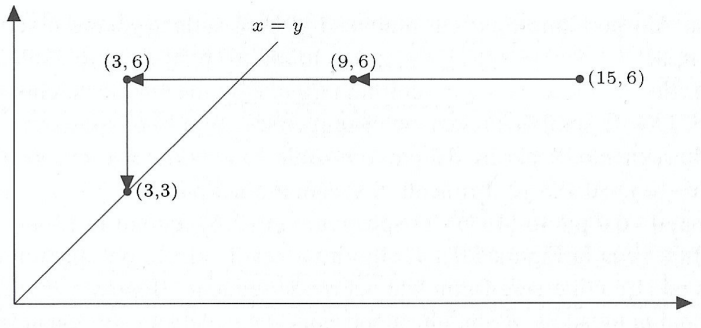


- Modello
  - Determinare il massimo numero intero che divide sia n che m
  - Calcolare il *massimo comun divisore* (MCD) di n e m
    - $MCD(6, 15) = 3$

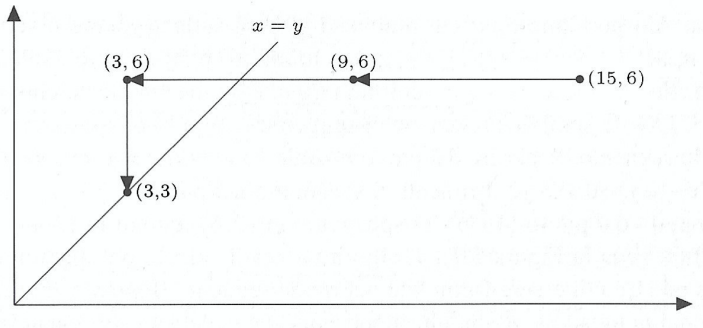
- Primo algoritmo basato su definizione
  - Supponiamo che $n < m$ e che n non divide m
  - Esaminiamo tutti i numeri d tra $n/2$ e 2 (in ordine inverso)
    - Se d divide n e divide m, allora $MCD(n, m) = d$
  - Se non troviamo nessun d con tale proprietà allora $MCD(n, m) = 1$
- Esempio: $n = 111$ e $m = 259$
  - $n/2 = 55$
  - Tutti i numeri tra 55 e 37 non dividono 111
  - 37 divide $111 = 37 \times 3$ e divide $259 = 37 \times 7$
  - $MCD(111, 259) = 37$

- Caso pessimo $MCD(n, m) = 1$
- Bisogna provare tutti i numeri tra $n/2$ e 2...
- La quantità di questi numeri può essere estremamente elevata:
  - Ad esempio, se n è formato da 20 cifre
  - tale quantità è circa 10^{20}
  - anche immaginando di eseguire 10^{10} operazioni al secondo
  - l'algoritmo richiederebbe 10^{10} secondi...
  - ...cioè più di un secolo!

- Secondo algoritmo: basato su una formulazione geometrica
  - Se $x > y$, $MCD(x, y) = MCD(x - y, y)$
    - Ogni numero che divide sia x che y, divide $x - y$
    - Ogni numero che divide sia $x - y$ che y, divide x



- Secondo algoritmo: formulazione algoritmica
  - Fintanto che $n \neq m$, se $n < m$ poni m uguale a $m - n$, altrimenti poni n uguale a $n - m$
  - Quando $n = m$, il loro valore è il MCD
- Correttezza
  - Segue dal fatto che, se $x > y$, $MCD(x, y) = MCD(x - y, y)$
    - Ogni numero che divide sia x che y, divide $x - y$
    - Ogni numero che divide sia $x - y$ che y, divide x
- Caso pessimo?



- Secondo algoritmo: formulazione algoritmica
  - Fintanto che $n \neq m$, se $n < m$ poni m uguale a $m - n$, altrimenti poni n uguale a $n - m$
  - Quando $n = m$, il loro valore è il MCD
- Correttezza
  - Segue dal fatto che, se $x > y$, $MCD(x, y) = MCD(x - y, y)$
    - Ogni numero che divide sia x che y, divide $x - y$
    - Ogni numero che divide sia $x - y$ che y, divide x
- Caso pessimo
  - n molto grande e m molto piccolo
  - Non molto diverso dal primo algoritmo: 50 anni invece di un secolo, quindi sempre inutilizzabile.

