

Rappresentazione e codifica dell'informazione



Prof.ssa Lucia Migliorelli

lmigliorelli@unite.it

Corso di Sistemi multimediali e web per il turismo

Dipartimento di Scienze Politiche, Università di Teramo

Agenda

- Digitalizzazione dell'informazione
- Rappresentazione dei caratteri
- Codifica dei caratteri
- Codice ASCII
- Codice ASCII esteso
- Codifiche ISO - Unicode
- I testi
- I vantaggi di un testo elettronico
- Acquisizione di un testo

Definizione di Informatica

Secondo la “Association for Computer Machinery (ACM)”

L'informatica è lo studio sistematico degli algoritmi che descrivono e trasformano l'informazione: la loro teoria, analisi, progetto, efficienza, realizzazione e applicazione.

Digitalizzazione dell'informazione

Informatica, definizione:

«Scienza che studia l'elaborazione delle informazioni e le sue applicazioni; più precisamente l'informatica si occupa della rappresentazione, dell'organizzazione e del trattamento automatico della informazione.»

(Da <https://www.treccani.it/enciclopedia/ricerca/digitalizzare/>)

Codificare informazione multimediale in forma digitale

Concepire algoritmi per elaborarla al fine di risolvere problemi dati

Informatica

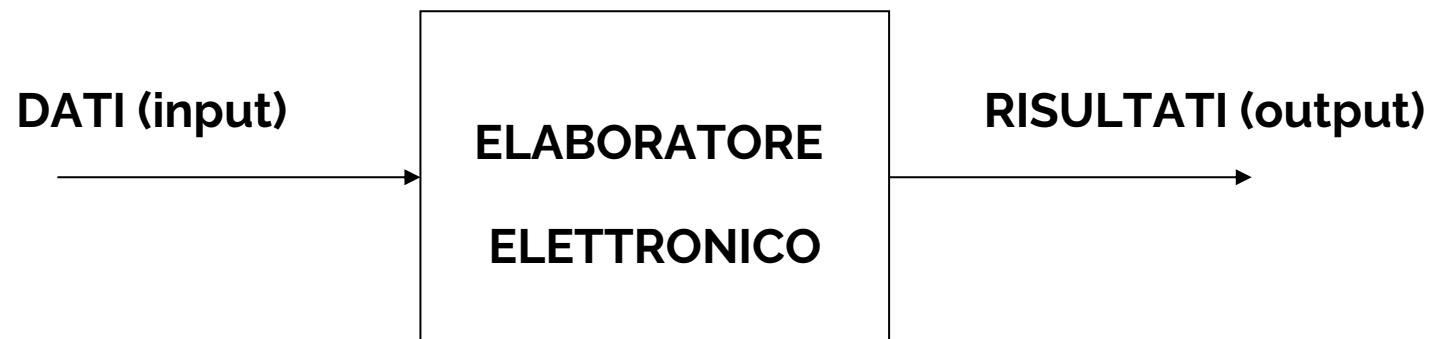
- L'informatica comprende:
 - Architettura dei calcolatori
 - Sistemi operativi
 - Reti di calcolatori
 - Metodi per la rappresentazione delle informazioni
 - Metodi per la rappresentazione delle soluzioni
 - Linguaggi di programmazione

Elaboratore elettronico

- Il calcolatore elettronico (computer) è uno strumento in grado di eseguire un insieme di azioni elementari
- le azioni vengono eseguite su oggetti (**dati**) per produrre altri oggetti (**risultati**)
- l'esecuzione di azioni viene richiesta all'elaboratore attraverso frasi scritte in un qualche linguaggio di programmazione (**istruzioni**)

Programmazione

L'attività con cui si predispone l'elaboratore a eseguire un particolare insieme di azioni su dati specifici, allo scopo di risolvere un problema.



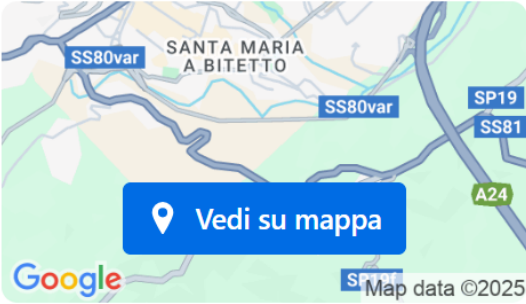
 Teramo ×

 mar 7 ott — gio 9 ott

 1 adulto · 0 bambini · 1 camera ▼

Cerca

[Home](#) > [Italia](#) > [Abruzzo](#) > [Teramo](#) > Risultati della ricerca


[Vedi su mappa](#)

Teramo: 15 strutture trovate

↓↑ Ordina per: Scelte top per i viaggiatori singoli ◇

La commissione pagata sulle prenotazioni e altri fattori potrebbero influenzare il pos
risultati di ricerca. Leggi maggiori dettagli su questi parametri di posizionamento e su come sceglierli e modificarli.
[Scopri di più](#)

noi inseriamo informazioni
(data, città, numero di persone) e il
sistema elabora (un algoritmo) ci
restituendoci il risultato:

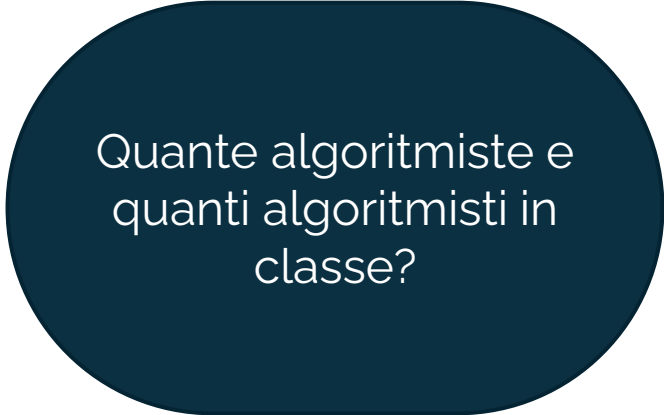
l'elenco degli hotel disponibili

Problemi da risolvere

- I problemi che si possono risolvere attraverso l'uso di un calcolatore sono di natura molto varia:
 - Dati due numeri trovare il **maggiore**
 - Dato un elenco di nomi e relativi numeri di telefono **trovare** il numero di telefono di una determinata persona
 - Dati a e b, **risolvere l'equazione** $ax+b=0$
 - Stabilire se una parola viene **alfabeticamente** prima di un'altra
 - **Somma** di due numeri interi
 - **Ordinare** una lista di elementi
 - Calcolare il **massimo comune divisore** fra due numeri dati
 - Calcolare il **massimo** in un insieme

Algoritmo

- Un algoritmo è una sequenza finita di azioni che risolvono in un tempo finito una classe di problemi
- L'esecuzione delle azioni nell'ordine specificato dall'algoritmo consente di ottenere, a partire dai dati iniziali, i risultati che risolvono il problema
- L'esecuzione delle azioni specificate dall'algoritmo avviene attraverso una macchina astratta (ESECUTORE)

A dark blue rounded rectangle with a thin white border, containing white text.

Quante algoritmiste e
quanti algoritmisti in
classe?

Digitalizzazione dell'informazione

Algoritmo:

Insieme di regole volte a risolvere un determinato problema in un **numero finito** di passi (operazioni, istruzioni)



Digitalizzazione dell'informazione

Algoritmo, Esempio:

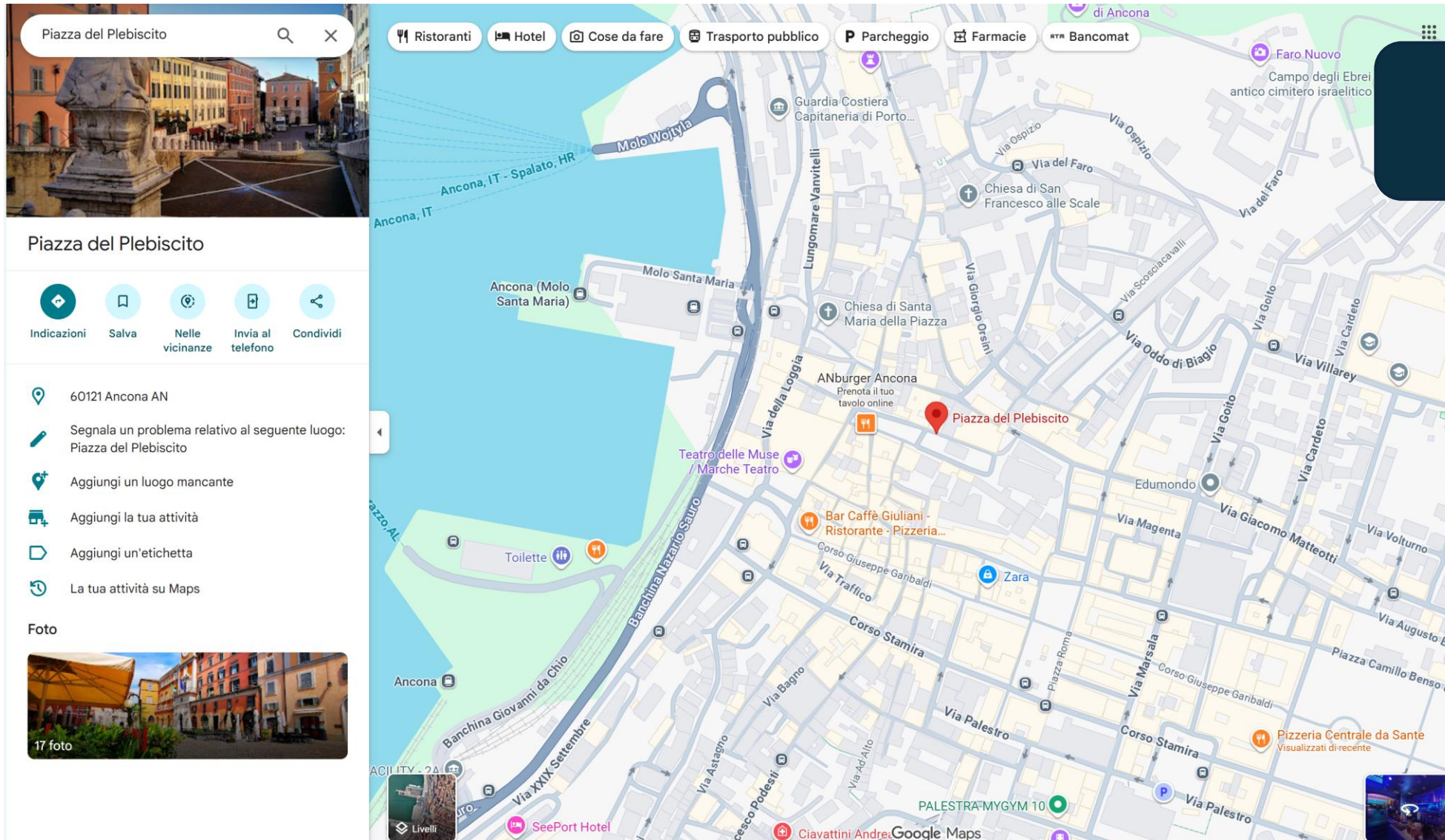
Carbonara

1. Prendere un uovo per commensale
2. Aggiungere pecorino q.b.
3. Sbattere le uova
4. Unire guanciale
5. Portare dell'acqua ad ebollizione
6. Salare l'acqua
7. Immergere la pasta nell'acqua
8. Scolare dopo 10 minuti
9. Versare la pasta in una terrina
10. Mescolare

