

# Introduzione agli Algoritmi

# Sommario

- Definizione informale di algoritmo e esempi
- Proprietà degli algoritmi
- Input/Output, Variabili
- Algoritmi senza input o output

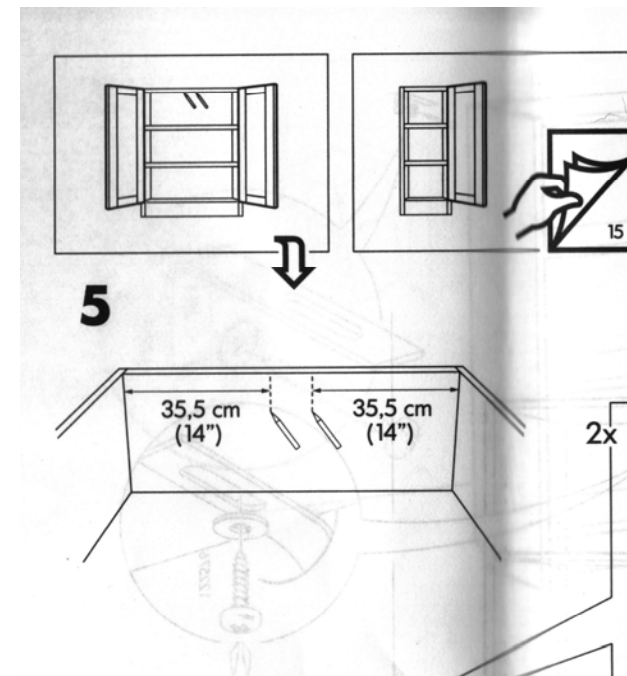
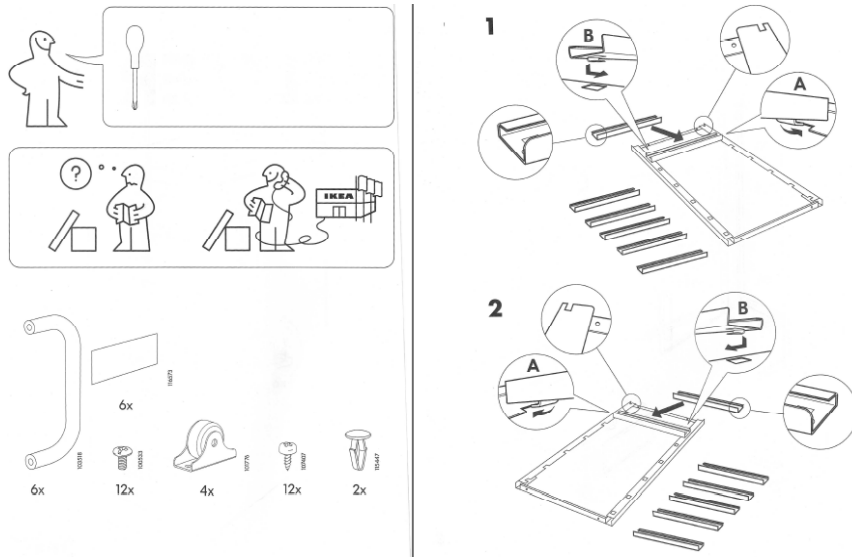
# Algoritmo

- Serie di prescrizioni o istruzioni che specifica l'insieme delle azioni da compiere per poter risolvere un problema
  - [Al-Khowarizmi, IX sec.]
    - Regole per eseguire le 4 operazioni aritmetiche sui numeri scritti in notazione decimale

# Esempi di Algoritmi

- Ricetta da cucina
- Istruzioni per il montaggio dei mobili IKEA
- MCD
- Moltiplicazione per raddoppio
- ...

## Istruzioni Montaggio Mobili (1)



## Istruzioni Montaggio Mobili (2)

## Massimo Comun Divisore: Algoritmo

- **Dati** due **interi**  $n$  e  $m$ , individuare il MCD ( $n, m$ )
- Sia  $n$  il maggiore tra i due e  $m$  il minore
- **Dividere**  $n$  per  $m$ ,
  - si ottiene un **quoziente** ( $q$ ) e un **resto** ( $r$ )
- **Ripetere** l'operazione precedente con  $m$  e  $r$  al posto rispettivamente di  $n$  e  $m$
- **Quando** si ottiene  $r = 0$ 
  - l'algoritmo si **arresta** e
  - il **resto ottenuto** all'iterazione precedente è il MCD cercato

## Calcolo Massimo Comun Divisore: Esempio

- $n = 2079, m = 987$
- $2079 \text{ diviso } 987 = 2 \text{ (} r = 105 \text{)}$
- $987 \text{ diviso } 105 = 9 \text{ (} r = 42 \text{)}$
- $105 \text{ diviso } 42 = 2 \text{ (} r = 21 \text{)}$
- $42 \text{ diviso } 21 = 2 \text{ (} r = 0 \text{, quindi fine iterazione)}$
- $\text{MCD}(2079, 987) = 21$

