Ultimo aggiornamento: November 1, 2026

🗺️ Percorso di quattro anni · Anno 1, Primavera

Informazioni sul corso

Questo corso gratuito ha 9 unità e 4 laboratori. Sono i totali reali, non un programma più lungo accorciato.

Un controllo è un numero o un risultato di codice in una pagina del corso che uno script ricalcola prima della pubblicazione. Il numero stampato e quello ricalcolato devono coincidere.

> Questo è lo schema di un corso gratuito di autoapprendimento scritto da Saving Planets. Non è un corso universitario, non dà crediti accademici e nessuna scuola è coinvolta.
>
> I contenuti seguono i corsi di programmazione del primo anno come MIT 6.100L (già 6.0001), CS50 di Harvard, CS106A di Stanford e CS 1371 di Georgia Tech per ingegneri. Rimandiamo ovunque a materiali gratuiti. Nessuna di queste organizzazioni approva questa pagina.
>
> Cosa aggiungiamo: ogni esempio è vero codice Python che il nostro script di build esegue prima della pubblicazione della pagina, e l'output stampato deve corrispondere esattamente a quello che vedi qui. I dati provengono da vere fonti pubbliche: la serie di CO_2 di Mauna Loa della NOAA, l'analisi delle temperature GISTEMP della NASA e gli idrometri dell'USGS.
>
> Non serve installare nulla: usa Google Colab o JupyterLite, che esegue Python nel browser con Pyodide. I tuoi progressi sono salvati solo in questo browser.

Come questo corso aiuta a salvare il pianeta

Le scienze del clima e dell'ambiente si basano sul codice. La curva della CO_2, le serie di temperatura globale e le previsioni dei fiumi sono tutte prodotte da programmi che leggono dati pubblici, e tu puoi leggere gli stessi file.

Ogni unità termina con una breve sezione con fonti che mostra come le sue idee aiutano le persone e il pianeta.

Per gli insegnanti

Puoi insegnare questo corso, oppure lasciare che uno studente lo faccia da solo. È gratis. Saving Planets non dà un voto né crediti scolastici.

- Questa pagina è l'intero corso: 9 unità e 4 laboratori (ENGR 130).
- Prevedi 15 settimane, circa 6–9 ore a settimana, comprese le settimane di laboratorio.
- Segui il calendario di 15 settimane. Leggi l'unità, fai la pratica su carta, poi apri la risposta. Nella settimana di laboratorio, fai il laboratorio o la versione a casa.
- Un controllo è un numero o un risultato di codice nella pagina che uno script ricalcola prima della pubblicazione. Il numero che vedi e quello ricalcolato devono coincidere.
- Usa Stampa questa pagina, oppure scarica la guida di studio PDF. Il formulario è verso la fine.
- Chiedi: in che modo questa unità aiuta le persone e il pianeta? Ogni unità ha un riquadro verde con un fatto e la sua fonte.
- Copia il modello di guida per insegnanti e scrivi il nome della classe, il tempo che hai e le domande che farai. Apri il modello di guida per insegnanti

Prima di iniziare

Prerequisiti

- Algebra delle superiori.
- Nessuna esperienza di programmazione richiesta.
- Fisica I e Analisi I aiutano per l'unità di simulazione, ma non sono obbligatori.

Come studiare

- Digita tu ogni esempio invece di incollarlo; le dita imparano la sintassi.
- Prevedi l'output prima di eseguire il codice.
- Quando sei bloccato, leggi il messaggio di errore dal basso verso l'alto.
- Cambia una cosa alla volta e riesegui.
- Svolgi ogni esercizio prima di aprire la risposta.
- Rifai i quiz qualche giorno dopo. La pratica distribuita batte una lunga sessione unica.

Che cosa ti serve

- Un computer o tablet con un browser web moderno.
- Un account Google gratuito per Colab, oppure JupyterLite senza alcun account.
- Facoltativo: Python 3 installato localmente con NumPy e Matplotlib.