Proprietà degli algoritmi (1)

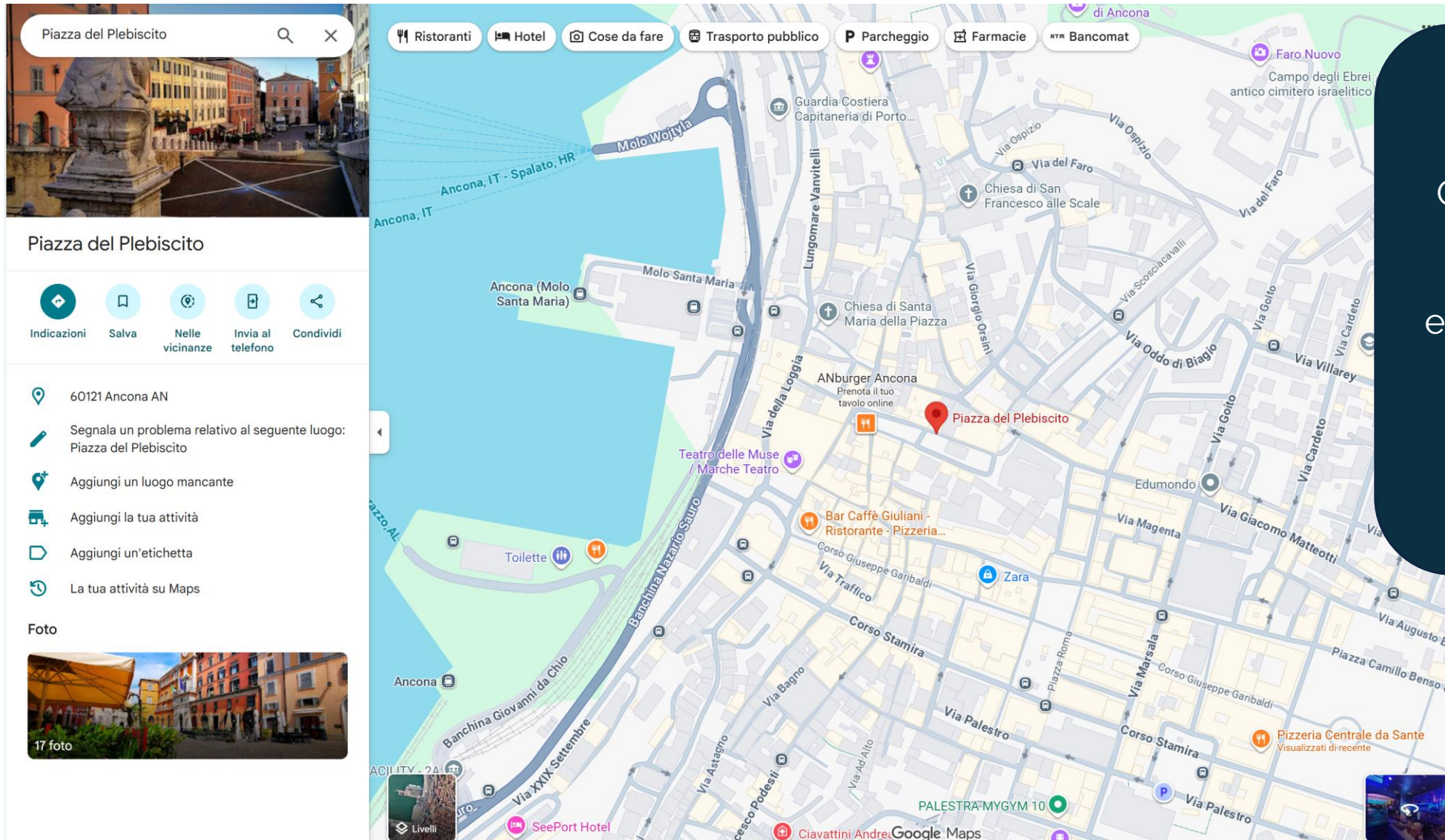
- ESEGUIBILITA': ogni azione deve essere eseguibile dall'esecutore in un tempo finito
- NON-AMBIGUITÀ: ogni azione deve essere univocamente interpretabile dall'esecutore
- FINITEZZA: il numero totale di azioni da eseguire, per ogni insieme di dati di ingresso, deve essere finito

Proprietà degli algoritmi (2)



Eseguibilità?

Proprietà degli algoritmi (3)

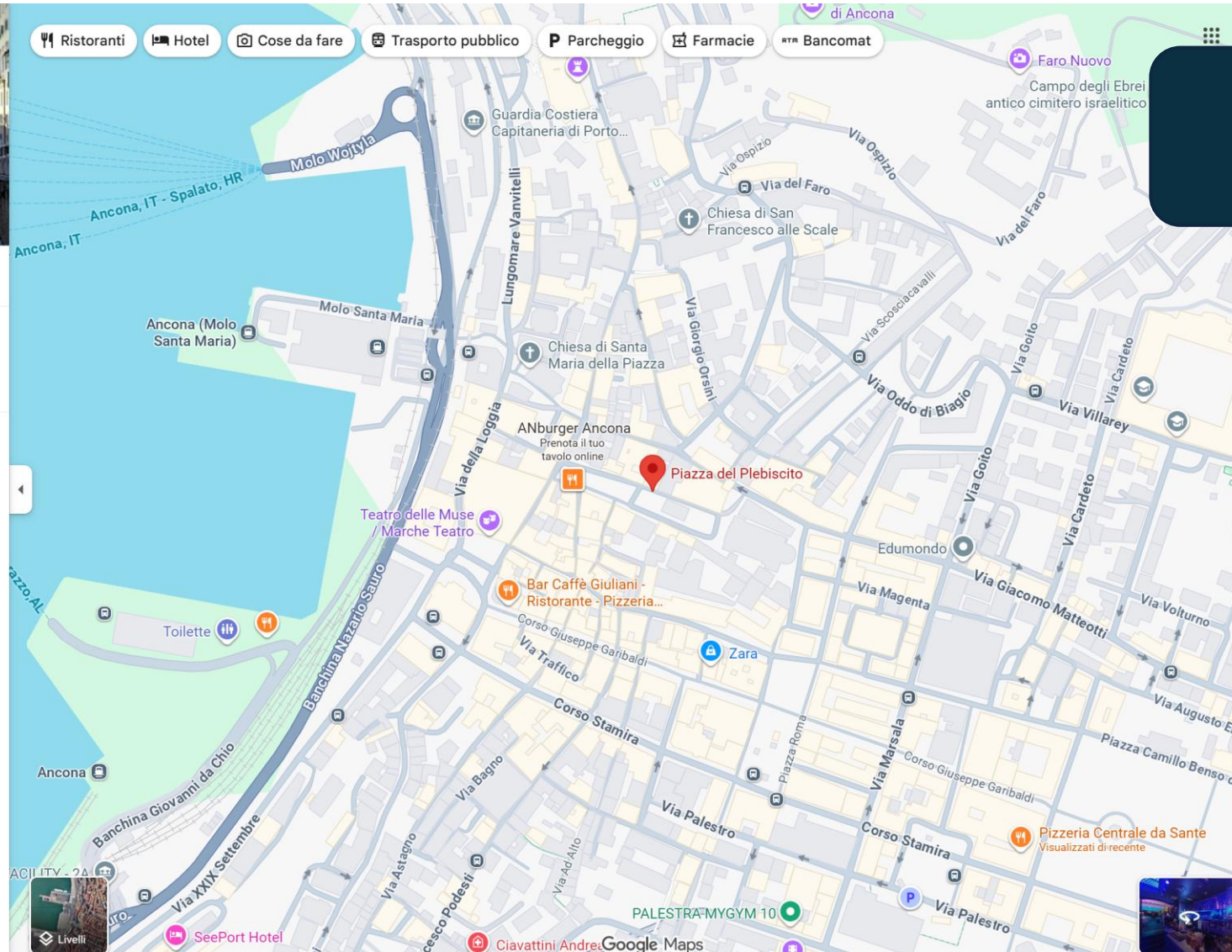


Eseguibilità?

Ogni istruzione deve poter essere «**fisicamente**» eseguibile dall'utente.