Algoritmo di Euclide

- Miglioramento rispetto al secondo algoritmo
  - Cercare di raggiungere un punto sull'asse delle ascisse, saltando direttamente al punto a esso più vicino

Algoritmo di Euclide

- Miglioramento rispetto al secondo algoritmo
  - Cercare di raggiungere un punto sull'asse delle ascisse, saltando direttamente al punto a esso più vicino
  - Fintanto che $n \neq 0$ e $m \neq 0$, se $n < m$ passa alla coppia $(\mathrm{n}, \mathrm{m} \bmod n)$, altrimenti passa alla coppia $(m, n \mod m)$
    - $x \mod y$: resto della divisione di x per y
  - Quando $n = 0$, m è l'MCD (e viceversa, rispettivamente)

Algoritmo di Euclide

- Miglioramento rispetto al secondo algoritmo
  - Cercare di raggiungere un punto sull'asse delle ascisse, saltando direttamente al punto a esso più vicino
  - Fintanto che $n \neq 0$ e $m \neq 0$, se $n < m$ passa alla coppia $(n, m \bmod n)$, altrimenti passa alla coppia $(m, n \bmod m)$
    - $x \bmod y$: resto della divisione di x per y
  - Quando $n = 0$, m è l'MCD (e viceversa, rispettivamente)
- Correttezza
  - vedere libro

Algoritmo di Euclide

- Miglioramento rispetto al secondo algoritmo
  - Cercare di raggiungere un punto sull'asse delle ascisse, saltando direttamente al punto a esso più vicino
  - Fintanto che $n \neq 0$ e $m \neq 0$, se $n < m$ passa alla coppia $(n, m \bmod n)$, altrimenti passa alla coppia $(m, n \bmod m)$
    - $x \bmod y$: resto della divisione di x per y
  - Quando $n = 0$, m è l'*MCD* (e viceversa, rispettivamente)
- Correttezza
  - vedere libro
- Efficienza
  - Ottimale (ma esula da questo corso)
  - Anche con numeri di 20 cifre, l'algoritmo richiede pochi millesimi di secondo!

Algoritmi

- **Informatica**: studio sistematico dei processi algoritmici che descrivono e trasformano l'informazione: la loro teoria, analisi, progettazione, efficienza, implementazione e applicazione
- **Algoritmo**: successione finita di istruzioni o passi che definiscono le operazioni da eseguire su dei dati (che formano l'istanza di un problema) per ottenere dei risultati (intesi come la soluzione dell'istanza specificata)

Proprietà degli algoritmi

- Un algoritmo deve essere
  - **Finito**: ogni istruzione deve essere eseguita in un intervallo finito di tempo e un numero finito di volte.
  - **Generale**: fornire la soluzione per tutti i problemi appartenenti a una data classe.
  - **Non ambiguo**: i passi devono essere definiti in modo univoco e non ambiguo, evitando paradossi.
  - **Corretto**
  - **Efficiente**

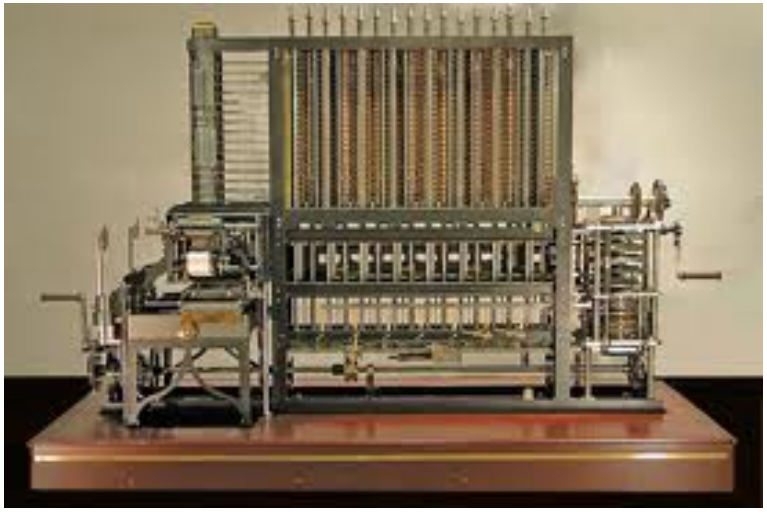
- Un calcolatore consiste di hardware e di software
  - **Hardware**: unità di elaborazione centrale, memoria principale, memoria ausiliaria, periferiche
  - **Software**: istruzioni raccolte in programma