Programma suggerito di 15 settimane

Circa 6-9 ore a settimana: lettura, esercizi e, nelle settimane di laboratorio, il laboratorio.

Settimana	Unità	Argomenti	Lettura
1	Primi passi: variabili, tipi ed espressioni	Notebook, variabili, tipi, espressioni	Tutorial Python §3

Settimana	Unità	Argomenti	Lettura
2	Controllo del flusso: condizioni e cicli	Condizioni e cicli	Tutorial Python §4
3	Funzioni	Funzioni	MIT 6.100L
4	Liste, dizionari e stringhe	Liste, dizionari, stringhe	Tutorial Python §5
5	File e dati CSV	File e CSV	Documentazione del modulo csv
6	File e dati CSV	Dati reali con errori; laboratorio A	NOAA GML
7	NumPy e Matplotlib	Array NumPy; esercizi per l'esame intermedio	Basi di NumPy
8	NumPy e Matplotlib	Matplotlib; laboratorio B	Matplotlib pyplot
9	Debug e test	Debug e test; laboratorio C	Documentazione di pytest
10	Algoritmi e complessità	Ricerca e complessità	MIT 6.100L
11	Algoritmi e complessità	Bisezione e metodi numerici	MIT 6.100L
12	Simulazione numerica	Simulazione di un proiettile	Fisica I
13	Simulazione numerica	Orbite e popolazioni	Analisi II
14	Simulazione numerica	Resistenza e convergenza; laboratorio D	Introduzione all'ingegneria aerospaziale
15	Ripasso	Ripasso ed esercitazione finale	Tutte le unità

I tuoi progressi

Spunta unità e laboratori man mano che li finisci, e fai i quiz. Progressi e punteggi sono salvati solo in questo browser e non ci vengono mai inviati.

Unità

Primi passi: variabili, tipi ed espressioni

Esegui il tuo primo Python in un notebook nel browser, memorizza valori in variabili e calcola con numeri e testo.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Aprire un notebook Colab o JupyterLite ed eseguire le celle.
- Usare variabili e i tipi int, float, str e bool.
- Usare gli operatori aritmetici, inclusi //, % e **.
- Formattare l'output con le f-string.

Concetti chiave

- Un programma è una sequenza di istruzioni. Un notebook permette di eseguirle una cella alla volta e vedere subito i risultati.
- Una variabile è un nome legato a un valore: mass_kg = 1100. Scegli nomi che dicano cosa e in quali unità.
- Ogni valore ha un tipo. int (numeri interi), float (decimali), str (testo) e bool (True/False) sono i tipi di base; type() ti dice quale.
- / dà sempre un float; // è la divisione intera; % è il resto; ** è una potenza.
- I float sono approssimazioni binarie: 0.1 + 0.2 non è esattamente 0.3, quindi confronta i float con una tolleranza.

Equazioni chiave

$$W = mg$$

`W=mg`

L'esempio 1.1 calcola un peso

$$\text{change} = \frac{x_{new} - x_{old}}{x_{old}}$$

`change=_new-x_oldx_old`

Variazione relativa (esempio 1.2)

Esempi svolti passo passo

Esempio 1.1 Calcola il peso di un aereo da 1100 kg e controlla i tipi.

```
mass_kg = 1100
g = 9.81
weight = mass_kg * g
print(weight)
print(type(mass_kg), type(weight))
```

1. mass_kg è un int; moltiplicarlo per il float g dà un float.

Output

```
10791.0
<class 'int'> <class 'float'>
```

Risposta: 10791 N, un float.

Esempio 1.2 Di quanto è aumentata la CO_2 di Mauna Loa dalla media annua del 1959 a quella del 2025 (NOAA)?

```
ppm_1959 = 315.98
ppm_2025 = 427.35
rise = ppm_2025 - ppm_1959
print(f"CO2 rose {rise:.2f} ppm, or {rise / ppm_1959:.1%}")
```

1. Sottrai, poi dividi per il valore iniziale.
2. La f-string formatta con :.2f (2 decimali) e :.1% (percentuale).

Output

```
CO2 rose 111.37 ppm, or 35.2%
```

Risposta: 111.37 ppm, un aumento del 35.2%.