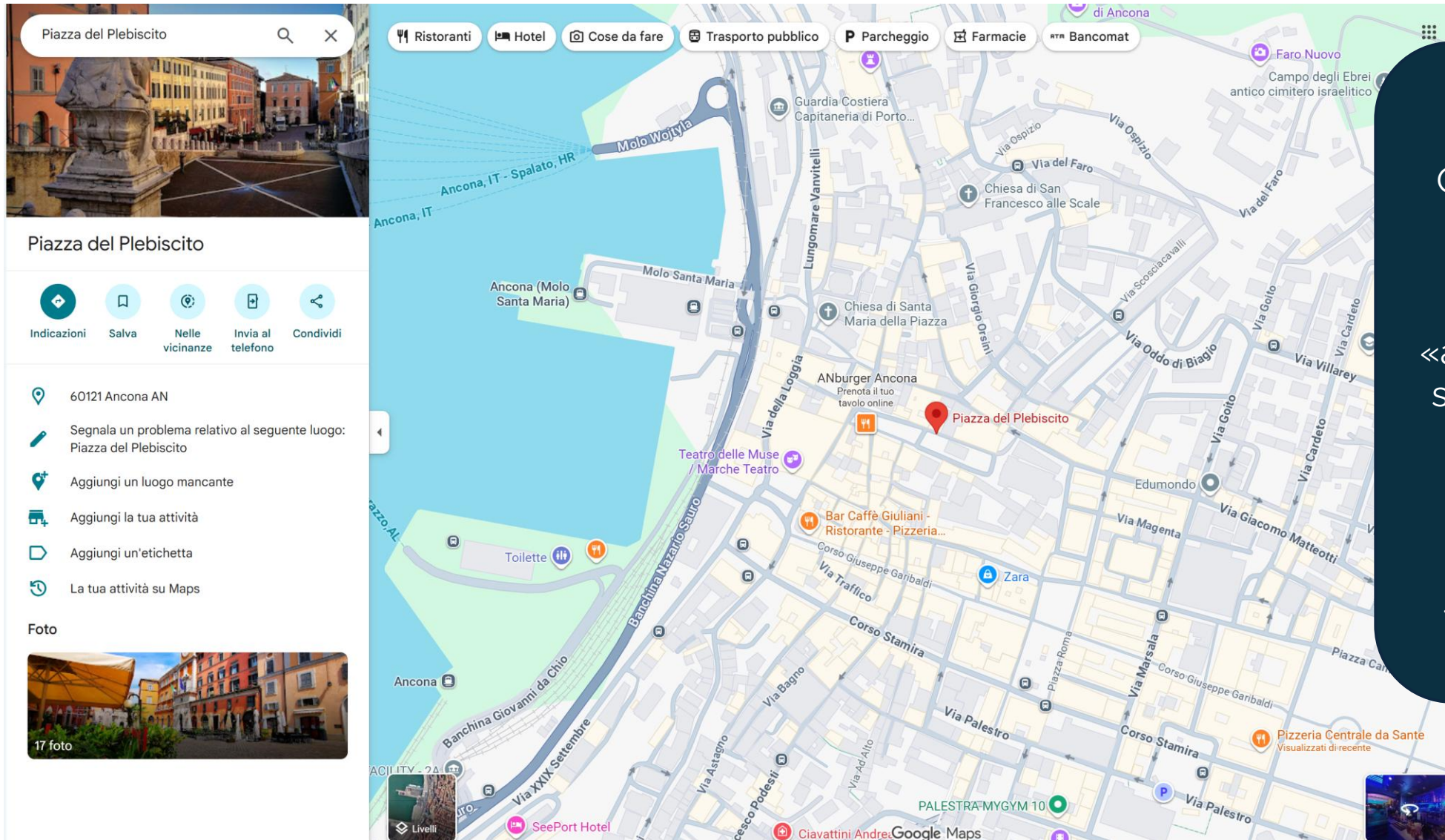
Controesempio:
attraversa il molo a nuovo

Proprietà degli algoritmi (4)



Non ambiguità?

Proprietà degli algoritmi (5)



Non ambiguità?

Ogni istruzione deve avere un'unica interpretazione

«alla rotonda prendi la seconda uscita verso via Pizzecolli»

Controesempio:

«gira a sinistra più o meno qui»

Proprietà degli algoritmi (6)