Primi strumenti di calcolo

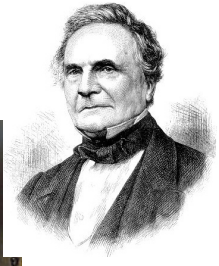
Pascalina

Primi calcolatori

Numerosi approfondimenti in rete, anche in video

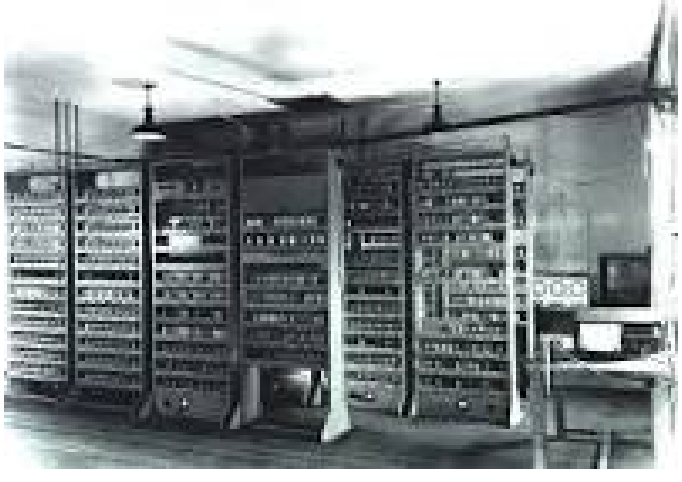


Macchina analitica

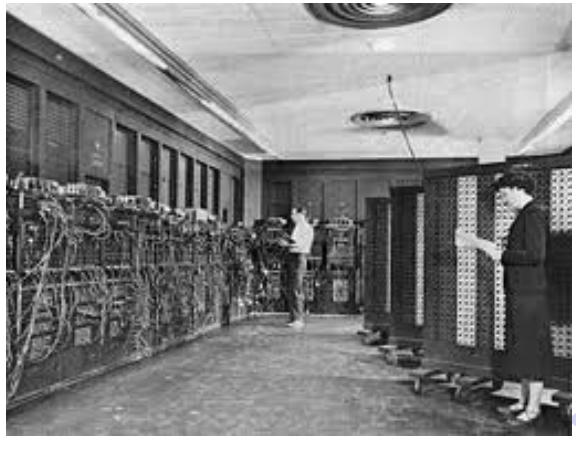


Charles Babbage

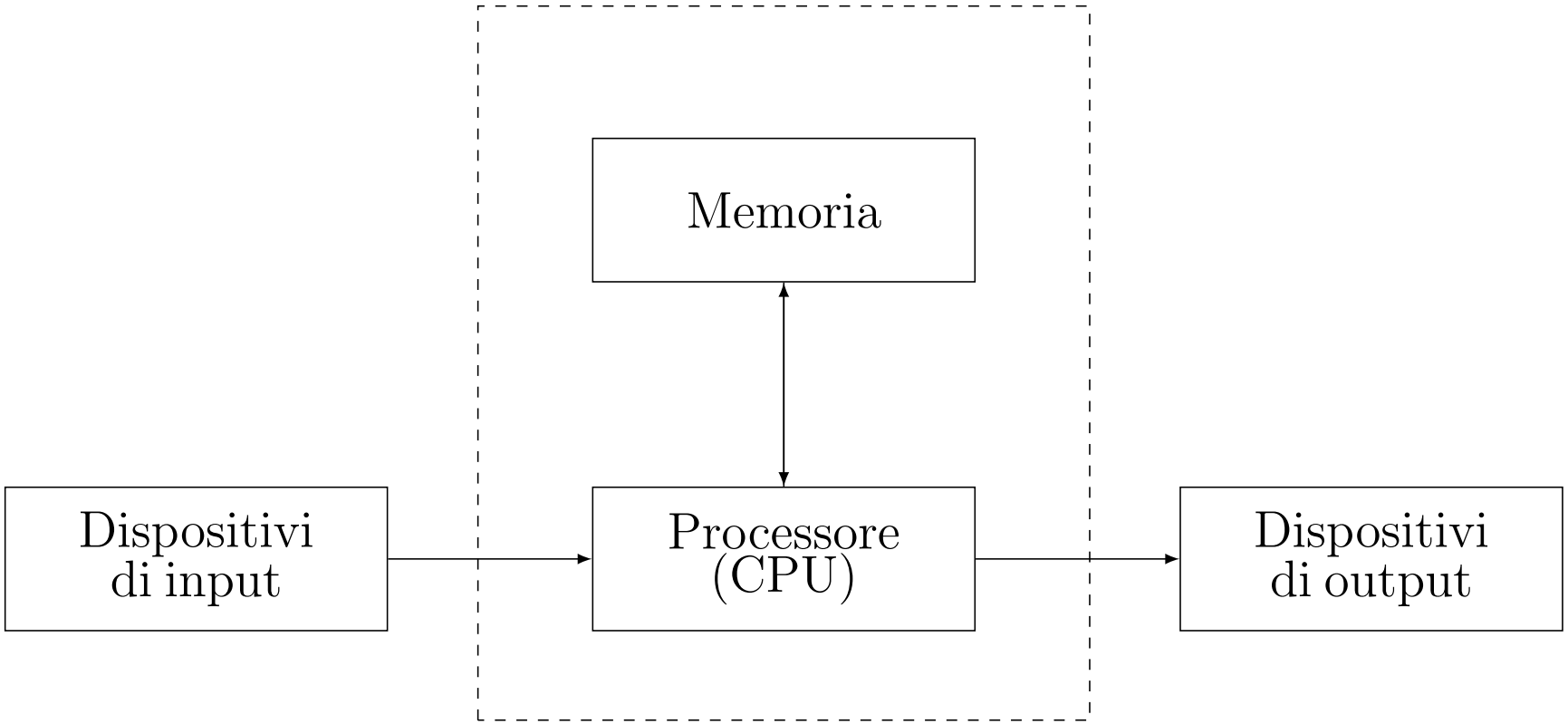
Ada Augusta



EDSAC



ENIAC


Memoria
Dispositivi
di input
Processore
(CPU)
Dispositivi
di output

- **CPU**: dispositivo che esegue le istruzioni di un programma
  - Solo operazioni molto semplici, come trasferimento di un dato oppure operazioni aritmetiche elementari
- **Memoria principale**: veloce, ma costosa e volatile (conserva il programma attualmente in esecuzione ed i dati da esso usati)
- **Memoria ausiliaria**: meno costosa e che perdura anche in assenza di elettricità, ma più lenta (utilizzata per conservare programmi e dati in modo più o meno permanente)

- In un calcolatore i dati e le istruzioni sono codificati in forma binaria
- **Bit**: può assumere due soli valori (0 ed 1): la più piccola unità di informazione memorizzabile o elaborabile
- **Byte**: pari a 8 bit ($2^8 = 256$ possibili valori)

- Sia la memoria principale che quella secondaria sono misurate in **byte**
- la memoria principale non è altro che una lunga lista di posizioni numerate
- ogni posizione contiene un byte
- il numero di una posizione è detto **indirizzo**