Esempio 1.3 Esplora gli operatori e le sorprese dei float.

```
print(7 / 2, 7 // 2, 7 % 2, 2 ** 10)
print(0.1 + 0.2 == 0.3, abs((0.1 + 0.2) - 0.3) < 1e-9)
print(int("42") + 1, str(42) + "1")
```

1. 7/2 = 3.5, 7//2 = 3, 7%2 = 1.
2. 0.1 + 0.2 == 0.3 è False; un test con tolleranza è True.
3. int("42") converte testo in numero; str(42) + "1" concatena testo.

Output

```
3.5 3 1 1024
False True
43 421
```

Risposta: Usa tolleranze per i float e converti i tipi di proposito.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Che cosa dà 17 % 5?

   2.

2. Di che tipo è 3 * 2.0?

   float (6.0).

3. Scrivi una f-string che stampi pi con 3 decimali.

   f"{math.pi:.3f}" → 3.142 (dopo import math).

4. Perché mass_kg è un nome migliore di m?

   Indica la grandezza e la sua unità, evitando errori di unità.

Errori concettuali comuni

Mito: = significa «uguale».
In realtà: In Python = assegna; == confronta.

Mito: 0.1 + 0.2 == 0.3 è True.
In realtà: È False a causa dell'arrotondamento binario.

Mito: "5" + 5 dà 10.
In realtà: È un TypeError; converti con int("5").

Flashcard

Variabile	**int / float**
Nome legato a un valore.	Numero intero / numero decimale.

str

Testo.

bool

True o False.

f-string

Formattazione f"{x:.2f}".

//, %

Divisione intera, resto.

Quiz di autoverifica

1. 7 // 2 =

3.5

3

4

1

3. Divisione intera.

2. type(2.0) è...

int

float

str

bool

float. Ha un punto decimale.

3. 2 ** 10 =

20

100

1024

12

1024. Potenza.

4. L’assegnazione in Python usa...

==

=

:=only

->

=. = lega un nome.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

I numeri dell’esempio 1.2 vengono direttamente dal Global Monitoring Laboratory della NOAA, che pubblica la serie di CO_2 di Mauna Loa come file CSV gratuito. Poche righe di Python la trasformano nei fatti chiave della scienza del clima.

Su Saving Planets: Analisi I

Fonti: NOAA GML: Trends in atmospheric CO_2, data (https://gml.noaa.gov/ccgg/trends/data.html) · Google Colab (notebook gratuiti) (https://colab.research.google.com/)

Risorse di apprendimento gratuite

- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Sezioni 3 e 4.
- Google Colab (notebook gratuiti) (https://colab.research.google.com/): Avvia un notebook.
- JupyterLite: Python nel browser (Pyodide) (https://jupyter.org/try-jupyter/lab/): Opzione senza account.

☐ **Ho finito questa unità**

Controllo del flusso: condizioni e cicli

Far decidere i programmi con if/elif/else e ripetere con i cicli for e while.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Scrivere if/elif/else con operatori di confronto e booleani.
- Iterare su intervalli e sequenze con for.
- Usare cicli while con una condizione di arresto chiara.
- Accumulare totali e conteggi.

Concetti chiave

- L'indentazione definisce i blocchi in Python. Lo standard è quattro spazi per livello.
- I confronti (<, >=, ==, !=) producono booleani; combinali con and, or, not.
- for x in sequence si ripete una volta per elemento; range(a, b) dà a, a+1, ..., b−1.
- while si ripete finché la sua condizione non è False. Assicurati che qualcosa nel ciclo la modifichi, altrimenti non finisce mai.
- Lo schema dell'accumulatore: inizia un totale da 0 e somma dentro il ciclo.