Piazza del Plebiscito



Indicazioni

Salva

Nelle vicinanze

Invia al telefono

Condividi

 60121 Ancona AN

 Segnala un problema relativo al seguente luogo:
Piazza del Plebiscito

 Aggiungi un luogo mancante

 Aggiungi la tua attività

 Aggiungi un'etichetta

 La tua attività su Maps

Foto



17 foto

Ristoranti

Hotel

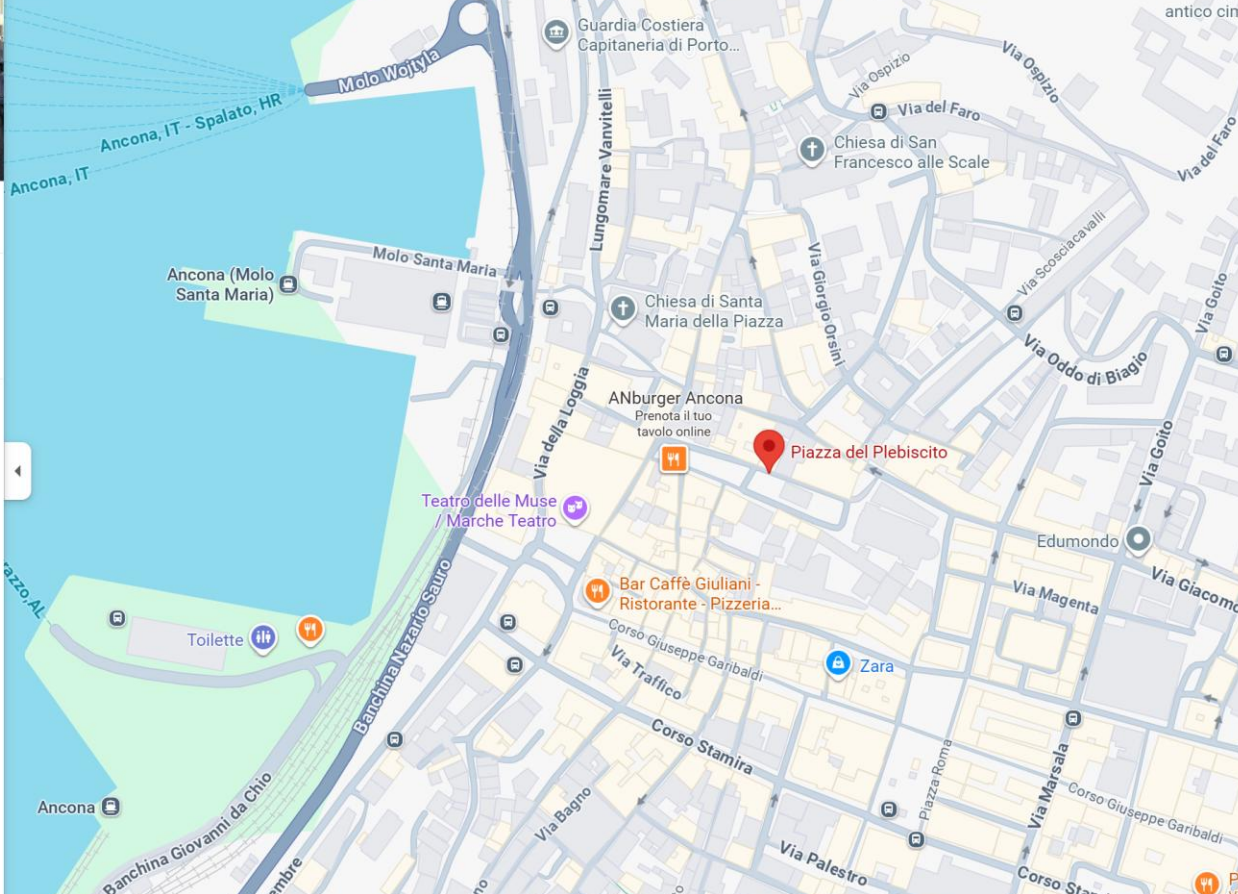
Cose da fare

Trasporto pubblico

P Parcheggio

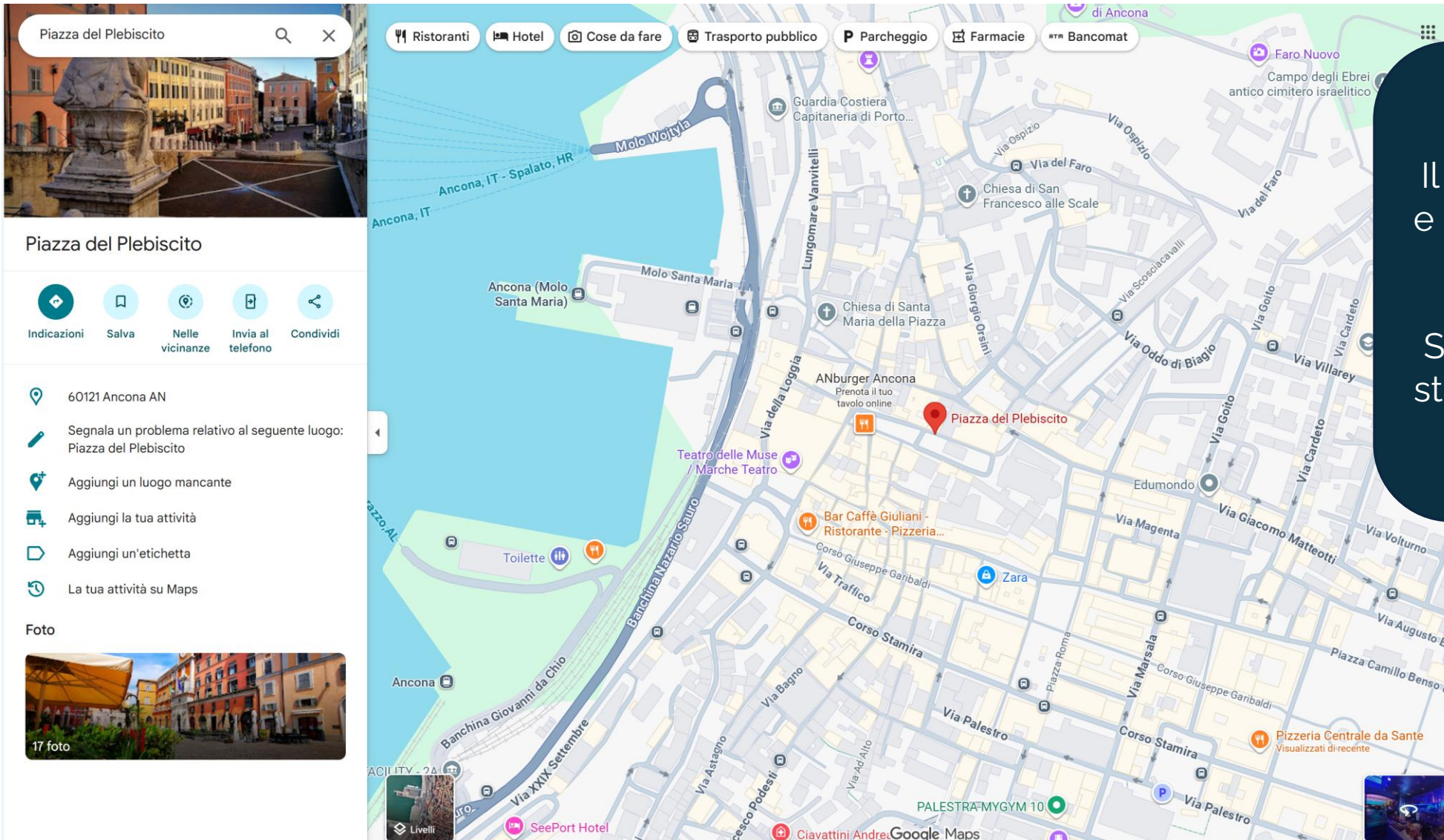
Farmacie

Bancomat



Finitezza?

Proprietà degli algoritmi (7)



Finitezza?

Il calcolo del percorso
e la sequenza dei passi
terminano

Se si sbaglia strada, la
strada viene ricalcolata
e ogni ricalcolo
termina

Algoritmi e programmi (1)

- Ogni elaboratore elettronico è una macchina in grado di eseguire azioni elementari su oggetti detti DATI
- L'esecuzione delle azioni è richiesta all'elaboratore tramite comandi elementari detti ISTRUZIONI espresse attraverso un opportuno formalismo: il LINGUAGGIO di PROGRAMMAZIONE
- La formulazione testuale di un algoritmo in un linguaggio comprensibile a un elaboratore è detta PROGRAMMA

Algoritmi e programmi (2)

Dati:

le temperature
orarie in °C di oggi:

D=[18,20,23,22,19]

Azioni/istruzioni:

passi elementari che
l'elaboratore sa eseguire
(somma, confronto,
assegnazione, salto,
lettura/scrittura in memoria,
ecc.).

Linguaggio di
programmazione:

Formalismo con cui
scrivo le istruzioni in
modo che il
calcolatore le
capisca (es. python)

Programma:

la formulazione
testuale completa
di un algoritmo in
un linguaggio (il file
di codice che
eseguo).

Algoritmi e programmi (3)

Obiettivo dell'algoritmo)

Calcolare la media delle temperature in montagna

Dati = [18,20,23,22,19]

Istruzioni

- Inizializza somma = 0
- Per ogni valore t in D, aggiorna somma = somma + t
- Calcola media = somma / n, dove n è la quantità di valori
 - Restituisci media

Linguaggio di programmazione & programma

```
D = [18, 20, 23, 22, 19]    # DATI (input)
somma = 0                   # ISTRUZIONI
for t in D:                 # ISTRUZIONI
    somma = somma + t       # ISTRUZIONI
media = somma / len(D)      # ISTRUZIONI
print(media)               # OUTPUT (parte del PROGRAMMA)
```

Zoom sul concetto di programma

- Un programma è un testo scritto in accordo alla **sintassi** e alla **semantica** di un linguaggio di programmazione
- Un programma è la formulazione testuale, in un certo linguaggio di programmazione, di un algoritmo che risolve un dato problema

Sintassi e semantica di un linguaggio (1)

- **SINTASSI:** insieme di regole formali per la scrittura di programmi in un linguaggio, che dettano le modalità per costruire frasi corrette nel linguaggio stesso
 - È come la grammatica di una lingua naturale: stabilisce le regole su come scrivere correttamente.
 - In un linguaggio di programmazione, la sintassi dice come vanno messi i simboli, le parentesi, le parole chiave, ecc.
 - Es: `print("ciao` → non è sintatticamente corrente in un linguaggio come python in quanto non sono presenti `«"»` e parentesi tonda di chiusura

Sintassi e semantica di un linguaggio (2)

- **SINTASSI:** insieme di regole formali per la scrittura di programmi in un linguaggio, che dettano le modalità per costruire frasi corrette nel linguaggio stesso
 - È come la grammatica di una lingua naturale: stabilisce le regole su come scrivere correttamente.
 - In un linguaggio di programmazione, la sintassi dice come vanno messi i simboli, le parentesi, le parole chiave, ecc.
 - Es: `print("ciao` → non è sintatticamente corrente in un linguaggio come python in quanto non sono presenti `«"»` e parentesi tonda di chiusura
- **SEMANTICA:** insieme dei significati da attribuire alle frasi (sintatticamente corrette) costruite nel linguaggio ovvero: una volta rispettata la sintassi, bisogna capire **cosa fa il programma**.
 - `X= "ciao " + "mondo"`
 - Nella variabile X è salvata una concatenazione di «stringhe» (ovvero di un dato non numerico)

Definizione di linguaggio (1)

- Un linguaggio è un insieme di frasi:
 - Una frase è una sequenza di simboli appartenenti a un certo alfabeto.
 - L'alfabeto è l'insieme dei simboli di base che possiamo usare.

{?????}

- Un LINGUAGGIO deve essere effettivamente generabile
- Un LINGUAGGIO di PROGRAMMAZIONE deve essere decidibile

Definizione di linguaggio (2)

- Un linguaggio è un insieme di frasi:
 - Una frase è una sequenza di simboli appartenenti a un certo alfabeto.
 - L'alfabeto è l'insieme dei simboli di base che possiamo usare.
 - Alfabeto {a,b}
 - Possibili frasi {aa, bb, ab, ba, ...}
 - L'insieme di tutte queste frasi costituisce un linguaggio
- Un LINGUAGGIO deve essere effettivamente generabile
- Un LINGUAGGIO di PROGRAMMAZIONE deve essere decidibile

Definizione di linguaggio (3)

- Un linguaggio è un insieme di frasi:
 - Una frase è una sequenza di simboli appartenenti a un certo alfabeto.
 - L'alfabeto è l'insieme dei simboli di base che possiamo usare.
- Un LINGUAGGIO deve essere effettivamente generabile

{?????}

- Un LINGUAGGIO di PROGRAMMAZIONE deve essere decidibile

Definizione di linguaggio (4)

- Un linguaggio è un insieme di frasi:
 - Una frase è una sequenza di simboli appartenenti a un certo alfabeto.
 - L'alfabeto è l'insieme dei simboli di base che possiamo usare.
- Un LINGUAGGIO deve essere effettivamente generabile

Esiste un metodo (una regola, un algoritmo) per produrre tutte le frasi del linguaggio.

- Linguaggio: {a,b}
- Regola per generarlo: parti dalla stringa vuota "" e ad ogni passo aggiungi due a oppure qualsiasi numero di b.
- Si può costruire un generatore che produce: "", bb, aa, abba, bbaa, ecc.
- Un LINGUAGGIO di PROGRAMMAZIONE deve essere decidibile

Definizione di linguaggio (5)

- Un linguaggio è un insieme di frasi:
 - Una frase è una sequenza di simboli appartenenti a un certo alfabeto.
 - L'alfabeto è l'insieme dei simboli di base che possiamo usare.
- Un LINGUAGGIO deve essere effettivamente generabile

{?????}

- Un LINGUAGGIO di PROGRAMMAZIONE deve essere decidibile

{?????}

Definizione di linguaggio (6)

- Un linguaggio è un insieme di frasi:
 - Una frase è una sequenza di simboli appartenenti a un certo alfabeto.
 - L'alfabeto è l'insieme dei simboli di base che possiamo usare.
- Un LINGUAGGIO deve essere effettivamente generabile
- Un LINGUAGGIO di PROGRAMMAZIONE deve essere decidibile
 - Un programma, decide con certezza: "Sì, ciò che è scritto è valido" "No, ciò che è scritto contiene errori di sintassi"
 - `print("ciao")` → valido (appartiene al linguaggio Python).
 - `print("ciao"` → non valido (parentesi non chiusa).
 - **Il compilatore di Python riesce a decidere immediatamente se la stringa è accettata o no.**

Linguaggi di programmazione

- **Linguaggio Macchina:**

→Linguaggio formale che un computer è in grado di interpretare ed eseguire senza mediazioni. I programmi sono rappresentati da una sequenza di cifre binarie che codificano le istruzioni e i dati su cui lavora la CPU.

- **Linguaggio Assembler:**

→Linguaggio in cui le singole istruzioni binarie sono rappresentate da un codice mnemonico più facilmente comprensibile per l'uomo.

- **Linguaggio di Alto Livello:**

→Linguaggi che utilizzano simboli matematici e parole tipiche delle lingue naturali (soprattutto inglese) e sono di facile interpretazione.

Sviluppo di un programma (1)

- Fase di EDITING: Scrivere il testo del programma e memorizzarlo su supporti di memoria permanenti
- Se il linguaggio è *compilato*:
 - Compilare il programma, ossia utilizzare il compilatore che effettua una traduzione automatica del programma scritto in un linguaggio qualunque in un programma equivalente scritto in linguaggio macchina
 - Eseguire il programma tradotto
- Se il linguaggio è *interpretato*:
 - Usare l'interprete per eseguire il programma

Sviluppo di un programma (2)

Fase di Editing: Lo sviluppatore scrive il programma (ad esempio in Python, Java, C, ecc.); Il file viene salvato su un supporto di memoria permanente (es. hard disk, chiavetta USB).