- Un calcolatore può trattare diversi tipi di dati (numeri, testi, immagini, suoni)
- Tutti i dati devono essere trasformati in sequenze di bit per poter essere elaborati
- Dati di tipo diverso possono richiedere uno o più byte per essere codificati

- **Locazione di memoria**: sequenza di byte adiacenti associata al dato il cui indirizzo è l'indirizzo del primo byte della sequenza

Indirizzo	Dato	
...	...	...
484	00011110	Primo dato:
485	00001001	2 byte
486	00000100	Secondo dato: 1 byte
487	01001100	
488	01111100	Terzo dato:
489	01010101	4 byte
490	01001001	
491	01000111	Quarto dato:
492	01001001	2 byte
...	...	...

Codice ASCII

	000	001	010	011	100	101	110	111
0000		□	□	□	□	□	□	□
0001	□						□	□
0010	□	□	□	□	□	□	□	□
0011	□	□	□	□	□	□	□	□
0100		!	"	#	$	%	&	'
0101	(	)	*	+	,	-	.	/
0110	0	1	2	3	4	5	6	7
0111	8	9	:	;	<	=	>	?
1000	@	A	B	C	D	E	F	G
1001	H	I	J	K	L	M	N	O
1010	P	Q	R	S	T	U	V	W
1011	X	Y	Z	[	\	]	^	_
1100	‘	a	b	c	d	e	f	g
1101	h	i	j	k	l	m	n	o
1110	p	q	r	s	t	u	v	w
1111	x	y	z	{	|	}	~	□

Numerazione binaria

- Come in quella decimale, posizione di una cifra indica valore relativo
  - Il sistema binario usa potenze crescenti di 2

Binario	Decimale	Binario	Decimale
0000	0	1000	8
0001	1	1001	9
0010	2	1010	10
0011	3	1011	11
0100	4	1100	12
0101	5	1101	13
0110	6	1110	14
0111	7	1111	15

- Notazione realmente usata: **complemento a due**