Equazioni chiave

$$\sum_{n=1}^{100} n = \frac{100 \cdot 101}{2} = 5050$$

_n=1^100n=(100·101)/(2)=5050

La somma di Gauss verifica l'esempio 2.2

$$x_{k+1} = x_k + \Delta$$

x_k+1=x_k+Δ

Un ciclo che fa avanzare un valore

Esempi svolti passo passo

Esempio 2.1 Classifica l'anomalia di temperatura globale 2024 della NASA.

```
anomaly = 1.28   # NASA GISTEMP, 2024, deg C vs 1951-1980
if anomaly >= 1.0:
    print("at least 1.0 C above the 1951-1980 average")
```

```
elif anomaly > 0:
    print("warmer than average")
else:
    print("at or below average")
```

1. 1.28 ≥ 1.0, quindi viene eseguito il primo ramo e gli altri vengono saltati.

Output

```
at least 1.0 C above the 1951-1980 average
```

Risposta: I rami sono valutati in ordine; viene eseguito solo il primo vero.

Esempio 2.2 Somma i numeri da 1 a 100 con un ciclo.

```
total = 0
for n in range(1, 101):
    total += n
print(total)
```

1. range(1, 101) si ferma prima di 101.
2. total += n aggiunge ogni numero.

Output

```
5050
```

Risposta: 5050, in accordo con la formula di Gauss.

Esempio 2.3 Se la CO_2 continuasse a crescere di 2.5 ppm all'anno dalla media del 2025, quando supererebbe 450 ppm?

```
ppm = 427.35   # 2025 annual mean, Mauna Loa
year = 2025
while ppm < 450:
    ppm += 2.5
    year += 1
print(year, round(ppm, 2))
```

1. Il ciclo aggiunge 2.5 ppm e un anno finché ppm ≥ 450.

Output

```
2035 452.35
```

Risposta: Nel 2035, a 452.35 ppm in questo semplice scenario a tasso costante (non una previsione).

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Quante volte viene eseguito for i in range(3, 10, 2)?

   4 volte (3, 5, 7, 9).

2. Scrivi un ciclo che conti i numeri dispari minori di 20.

   count = sum(1 for n in range(20) if n % 2): 10.

3. Cosa c'è di sbagliato in: while x < 10: print(x)

   x non cambia mai, quindi il ciclo è infinito.

Errori concettuali comuni

Mito: range(1, 10) include 10.
In realtà: Si ferma a 9.

Mito: else dopo if è obbligatorio.
In realtà: È facoltativo.

Mito: L'indentazione è solo stile.
In realtà: In Python definisce la struttura del programma.

Flashcard

if / elif / else

Scegliere un ramo.

for

Ripetere per ogni elemento.

while

Ripetere finché è vero.

range(a, b, s)

Da a fino a b−1, con passo s.

Accumulatore

break / continue

total += x in un ciclo.	Uscire dal ciclo / passare al giro successivo.

Quiz di autoverifica

1. list(range(2, 6)) è...

[2, 3, 4, 5, 6]

[2, 3, 4, 5]

[3, 4, 5, 6]

[2, 6]

[2, 3, 4, 5]. Si ferma prima di 6.

2. Quale parola chiave aggiunge un altro test dopo if?

else if

elif

then

case

elif. elif.

3. Un ciclo while si ferma quando...

la sua condizione è False

dopo 10 volte

mai

a un print

la sua condizione è False. La condizione viene verificata a ogni giro.

4. not (True and False) è...

True

False

None

un errore

True. True and False è False.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

Soglie come 1.0 °C o 450 ppm sono proprio ciò che i programmi di monitoraggio verificano con istruzioni if, segnalando automaticamente record e allarmi all'arrivo di nuovi dati.

Su Saving Planets: Fisica I

Fonti: NASA GISS: GISTEMP v4 surface temperature analysis (https://data.giss.nasa.gov/gistemp/) · NOAA GML: Trends in atmospheric CO_2, data (https://gml.noaa.gov/ccgg/trends/data.html)

Risorse di apprendimento gratuite

- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Sezione 4: controllo del flusso.
- MIT OpenCourseWare 6.100L: Introduction to CS and Programming Using Python (https://ocw.mit.edu/courses/6-100l-introduction-to-cs-and-programming-using-python-fall-2022/): Lezioni su diramazioni e iterazione.