Se il linguaggio è compilato

- Serve un compilatore: un programma che traduce il codice sorgente (scritto dal programmatore) in linguaggio macchina, cioè quello che il computer capisce davvero (sequenze di 0 e 1). Dopo la compilazione, si ottiene un file eseguibile. L'utente può quindi eseguire il programma tradotto senza più bisogno del codice sorgente.

Esempio: In C scrivo hello.c; lo compilo → ottengo hello.exe (su Windows); Faccio doppio clic e il programma parte.

Se il linguaggio è interpretato

- Non c'è traduzione in un file eseguibile. Serve un interprete, che legge il programma riga per riga ed esegue direttamente le istruzioni.

Esempio: In Python scrivo hello.p; Per eseguirlo scrivo: python hello.py; L'interprete Python legge ed esegue subito il codice.

Differenza tra compilato ed interpretato

- Compilato → traduzione una volta sola, poi eseguo velocemente quante volte voglio.
- Interpretato → il codice viene letto ed eseguito ogni volta.

Sviluppo di un programma (3)

Fase di Editing: Lo sviluppatore scrive il programma (ad esempio in Python, Java, C, ecc.); Il file viene salvato su un supporto di memoria permanente (es. hard disk, chiavetta USB).

Se il linguaggio è compilato

- Serve un compilatore: un programma che traduce il codice sorgente (scritto dal programmatore) in linguaggio macchina, cioè quello che il computer capisce davvero (sequenze di 0 e 1). Dopo la compilazione, si ottiene un file eseguibile. L'utente può quindi eseguire il programma tradotto senza più bisogno del codice sorgente.

Esempio: In C scrivo hello.c; lo compilo → ottengo hello.exe (su Windows); Faccio doppio clic e il programma parte.

Sviluppo di un programma (4)

Fase di Editing: Lo sviluppatore scrive il programma (ad esempio in Python, Java, C, ecc.); Il file viene salvato su un supporto di memoria permanente (es. hard disk, chiavetta USB).

Se il linguaggio è interpretato

- Non c'è traduzione in un file eseguibile. Serve un interprete, che legge il programma riga per riga ed esegue direttamente le istruzioni.
- Esempio: In Python scrivo hello.p; Per eseguirlo scrivo: python hello.py; L'interprete Python legge ed esegue subito il codice.

Sviluppo di un programma (5)

Fase di Editing: Lo sviluppatore scrive il programma (ad esempio in Python, Java, C, ecc.); Il file viene salvato su un supporto di memoria permanente (es. hard disk, chiavetta USB).

Se il linguaggio è interpretato

- Non c'è traduzione in un file eseguibile. Serve un interprete, che legge il programma riga per riga ed esegue direttamente le istruzioni.
- Esempio: In Python scrivo hello.p; Per eseguirlo scrivo: python hello.py; L'interprete Python legge ed esegue subito il codice.

Differenza tra compilato ed interpretato

- Compilato → traduzione una volta sola, poi eseguo velocemente quante volte voglio.
- Interpretato → il codice viene letto ed eseguito ogni volta.

Digitalizzazione dell'informazione

Informazione, Definizione:

«Notizia, dato o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni, modi di essere. In senso più generale, anche la trasmissione dei dati e l'insieme delle strutture che la consentono»

(da <https://www.treccani.it/enciclopedia/informazione>)

Digitalizzazione dell'informazione

Informazione, definizione:

Trasferimento di contenuti (che si trasformeranno dentro di noi in concetti)

Per trasferire e memorizzare questi contenuti in un supporto serve una grandezza fisica che abbia almeno due stati.

Es:

- Pressione
- Tensione
- Colore
- ...

Digitalizzazione dell'informazione

Digitalizzare un'informazione: rappresentarla come una sequenza di numeri (dall'inglese *digit* – cifra – che deriva dal latino *digitus* – dito)

«**Digitalizzazione:** Nella tecnica, conversione di grandezze analogiche in informazioni digitali, effettuata mediante un dispositivo, detto digitalizzatore o convertitore analogico-digitale.»

(Da <https://www.treccani.it/enciclopedia/ricerca/digitalizzare/>)

Digitalizzazione dell'informazione

Un elaboratore digitale rappresenta i numeri con **solo due cifre**, 0 e 1 (**codifica binaria**)

Possiamo digitalizzare

- Testi
- Suoni
- Immagini
- Filmati
- E ovviamente qualsiasi tipo di numero

Digitalizzazione dell'informazione

Programmare:

Predisporre un elaboratore ad eseguire una sequenza di azioni (***algoritmo***) su un'informazione digitale presa in ingresso (***input***) al fine di produrre in uscita (***output***) un'informazione digitale, funzione di quella presa in ingresso, che risolva un problema dato

Digitalizzazione dell'informazione

Esempio, Riconoscimento Facciale:

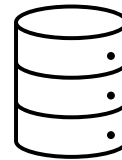
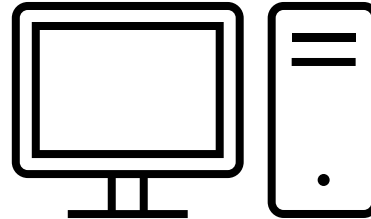


codifica binaria
di un'immagine

INPUT

00110010
00011101
11000011
...
11000101

PROGRAMMA



ARCHIVIO

OUTPUT

00000010
00010001
11000111
...
11110101

decodifica binaria
di un testo

Jennifer Lawrence
Bradley Cooper
Brad Pitt
Kevin Spacey
Meryl Streep

Digitalizzazione dell'informazione

Sistema numerico decimale

Sistema di numerazione posizionale in base 10. Usa 10 cifre per rappresentare i numeri {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

$$250 = 2 \times 10^2 + 5 \times 10^1 + 0 \times 10^0$$

Sistema numerico binario

Sistema di numerazione posizionale in base 2. Usa due cifre per rappresentare i numeri {0, 1}

$$\begin{aligned} 11111010 &= 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\ &= 128 + 64 + 32 + 16 + 8 + 0 + 2 + 0 \\ &= 250 \end{aligned}$$

Digitalizzazione dell'informazione

Codifica Binaria:

Bit (Binary Digit): unità elementare dell'informazione trattata da un elaboratore. Può assumere due (2^1) valori: **0** oppure **1**.

- Due bit (2 b): quattro (2^2) possibili valori {00, 01, 10, 11}
- Tre bit (3 b): otto (2^3) possibili valori {000, 001, 010, 011, 100, 101, 110, 111}
- Quattro bit (4 b): sedici (2^4) possibili valori
- ...
- Otto bit (8 b): 256 (2^8) possibili valori

8 bit (8 b) = 1 Byte (1 B)

Digitalizzazione dell'informazione

Codifica Binaria:

Qual è il numero minimo di bit necessario per rappresentare i mesi dell'anno? **4 bit**

3 bit non sono sufficienti.
Ci sono infatti 8 possibili
combinazioni (2^3), ma i
mesi sono 12

4 bit offrono 16 possibili
combinazioni (2^4).

Gennaio	0001	Luglio	0111
Febbraio	0010	Agosto	1000
Marzo	0011	Settembre	1001
Aprile	0100	Ottobre	1010
Maggio	0101	Novembre	1011
Giugno	0110	Dicembre	1100

Digitalizzazione dell'informazione

Byte – Unità di misura

Valore (in Byte)	Nome	Abbreviazione	Potenza
1	Byte	B	2^0
1 024	Kilobyte	KB	2^{10}
1 048 576	Megabyte	MB	2^{20}
1 073 741 824	Gigabyte	GB	2^{30}
1 099 511 627 776	Terabyte	TB	2^{40}
1 125 899 906 842 624	Petabyte	PB	2^{50}



Ogni salto di unità
cresce di un fattore
 2^{10}

Digitalizzazione dell'informazione

Byte – Unità di misura

Spesso i multipli del byte vengono arrotondati a potenze di 10 invece che di 2, sebbene questo sia formalmente sbagliato. Tale ambiguità ha portato l'*International Electrotechnical Commission* (IEC) a definire nuovi prefissi per multipli binari (che non sono entrati comunque nell'uso comune).

Nome (Simbolo)	Potenza	Nome (Simbolo)	Potenza
Kilobyte (KB)	10^3 (= 1000)	Kibibyte (KiB)	2^{10} (= 1024)
Megabyte (MB)	10^6 (= 1000000)	Mebibyte (MiB)	2^{20} (= 1048576)
Gibabyte (GB)	10^9	Gibibyte (GiB)	2^{30}
Terabyte (TB)	10^{12}	Tebibyte (TiB)	2^{40}
Petabyte (PB)	10^{15}	Pebibyte (PiB)	2^{50}

Rappresentazione dei caratteri

- **CARATTERI**→ Informazioni contenute nei documenti di testo:
 - ❖ cifre, lettere, simboli di punteggiatura;
 - ❖ Simboli speciali: #, %, (,) , & , \$
 - ❖ Caratteri speciali, informazioni di controllo: ritorno a capo, tabulazione, controllo.

Codifica dei caratteri (1)

I caratteri vengono rappresentati mediante codici: si associa ad ogni carattere una sequenza di bit (un numero).

È fondamentale:

- Essere univoci: definire un metodo con cui noi ci scambiamo codifiche in maniera univoca tra tutti i dispositivi con cui ci scambiamo messaggi. Ciò significa poter comunicare in maniera univoca con programmi come Whatsapp tra dispositivi che utilizzano sistemi operativi differenti (Android vs iOS).
- Combattere uno **spreco di memoria** o **ottimizzazione della banda**: produrre applicazioni che siano utilizzabili dall'utente perché la **fruibilità** e **l'inclusività** sono fondamentali.

Codifica dei caratteri (2)

- E quando si parla di codifica parliamo anche di standardizzazione ossia che la lettera A è codificato in maniera standard con una determinata sequenza di bit.
- Esempio: Se copiamo in un post di LinkedIn l'emoticons di un post di Facebook, non viene rappresentata la «faccina» ma il codice. Questo significa che le due piattaforme utilizzano codifiche differenti. In ogni caso, la standardizzazione viene definita da organismi internazionali.
- Affinchè due diversi calcolatori si possano parlare correttamente tra loro è necessario che utilizzino lo stesso codice nella trasmissione dei caratteri ma anche nella memorizzazione.