## Moltiplicazione per raddoppio: Algoritmo

- **Trovare** il prodotto di due fattori  $n$  e  $m$
- **Si compila** una tabella a due colonne
  - La prima riga è costituita da 1 e da un fattore
  - **finché** nella prima colonna si ottengono numeri **non maggiori** del secondo fattore, si aggiunge una riga raddoppiando gli elementi della precedente
- Si scelgono gli elementi della prima colonna la cui somma è uguale al secondo fattore
- La somma dei corrispondenti elementi della seconda colonna fornisce il risultato

## Moltiplicazione per raddoppio: Esempio

- $n=18$  e  $m=13$

1            18

2            36

4            72

8            144

16 (> 13 quindi **fine iterazione**)

$1+4+8=13$

Quindi  $18 + 72 + 144 = 234 = 18 \times 13$

## Proprietà degli algoritmi

- **Finitezza**
  - Spaziale: espressi in uno spazio finito
  - Temporale: esecuzione in tempo finito
- **Generalità**
  - Classe di problemi
  - Dominio di definizione
- **Completezza**
  - tutte le istanze del problema devono essere risolubili dall'algoritmo
- **Non ambiguità**
  - le istruzioni devono essere univocamente interpretabili
- **Eseguibilità**
  - l'esecutore deve essere in grado di eseguire ogni istruzione

## Algoritmi Deterministici

- È detto **deterministico** un algoritmo tale per cui al momento dell'esecuzione di ogni istruzione è nota l'istruzione successiva
  - Come effetto, due diverse esecuzioni dello stesso algoritmo deterministico con gli stessi dati producono gli stessi risultati
  - Esistono algoritmi **non deterministici**, di cui non ci occuperemo

## Altre caratteristiche

- Alcuni algoritmi possono dare risposte del tipo SI'/NO oppure VERO/FALSO
- **NON** è necessario che i dati elaborati dagli algoritmi siano numerici

## Altre caratteristiche: Esempio 1

- Verificare se una stringa di caratteri è palindroma
  - È palindroma una stringa se è uguale leggerla da sinistra a destra o viceversa:
    - “**ossesso**” è palindroma, “**possesso**” no;
    - “**a3#3a**” è palindroma, “**a3h#a**” no;
    - “**QFWFQ**” è palindroma, “**(K)YK**” no
- Confrontare il primo e l'ultimo carattere; se sono diversi la risposta è no; fine algoritmo
- Se sono uguali, cancellare il primo e l'ultimo carattere e ripetere il procedimento, sino a che ci sono caratteri

## Altre caratteristiche: Esempio 2

- Verificare se un elemento appartiene a una lista di elementi:
- Finché ci sono elementi nella lista, confrontare l'elemento cercato con il primo della lista
  - Se sono uguali, la risposta è SI'; fine algoritmo
  - Altrimenti passare all'elemento successivo della lista
- Il precedente algoritmo è noto come **ricerca sequenziale**

## Input e Output (1)

- Gli algoritmi per poter godere della proprietà di generalità, devono essere in grado di risolvere **una classe di problemi**
  - L'algoritmo seguente calcola l'ipotenusa di **un triangolo rettangolo, i cui cateti hanno valori 3 e 4**
    - Calcola il quadrato di 3 e somma ad esso il quadrato di 4, quindi estrai la radice quadrata del valore ottenuto
  - L'algoritmo seguente calcola l'ipotenusa di **tutti i triangoli rettangolo, i cui cateti hanno valori noti**
    - Calcola il quadrato di un cateto e aggiungi ad esso il quadrato dell'altro, quindi estrai la radice quadrata della somma ottenuta

## Input e Output (2)

- La generalizzazione è ottenuta mediante **parametrizzazione** dei valori su cui opera l'algoritmo
- Per risolvere una **istanza** del problema, si sostituiscono ai parametri le istanze dei valori
- I dati forniti in ingresso, sui cui opera l'algoritmo sono detti **Input**
- I risultati forniti dall'esecuzione dell'algoritmo sono gli **Output**

## Input e Output (3)

- Negli esempi precedenti
  - Per l'algoritmo del MCD 2079 e 987 sono input e 21 è l'output
  - Per il prodotto, 18 e 13 sono input e 234 output
  - Per il palindromo, "ossesso", "possesso", "a3#3a", "a3h#a" sono input, "SI" e "NO" output
  - L'elemento da ricercare e la lista degli elementi sono input, "TROVATO" / "NON TROVATO" output

## Input e Output (4)

- Possono esistere algoritmi che **NON** hanno I/O
- Esempio: calcolo della differenza tra n e m:
- Se  $n \geq m$ , allora output (n-m), fine algoritmo
- Altrimenti fine algoritmo

## Variabili

- Oltre ai dati di I/O, gli algoritmi spesso utilizzano dati di supporto per svolgere le proprie attività, detti **variabili**
- Per il momento possiamo assumere che una variabile sia un **contenitore di un valore**
  - I valori possono essere di diversi **tipi**, ad es. numeri interi, reali, caratteri singoli, stringhe di caratteri, valori logici, etc.
- Anche i dati di I/O sono variabili