☐ **Ho finito questa unità**

Funzioni

Racchiudere la logica in funzioni riutilizzabili e testabili, con parametri, valori predefiniti e valori di ritorno.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Definire funzioni con def, parametri e return.
- Usare argomenti predefiniti e per parola chiave.
- Scrivere docstring.
- Capire l'ambito locale.

Concetti chiave

- Una funzione dà un nome a un pezzo di logica. Scrivila una volta, testala una volta, usala ovunque.
- I parametri ricevono valori; return restituisce un risultato. Una funzione senza return restituisce None.
- Gli argomenti predefiniti (cl_max=1.6) accorciano i casi comuni; gli argomenti per parola chiave rendono leggibili le chiamate.
- Le variabili create in una funzione sono locali e spariscono al ritorno.
- Una docstring (la stringa sulla prima riga) documenta cosa fa la funzione.

Equazioni chiave

$$F = \frac{9}{5}C + 32$$

F=95C+32

Esempio 3.1

$$V_s = \sqrt{\frac{2W}{\rho S C_{L,\max}}}$$

V_s=2Wρ SC_L,

Esempio 3.2, da Introduzione all'ingegneria aerospaziale

Esempi svolti passo passo

Esempio 3.1 Scrivi e testa una funzione da Celsius a Fahrenheit.

```
def c_to_f(c):
    """Convert Celsius to Fahrenheit."""
    return c * 9 / 5 + 32

for c in (-40, 0, 1.28, 100):
    print(c, "->", round(c_to_f(c), 2))
```

1. −40 è uguale su entrambe le scale, un controllo comodo.
2. round(..., 2) sistema i float.

Output

```
-40 -> -40.0
0 -> 32.0
1.28 -> 34.3
100 -> 212.0
```

Risposta: Un'anomalia di 1.28 °C è una differenza di 2.3 °F, ma 1.28 °C come temperatura sono 34.3 °F; le unità contano.

Esempio 3.2 Trasforma la formula della velocità di stallo in una funzione e usala a due densità.

```
def stall_speed(weight_n, rho, area, cl_max=1.6):
    return (2 * weight_n / (rho * area * cl_max)) ** 0.5

print(round(stall_speed(10791, 1.227, 16.2), 1))
print(round(stall_speed(10791, 0.738, 16.2), 1))
```

1. cl_max ha un valore predefinito, quindi chi chiama può ometterlo.
2. ** 0.5 è una radice quadrata.

Output

```
26.0
33.6
```

Risposta: 26.0 m/s al livello del mare e 33.6 m/s a 5000 m, come nel corso di aerospaziale.

Esempio 3.3 Scrivi una funzione che restituisca le variazioni anno su anno di una lista (medie annue di Mauna Loa 2022–2025).

```
def growth(values):
    """Year-to-year changes of a list."""
    return [b - a for a, b in zip(values, values[1:])]

co2 = [418.53, 421.08, 424.61, 427.35]   # 2022-2025
print([round(d, 2) for d in growth(co2)])
```

1. zip accoppia ogni valore con il successivo.
2. Una list comprehension costruisce il risultato.

Output

```
[2.55, 3.53, 2.74]
```

Risposta: Aumenti di 2.55, 3.53 e 2.74 ppm.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Scrivi area_circle(r) e testala con r = 2.

   return math.pi * r**2 → 12.566....

2. Cosa restituisce una funzione senza return?

   None.

3. Perché testare c_to_f a −40?

   Lì le due scale coincidono, quindi la risposta attesa è nota.

Errori concettuali comuni

Mito: print e return sono la stessa cosa.

In realtà: print mostra un valore; return lo restituisce per un uso successivo.