Codifica dei caratteri (3)

I due principali Standard Internazionali per la codifica dei caratteri sono:

- **ASCII** (American Standard Code for Information Interchange).
- **UNICODE**

La differenza principale tra i due è nel modo in cui codificano il carattere e il numero di bit che usano per ciascuno.

TRADE-OFF: L'uso di più bit consente di utilizzare **più caratteri** a scapito di file più grandi, a **scapito di ottimizzazione di memoria e di banda**

Codifica dei caratteri (in sintesi)

Codifica: significa stabilire come rappresentare simboli (lettere, numeri, emoji...) con sequenze di bit (0 e 1). Per evitare confusione, queste regole devono essere **standardizzate**: la lettera A deve essere codificata sempre nello stesso modo, altrimenti i computer non si capirebbero

→ Lettere

Con lo standard ASCII:

A = 01000001

B = 01000010

→ Emoji

Qui le cose si complicano, perché gli emoji non esistevano negli standard più vecchi.

Oggi si usa **Unicode**, che assegna un codice univoco a ciascun carattere o emoji. Se però una piattaforma (es. Facebook) usa una variante diversa da un'altra (es. LinkedIn), può capitare che copiando un'emoji appaia il codice invece dell'immagine. Questo succede perché le due piattaforme non stanno interpretando il codice allo stesso modo.

Codice ASCII

La sigla ASCII sta per “**American Standard Code for Information Interchange**”. Sviluppato da una commissione dell'American Standard Association. Prima versione 1963, revisionato nel 1967

- Le cifre e i simboli di punteggiatura vengono prima delle lettere.
- Le lettere maiuscole vengono prima delle minuscole.
- Usa 7 bit (quindi rappresenta 128 caratteri).

Funzionamento:

Quando premi un tasto sulla tastiera:

- La tastiera invia un segnale elettrico al computer.
- Il sistema operativo lo traduce nel codice ASCII corrispondente.
- Il programma lo mostra a video come simbolo leggibile.

Codice ASCII

Carattere	Codice decimale	Codice binario (7 bit)
A	65	1000001
B	66	1000010
a	97	1100001
0	48	0110000
!	33	0100001

Quando a schermo si vede la lettera
A il computer sta manipolando il
numero 65

Codice ASCII

La tabella si legge prendendo prima i 3 bit dell'intestazione di colonna, poi i 4 bit dell'intestazione di riga.

Ad esempio il codice ASCII della lettera 'A' è 100 0001 (equivalente a 65 in base 10)

Il carattere composto solamente da 0 (000 0000) corrisponde al concetto di «niente», «nessun carattere».

L'ASCII nasce nel mondo anglosassone. E' sufficiente per lo scambio di caratteri in lingua inglese.

Esprime 128 caratteri ma...?

000	001	010	011	100	101	110	111	
NUL \0	DLE	SP	0	@	P	`	p	0000
SOH	XON	!	1	A	Q	a	q	0001
STX	DC2	"	2	B	R	b	r	0010
ETX	XOFF	#	3	C	S	c	s	0011
EQT	DC4	\$	4	D	T	d	t	0100
ENQ	NAK	%	5	E	U	e	u	0101
ACK	SYN	&	6	F	V	f	v	0110
BEL \a	ETB	'	7	G	W	g	w	0111
BS \b	CAN	(	8	H	X	h	x	1000
HT \t	EM	)	9	I	Y	i	y	1001
LF \n	SUB	*	:	J	Z	j	z	1010
VF \v	ESC	+	;	K	(	k	{	1011
FF \f	FS	,	<	L	\	l		1100
CR \r	GS	-	=	M	)	m	}	1101
SO	RS	.	>	N	^	n	~	1110
SI	US	/	?	O	_	o	DEL	1111

Codice ASCII

La tabella si legge prendendo prima i 3 bit dell'intestazione di colonna, poi i 4 bit dell'intestazione di riga.

Ad esempio il codice ASCII della lettera 'A' è 100 0001 (equivalente a 65 in base 10)

Il carattere composto solamente da 0 (000 0000) corrisponde al concetto di «niente», «nessun carattere».

L'ASCII nasce nel mondo anglosassone. E' sufficiente per lo scambio di caratteri in lingua inglese.

E simboli monetari diversi dal dollaro?

E i caratteri accentati?

000	001	010	011	100	101	110	111	
NUL \0	DLE	SP	0	@	P	`	p	0000
SOH	XON	!	1	A	Q	a	q	0001
STX	DC2	"	2	B	R	b	r	0010
ETX	XOFF	#	3	C	S	c	s	0011
EQT	DC4	\$	4	D	T	d	t	0100
ENQ	NAK	%	5	E	U	e	u	0101
ACK	SYN	&	6	F	V	f	v	0110
BEL \a	ETB	'	7	G	W	g	w	0111
BS \b	CAN	(	8	H	X	h	x	1000
HT \t	EM	)	9	I	Y	i	y	1001
LF \n	SUB	*	:	J	Z	j	z	1010
VF \v	ESC	+	;	K	(	k	{	1011
FF \f	FS	,	<	L	\	l		1100
CR \r	GS	-	=	M	)	m	}	1101
SO	RS	.	>	N	^	n	~	1110
SI	US	/	?	O	_	o	DEL	1111

Codice ASCII

La tabella si legge prendendo prima i 3 bit

000	001	010	011	100	101	110	111	
NUL \0	DLE	SP	0	@	P	`	p	0000
SOH	XON	!	1	A	Q	a	q	0001

- L'ASCII standard (7 bit) ha solo **128 simboli** → niente lettere accentate, niente simboli matematici avanzati, ... niente emoji.
- Se provo a salvare «è» in puro ASCII... **non esiste!**
 - Alcuni sistemi lo sostituivano con un punto interrogativo ?
 - Altri lo mostravano come carattere «strano» es. Ã
- Se un carattere non è definito nello standard ASCII, il computer **non sa come visualizzarlo.**

E i caratteri accentati?

SO	RS	.	>	N	^	n	~	1110
SI	US	/	?	O	_	o	DEL	1111

Codice ASCII Esteso

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
10000000	128	Ç	10100000	160	á	11000000	192	+	11100000	224	Ó
10000001	129	ü	10100001	161	í	11000001	193	-	11100001	225	ß
10000010	130	ë	10100010	162	ó	11000010	194	-	11100010	226	Ô
10000011	131	â	10100011	163	û	11000011	195	+	11100011	227	Õ
10000100	132	ä	10100100	164	ñ	11000100	196	-	11100100	228	ö
10000101	133	à	10100101	165	Ñ	11000101	197	+	11100101	229	Ö
10000110	134	â	10100110	166	ª	11000110	198	ã	11100110	230	µ
10000111	135	ç	10100111	167	º	11000111	199	Ä	11100111	231	þ
10001000	136	ê	10101000	168	¿	11001000	200	+	11101000	232	ð
10001001	137	ë	10101001	169	®	11001001	201	+	11101001	233	Û
10001010	138	è	10101010	170	¬	11001010	202	-	11101010	234	Ü
10001011	139	ï	10101011	171	½	11001011	203	-	11101011	235	Ý
10001100	140	î	10101100	172	¼	11001100	204	-	11101100	236	ÿ
10001101	141	í	10101101	173	¡	11001101	205	-	11101101	237	Ÿ
10001110	142	Ä	10101110	174	«	11001110	206	+	11101110	238	—
10001111	143	Å	10101111	175	»	11001111	207	©	11101111	239	ˆ
10010000	144	È	10110000	176	—	11010000	208	ð	11110000	240	-
10010001	145	Æ	10110001	177	—	11010001	209	Ð	11110001	241	±
10010010	146	Æ	10110010	178	—	11010010	210	Ê	11110010	242	—
10010011	147	ô	10110011	179	—	11010011	211	Ë	11110011	243	¼
10010100	148	õ	10110100	180	—	11010100	212	È	11110100	244	¶
10010101	149	ö	10110101	181	À	11010101	213	É	11110101	245	§
10010110	150	ù	10110110	182	Â	11010110	214	Ê	11110110	246	÷
10010111	151	û	10110111	183	Ã	11010111	215	Ë	11110111	247	˙
10011000	152	ÿ	10111000	184	©	11011000	216	Ë	11111000	248	˚
10011001	153	Ö	10111001	185	—	11011001	217	+	11111001	249	ˆ
10011010	154	Ü	10111010	186	—	11011010	218	+	11111010	250	˙
10011011	155	Ÿ	10111011	187	+	11011011	219	—	11111011	251	1
10011100	156	£	10111100	188	+	11011100	220	—	11111100	252	3
10011101	157	Ø	10111101	189	¢	11011101	221	—	11111101	253	2
10011110	158	×	10111110	190	¥	11011110	222	Ë	11111110	254	—
10011111	159	f	10111111	191	+	11011111	223	—	11111111	255	—

8 bit

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
00000000	0	Null	00100000	32	SpC	01000000	64	@	01100000	96	`
00000001	1	Start of heading	00100001	33	!	01000001	65	A	01100001	97	a
00000010	2	Start of text	00100010	34	"	01000010	66	B	01100010	98	b
00000011	3	End of text	00100011	35	#	01000011	67	C	01100011	99	c
00000100	4	End of transmit	00100100	36	\$	01000100	68	D	01100100	100	d
00000101	5	Enquiry	00100101	37	%	01000101	69	E	01100101	101	e
00000110	6	Acknowledge	00100110	38	&	01000110	70	F	01100110	102	f
00000111	7	Audible bell	00100111	39	'	01000111	71	G	01100111	103	g
00001000	8	Backspace	00101000	40	(	01001000	72	H	01101000	104	h
00001001	9	Horizontal tab	00101001	41	)	01001001	73	I	01101001	105	i
00001010	10	Line feed	00101010	42	*	01001010	74	J	01101010	106	j
00001011	11	Vertical tab	00101011	43	+	01001011	75	K	01101011	107	k
00001100	12	Form Feed	00101100	44	,	01001100	76	L	01101100	108	l
00001101	13	Carriage return	00101101	45	-	01001101	77	M	01101101	109	m
00001110	14	Shift out	00101110	46	.	01001110	78	N	01101110	110	n
00001111	15	Shift in	00101111	47	/	01001111	79	O	01101111	111	o
00010000	16	Data link escape	00110000	48	0	01010000	80	P	01110000	112	p
00010001	17	Device control 1	00110001	49	1	01010001	81	Q	01110001	113	q
00010010	18	Device control 2	00110010	50	2	01010010	82	R	01110010	114	r
00010011	19	Device control 3	00110011	51	3	01010011	83	S	01110011	115	s
00010100	20	Device control 4	00110100	52	4	01010100	84	T	01110100	116	t
00010101	21	Neg. acknowledge	00110101	53	5	01010101	85	U	01110101	117	u
00010110	22	Synchronous idle	00110110	54	6	01010110	86	V	01110110	118	v
00010111	23	End trans. block	00110111	55	7	01010111	87	W	01110111	119	w
00011000	24	Cancel	00111000	56	8	01011000	88	X	01111000	120	x
00011001	25	End of medium	00111001	57	9	01011001	89	Y	01111001	121	y
00011010	26	Substitution	00111010	58	:	01011010	90	Z	01111010	122	z
00011011	27	Escape	00111011	59	;	01011011	91	[	01111011	123	{
00011100	28	File separator	00111100	60	<	01011100	92	\	01111100	124	
00011101	29	Group separator	00111101	61	=	01011101	93	]	01111101	125	}
00011110	30	Record Separator	00111110	62	>	01011110	94	^	01111110	126	~
00011111	31	Unit separator	00111111	63	?	01011111	95	_	01111111	127	Del

Codice ASCII Esteso

Nel codice ASCII esteso, le lettere A,B,C sono rappresentate dalle seguenti configurazioni:

A→01000001 corrispondente al decimale 65

B→01000010 corrispondente al decimale 66

C→01000011 corrispondente al decimale 67

- l'ordine alfabetico corrisponde a numeri consecutivi in ASCII
- I primi 32 byte della tabella standard sono riservati per segnali di controllo e funzioni varie
 - I primi 32 valori (0–31) non rappresentano lettere o numeri, ma caratteri di controllo.
 - Servivano nei vecchi terminali e stampanti per funzioni come:
 - 0 → NULL (nessun carattere)
 - 7 → BEL (suono del campanello)
 - 8 → BACKSPACE (cancella indietro)
 - 10 → LINE FEED (vai a capo)
 - 13 → CARRIAGE RETURN (ritorno carrello, come le macchine da scrivere)

Codice ASCII & ASCII Esteso

- ASCII si basa su successioni di **7 bit**, $2^7=128$ caratteri, numeri, codici di controllo, segni di interpunzione;
- ASCII esteso utilizza tutti gli **8 bit** disponibili in un byte ($2^8=256$)
- **Vantaggi:** ASCII è un codice antico e molto sicuro
- **Limitazioni:** I **caratteri internazionali di numerose** lingue europee **non sono rappresentabili**, per non parlare delle **lingue asiatiche** per le quali il numero di simboli è elevatissimo.

Considerazioni su ASCII & ASCII Esteso

A che cosa serve l'ASCII? (1)

Scrivere e memorizzare testo

Quando si digita una lettera sulla tastiera, il PC non salva la lettera "A" ma il codice ASCII 65. In memoria: 01000001 (8 bit). Sullo schermo, il sistema operativo legge quel codice e lo mostra come A.

Comunicazione tra dispositivi

ASCII è stato creato per permettere ai computer e alle periferiche (stampanti, terminali, modem) di parlare la stessa lingua.

Esempio: se mando la stringa "CIAO" a una stampante, in realtà sto inviando la sequenza di codici ASCII 67 73 65 79.

Considerazioni su ASCII & ASCII Esteso

A che cosa serve l'ASCII? (esempio)

Se scrivo la parola HI e la salvo in un file:

H = ASCII 72 = binario 01001000

I = ASCII 73 = binario 01001001

Nel file non ci sono le lettere, ma la sequenza di byte:

01001000 01001001

Se il programma attivo è Word allora si mostra HI sul foglio bianco

Codifica ISO 8859

ISO 8859 – Serie di standard promossi dall'**I**nternational **S**tandard **O**rganization (ISO) e dall'International Elechtrotechnical Commision (IEC) per la codifica di caratteri su **8 bit**.

- ISO è un'organizzazione internazionale indipendente e non governativa alla quale aderiscono 167 enti di standardizzazione nazionali
- **Nasce per rimediare al problema della necessità di caratteri aggizionali**
- Aggiunge un ulteriore bit a sinistra, passando dai 7 bit dell'ASCII (non esteso) a 8 bit e raddoppiando le rappresentazioni da 128 (2^7) a 256 (2^8)

Osservazioni...?

Codifica ISO 8859

ISO 8859 – Serie di standard promossi dall'**I**nternational **S**tandard **O**rganization (ISO) e dall'International Elechtrotechnical Commision (IEC) per la codifica di caratteri su **8 bit**.

- ISO è un'organizzazione internazionale indipendente e non governativa alla quale aderiscono 167 enti di standardizzazione nazionali
- **Nasce per rimediare al problema della necessità di caratteri aggizionali**
- Aggiunge un ulteriore bit a sinistra, passando dai 7 bit dell'ASCII (non esteso) a 8 bit e raddoppiando le rappresentazioni da 128 (2^7) a 256 (2^8)

Osservazioni...? Non bastava
l'ASCII esteso?

Codifica ISO 8859

ISO 8859 – Serie di standard promossi dall'**I**nternational **S**tandard **O**rganization (ISO) e dall'International Elechtrotechnical Commision (IEC) per la codifica di caratteri su **8 bit**.

- ISO è un'organizzazione internazionale indipendente e non governativa alla quale aderiscono 167 enti di standardizzazione nazionali
- **Nasce per rimediare al problema della necessità di caratteri addizionali**
- Aggiunge un ulteriore bit a sinistra, passando dai 7 bit dell'ASCII (non esteso) a 8 bit e raddoppiando le rappresentazioni da 128 (2^7) a 256 (2^8)

Osservazioni...? Non bastava
l'ASCII esteso?

In altre lingue servono
lettere accentate o caratteri
speciali e non solo! Ogni
lingua vuole simboli diversi

Codifica ISO 8859

- ISO 8859-1 west European languages (Latin-1)
- ISO 8859-2 east European languages (Latin-2)
- ISO 8859-3 southeast European languages (Latin-3)
- ISO 8859-4 Scandinavian/Baltic languages (Latin-4)
- ISO 8859-5 Latin/Cyrillic
- ISO 8859-6 Latin/Arabic
- ISO 8859-7 Latin/Greek
- ISO 8859-8 Latin/Hebrew
- ISO 8859-9 Latin-1 modification for Turkish (Latin-5)
- ISO 8859-10 Lappish/Nordic/Eskimo languages (Latin-6)
- ISO 8859-11 Latin/Thai
- ~~ISO 8859-12 Latin/Devanagari~~
- ISO 8859-13 Baltic Rim languages (Latin-7)
- ISO 8859-14 Celtic (Latin-8)
- ISO 8859-15 west European languages (Latin-9)

Per questo l'**ISO** ha definito una **serie di standard** chiamata **ISO/IEC 8859**, ognuno pensato per un gruppo di lingue.

Codifica ISO 8859

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
NUL \0	DLE	SP	0	@	P	`	p			NBS	°	À	Đ	à	đ	0000
SOH	XON	!	1	A	Q	a	q			ı	±	Á	Ñ	á	ñ	0001
STX	DC2	"	2	B	R	b	r			ç	²	Â	Ò	â	ò	0010
ETX	XOFF	#	3	C	S	c	s			£	³	Ã	Ó	ã	ó	0011
EQT	DC4	\$	4	D	T	d	t			¤	´	Ä	Ô	ä	ô	0100
ENQ	NAK	%	5	E	U	e	u			¥	µ	Å	Õ	å	õ	0101
ACK	SYN	&	6	F	V	f	v			¦	¶	Æ	Ö	æ	ö	0110
BEL \a	ETB	'	7	G	W	g	w			§	*	Ç	×	ç	÷	0111
BS \b	CAN	(	8	H	X	h	x			"	,	È	Ø	è	ø	1000
HT \t	EM	)	9	I	Y	i	y			©	ı	É	Ù	é	ù	1001
LF \n	SUB	*	:	J	Z	j	z			ª	º	Ê	Ú	ê	ú	1010
VF \v	ESC	+	;	K	(	k	{			«	»	Ë	Û	ë	û	1011
FF \f	FS	,	<	L	\	l	l			¬	¼	Ì	Ü	ì	ü	1100
CR \r	GS	-	=	M	)	m	}			SH	½	Í	Ý	í	ý	1101
SO	RS	.	>	N	^	n	~			®	¾	Î	Þ	î	þ	1110
SI	US	/	?	O	_	o	DEL			¯	¿	Ï	ß	ï	ÿ	1111

Codifica ISO 8859

L'ISO 8859-15 è un aggiornamento dell'ISO 8859-1 per aggiungere il simbolo dell'Euro e completare la copertura di francese, finnico e estone.