Mito: Cambiare un parametro cambia la variabile del chiamante.

In realtà: Riassegnare un parametro è locale (gli oggetti mutabili sono un'eccezione).

Mito: Le funzioni devono essere lunghe e fare tutto.

In realtà: Le funzioni piccole e con un solo scopo sono più facili da testare.

Flashcard

def

Definisce una funzione.

return

Restituisce un valore.

Parametro

Nome per un input.

Argomento predefinito

Valore usato se omesso.

Docstring

Stringa di documentazione.

Ambito

Dove un nome è visibile.

Quiz di autoverifica

1. Si definisce una funzione con...

func

def

function

solo lambda

def. def name(params):

2. Una funzione senza istruzione return restituisce...

0

None

False

un errore

None. None implicito.

3. In def f(x, k=2), k è...

obbligatorio

un argomento predefinito

globale

una costante

un argomento predefinito. Ha un valore predefinito.

4. Le variabili create dentro una funzione sono...

globale

locali

condivise

permanenti

locali. Ambito locale.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

I codici di climatologia e idrologia sono costruiti con piccole funzioni testate come queste, per conversioni di unità, tassi di variazione e formule fisiche; è così che i grandi modelli restano affidabili.

Su Saving Planets: Introduzione all'ingegneria aerospaziale

Fonti: NOAA GML: Trends in atmospheric CO_2, data (https://gml.noaa.gov/ccgg/trends/data.html) · Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/)

Risorse di apprendimento gratuite

- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Sezione 4.8: definire funzioni.
- Harvard CS50's Introduction to Programming with Python (https://cs50.harvard.edu/python/): Funzioni e variabili.

☐ **Ho finito questa unità**

Liste, dizionari e stringhe

Memorizzare raccolte di dati: liste per le sequenze, dizionari per le ricerche e stringhe per il testo.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Indicizzare, affettare e modificare liste.
- Usare len, sum, min, max e sorted.
- Costruire e interrogare dizionari.
- Dividere stringhe e contare parole.

Concetti chiave

- Una lista contiene una sequenza ordinata: flows[0] è il primo elemento, flows[-1] l'ultimo, flows[2:5] una fetta.
- Le liste sono mutabili: append, insert, remove e sort le modificano sul posto.
- Un dizionario associa chiavi a valori: gistemp[2024] → 1.28. Le ricerche sono rapide a prescindere dalle dimensioni.
- Le list comprehension costruiscono liste in modo conciso: [x * 2 for x in xs if x > 0].
- Le stringhe sono sequenze di caratteri con metodi come split, strip, lower e startswith.

Equazioni chiave

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

x=1n_i=1^n x_i

Media di una lista (esempio 4.1)

Esempi svolti passo passo

Esempio 4.1 Riassumi dieci giorni di portata media giornaliera USGS del Potomac a Little Falls (1–10 giugno 2025, in piedi cubi al secondo).

```
flows = [44700, 40200, 29100, 22700, 17300, 14100, 12600, 12200, 14300, 22600]
print(len(flows), max(flows), min(flows))
print(sum(flows) / len(flows))
print(sorted(flows)[:3])
```

1. len, max e min funzionano su qualsiasi lista.
2. sum/len è la media; sorted restituisce una nuova lista.

Output

```
10 44700 12200
22980.0
[12200, 12600, 14100]
```

Risposta: Media 22980 cfs; intervallo 12200–44700 cfs.

Esempio 4.2 Trova l'anno più caldo in un dizionario di anomalie NASA GISTEMP.

```
gistemp = {2016: 1.01, 2020: 1.01, 2023: 1.17, 2024: 1.28, 2025: 1.18}
warmest = max(gistemp, key=gistemp.get)
print(warmest, gistemp[warmest])
print([y for y, a in gistemp.items() if a > 1.1])
```

1. max con key=gistemp.get confronta i valori.
2. Una comprehension filtra gli elementi.