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
NUL \0	DLE	SP	0	@	P	`	p			NBS	°	À	Đ	à	đ	0000
SOH	XON	!	1	A	Q	a	q			ı	±	Á	Ñ	á	ñ	0001
STX	DC2	"	2	B	R	b	r			ç	²	Â	Ò	â	ò	0010
ETX	XOFF	#	3	C	S	c	s			£	³	Ã	Ó	ã	ó	0011
EQT	DC4	\$	4	D	T	d	t			€	Ž	Ä	Ô	ä	ö	0100
ENQ	NAK	%	5	E	U	e	u			¥	μ	Å	Õ	å	õ	0101
ACK	SYN	&	6	F	V	f	v			Š	¶	Æ	Ö	æ	ö	0110
BEL \a	ETB	'	7	G	W	g	w			§	*	Ç	×	ç	÷	0111
BS \b	CAN	(	8	H	X	h	x			š	ž	È	Ø	è	ø	1000
HT \t	EM	)	9	I	Y	i	y			©	ı	É	Ù	é	ù	1001
LF \n	SUB	*	:	J	Z	j	z			ª	º	Ê	Ú	ê	ú	1010
VF \v	ESC	+	;	K	(	k	{			«	»	Ë	Û	ë	û	1011
FF \f	FS	,	<	L	\	l				¬	œ	Ì	Ü	ì	ü	1100
CR \r	GS	-	=	M	)	m	}			SH	œ	Í	Ý	í	ý	1101
SO	RS	.	>	N	^	n	~			®	ÿ	Î	Þ	î	þ	1110
SI	US	/	?	O	_	o	DEL			-	¿	İ	ß	ï	ÿ	1111

Codifica ISO 8859

L'ISO 8859-15 è un aggiornamento dell'ISO 8859-1 per aggiungere il simbolo dell'Euro e completare la copertura di francese, finnico e estone.

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
NUL \0	DLE	SP	0	@	P	`	p			NBS	°	À	Đ	à	đ	0000
SOH	XON	!	1	A	Q	a	q			ı	±	Á	Ñ	á	ñ	0001
STX	DC2	"	2	B	R	b	r			ç	²	Â	Ò	â	ò	0010
ETX	XOFF	#	3	C	S	c	s			£	³	Ã	Ó	ã	ó	0011
EQT	DC4	\$	4	D	T	d	t			€	Ž	Ä	Ô	ä	ö	0100
ENQ	NAK	%	5	E	U	e	u			¥	µ	Å	Õ	å	õ	0101
ACK	SYN	&	6	F	V	f	v			Š	¶	Æ	Ö	æ	ö	0110
BEL \a	ETB	'	7	G	W	g	w			§	*	Ç	×	ç	÷	0111
BS \b	CAN	(	8	H	X	h	x			š	ž	È	Ø	è	ø	1000
HT \t	EM	)	9	I	Y	i	y			©	ı	É	Ù	é	ù	1001
LF \n	SUB	*	:	J	Z	j	z			ª	º	Ê	Ú	ê	ú	1010
VF \v	ESC	+	;	K	(	k	{			«	»	Ë	Û	ë	û	1011
FF \f	FS	,	<	L	\	l				¬	œ	Ì	Ü	ì	ü	1100
CR \r	GS	-	=	M	)	m	}			SH	œ	Í	Ý	í	ý	1101
SO	RS	.	>	N	^	n	~			®	ÿ	Î	Þ	î	þ	1110
SI	US	/	?	O	_	o	DEL			-	¿	İ	Ɓ	ï	ÿ	1111

Verso Unicode

Il problema con ISO 8859

Ogni ISO 8859-x copriva solo un gruppo di lingue

- ISO 8859-1 → lingue dell'Europa occidentale
- ISO 8859-2 → lingue dell'Europa orientale
- ISO 8859-5 → cirillico
- ISO 8859-6 → arabo (...)

Risultato: servivano tabelle diverse a seconda della lingua → **scarsa compatibilità internazionale**

Un testo in francese (é) salvato in ISO 8859-1 → appare corrotto se letto con ISO 8859-5. Impossibile gestire in un unico documento più lingue (es. inglese, russo e arabo insieme).

Un **unico standard universale**, con codici per **tutti i caratteri** di tutte le lingue del mondo (e anche emoji, simboli matematici, ecc.).

Supera la frammentazione degli ISO.

Standard Unicode (ISO 10646)

Unicode è uno standard internazionale che assegna a ogni simbolo (lettera, numero, emoji, segno matematico...) un codice univoco. Questo codice è scritto in forma U+xxxx (esadecimale).

A:A B:B C:C D:D E:E

U+1F600 : 😄	U+263A : 😊	U+1F61F : 😏	U+1F61E : 😎
U+1F603 : 😆	U+1F61A : 😇	U+1F641 : 😡	U+1F648 : 🙈
U+1F604 : 😅	U+1F619 : 😆	U+2639 : 😖	U+1F649 : 🙉
U+1F601 : 😄	U+1F60B : 😊	U+1F62E : 😏	U+1F64A : 🙊
U+1F606 : 😌	U+1F61B : 😇	U+1F62F : 😏	U+1F44B : 🤝
U+1F605 : 😄	U+1F61C : 😇	U+1F632 : 😏	U+1F91A : 🤝
U+1F923 : 😄	U+1F92A : 😄	U+1F633 : 😏	U+1F590 : 🤝
U+1F602 : 😄	U+1F61D : 😇	U+1F626 : 😏	U+270B : 🤝
U+1F642 : 😊	U+1F911 : 🤝	U+1F627 : 😏	U+1F596 : 🤝
U+1F643 : 😊	U+1F917 : 🤝	U+1F628 : 😏	U+1F44C : 🤝
U+1F609 : 😊	U+1F92D : 🤝	U+1F630 : 😏	U+270C : 🤝
U+1F60A : 😊	U+1F92B : 🤝	U+1F625 : 😏	U+1F91E : 🤝
U+1F607 : 😊	U+1F914 : 🤝	U+1F622 : 😏	U+1F91F : 🤝
U+1F60D : 😊	U+1F60E : 😊	U+1F62D : 😏	U+1F918 : 🤝
U+1F929 : 😊	U+1F913 : 🤝	U+1F631 : 😏	U+1F919 : 🤝
U+1F618 : 😊	U+1F9D0 : 🤝	U+1F616 : 😏	U+1F44D : 🤝
U+1F617 : 😊	U+1F615 : 😊	U+1F623 : 😏	U+1F44E : 🤝

Unicode assegna i codici ma poi bisogna rappresentarli in memoria: tradurli in sequenze di byte.

Qui entrano in gioco le **codifiche Unicode**: UTF-8, UTF-16, UTF-32

Standard Unicode (ISO 10646)

Unicode è uno standard internazionale che assegna a ogni simbolo (lettera, numero, emoji, segno matematico...) un codice univoco. Questo codice è scritto in forma U+xxxx (esadecimale).

UTF-8

- Codifica variabile (1–4 byte).
- Caratteri ASCII (inglese) (1 byte) → compatibile con i vecchi sistemi.
- Lettere accentate o simboli (2–3 byte).
- Emoji e caratteri molto speciali (4 byte).

UTF-16

- Usa 2 byte per la maggior parte dei caratteri e 4 per i più rari

UTF-816

- Usa sempre 4 byte per ogni carattere.

Standard Unicode (ISO 10646)

Unicode è uno standard internazionale che assegna a ogni simbolo (lettera, numero, emoji, segno matematico...) un codice univoco. Questo codice è scritto in forma U+xxxx (esadecimale).



Codice Unicode: U+1F603

- In UTF-8 → F0 9F 98 83 (**4 byte**)
- In UTF-16 → D83D DE03 (**2 + 2 byte, [4 cifre esadecimali=16 bit=2 byte]**)
- In UTF-32 → 0001F603 (**4 byte, [8 cifre esadecimali=32 bit=4 byte]**)

Emoji e caratteri molto speciali (4 byte).

Decimale	Binario (4 bit)	Esadecimale
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Standard Unicode (ISO 10646)

- Con ASCII/ISO = tanti alfabeti separati.
- Con Unicode/UTF-8 = un solo alfabeto universale → vero multilinguismo.

Zoom su Unicode Transfer Format-8 (UTF-8)

In **UTF-8** un carattere può occupare **1, 2, 3 o 4 byte**.

Regola

- Se il primo bit è 0 → è un carattere ASCII → usa 1 byte solo.
- Se il primo byte inizia con 110 → il carattere usa 2 byte.
- Se inizia con 1110 → usa 3 byte.
- Se inizia con 11110 → usa 4 byte.

Esempio:

- A (ASCII, U+0041) → 01000001
- è (U+00E9) → 11000011 10101001 (il primo byte che comincia con 11-0 anticipa la presenza del secondo)
- € (U+20AC) → 11100010 10000010 10101100 (il primo byte che comincia con 111-0 anticipa la presenza del secondo e del terzo byte)
- 🤪 (U+1F603) → 11110000 10011111 10011000 10000011 (il primo byte che comincia con 1111-0 anticipa la presenza del secondo, del terzo e del quarto byte)

Testo

Il testo scritto è formato non solo da caratteri dell'alfabeto (necessari per formare le parole) ma anche da altri simboli necessari per rendere comprensibile il testo

In particolare in un testo possiamo distinguere:

- I limitatori di parola (spazi bianchi).
- I limitatori di frase (segni di interpunzione).
- I limitatori di pagina (margini).
- I simboli speciali (virgolette, simboli matematici ecc.).

La disposizione e l'aspetto del testo (formattazione)

Testo

- Per passare da un testo (in linguaggio naturale) a un testo elettronico (in binario) è necessario un processo di codifica che deve essere composto da due parti:
 - La codifica dei caratteri e dei simboli grafici
 - La codifica delle operazioni necessarie per esplicitare l'interpretazione del testo (markup)

Testo

I Vantaggi di un testo elettronico

- Le operazioni di manipolazione (modifica, cancellazione ecc.) di un testo elettronico sono molto semplici e veloci.
- Le ricerche effettuate sono più veloci e precise.
- La conservazione delle informazioni è più sicura, grazie alla possibilità di effettuare delle copie dei file che contengono il testo.
- Nel caso di un testo elettronico la condivisione e lo scambio di informazioni viene agevolato rispetto ad un testo tradizionale (cartaceo).

Testo

Acquisizione di un testo

- Per trattare elettronicamente il testo è necessario acquisirlo, ovvero fornirlo come input al computer.
- Per acquisire il testo si può operare in diversi modi:
- Digitare il testo tramite la tastiera.
- Usare uno scanner e un software di OCR (Optical Character Recognition),
- Copiare il testo precedentemente acquisito o creato da altri tramite la copia di file memorizzati su qualche tipo di supporto (floppy disk, CD-ROM ecc.).