Output

```
2024 1.28
[2023, 2024, 2025]
```

Risposta: 2024, a 1.28 °C; 2023–2025 sono tutti sopra 1.1 °C.

Esempio 4.3 Conta le parole con un dizionario.

```
words = "reduce reuse recycle reduce repair reduce".split()
counts = {}
for w in words:
    counts[w] = counts.get(w, 0) + 1
print(counts)
```

1. get(w, 0) restituisce 0 per una parola nuova.

Output

```
{'reduce': 3, 'reuse': 1, 'recycle': 1, 'repair': 1}
```

Risposta: «reduce» compare 3 volte.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Cos'è [10, 20, 30, 40][1:3]?

   [20, 30].

2. Come aggiungi 2026: 1.2 a gistemp?

   gistemp[2026] = 1.2.

3. Eleva al quadrato ogni numero di xs con una comprehension.

   [x**2 for x in xs].

Errori concettuali comuni

Mito: Le liste partono dall'indice 1.
In realtà: Partono da 0.

Mito: sorted(xs) modifica xs.
In realtà: sorted restituisce una nuova lista; xs.sort() la modifica.

Mito: Le chiavi di un dizionario possono ripetersi.
In realtà: Ogni chiave è unica; assegnarla di nuovo sostituisce il valore.

Flashcard

Lista

Sequenza ordinata e mutabile.

Indice / fetta

xs[i], xs[a:b].

Dizionario

Mappa chiave → valore.

Comprehension

[f(x) for x in xs if cond].

Tupla

Sequenza immutabile.

str.split

Testo → lista di pezzi.

Quiz di autoverifica

1. xs[-1] è...

il primo elemento

l'ultimo elemento

un errore

la lunghezza

l'ultimo elemento. Gli indici negativi contano dalla fine.

2. len({"a": 1, "b": 2}) =

1

2

4

3

2. Due chiavi.

3. "a,b,c".split(",") dà...

"abc"

["a", "b", "c"]

("a","b","c")

3

["a", "b", "c"]. Una lista di stringhe.

4. Le liste sono...

immutabili

mutabili

dizionari

sempre ordinate

mutabili. Possono cambiare sul posto.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

L'USGS pubblica la portata giornaliera di migliaia di idrometri. I dieci numeri dell'esempio 4.1 sono reali: un'onda di piena del Potomac che cala nell'arco di una settimana, il tipo di serie usata per l'approvvigionamento idrico e la pianificazione delle piene.

Su Saving Planets: Pagina Fiumi

Fonti: USGS: Potomac River near Washington, DC (Little Falls), site 01646500 (https://waterdata.usgs.gov/monitoring-location/USGS-01646500/) · NASA GISS: GISTEMP v4 surface temperature analysis (https://data.giss.nasa.gov/gistemp/)

Risorse di apprendimento gratuite

- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Sezione 5: strutture dati.
- USGS Water Services (API dei dati) (https://waterservices.usgs.gov/): Come richiedere i dati all'USGS.

☐ **Ho finito questa unità**

File e dati CSV

Leggere e scrivere file, analizzare dati CSV e gestire righe difettose senza bloccarsi.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Aprire, leggere e scrivere file di testo con with.
- Analizzare CSV con il modulo csv.
- Convertire stringhe in numeri e gestire gli errori con try/except.
- Scaricare un dataset reale e analizzarlo.

Concetti chiave

- with open(name) as f: apre un file e lo chiude automaticamente.
- Il CSV (valori separati da virgole) è il formato più comune per i dati pubblici. csv.DictReader trasforma ogni riga in un dizionario con chiavi prese dall'intestazione.
- Tutto ciò che si legge da un file è testo: converti con int() e float().
- I dati reali hanno commenti, campi vuoti e valori errati. try/except ValueError ti permette di saltarli o registrarli.
- Il file delle medie annue di Mauna Loa della NOAA inizia con righe di commento #, poi year,mean,unc.

Equazioni chiave

$$\text{rate} = \frac{x_{last} - x_{first}}{t_{last} - t_{first}}$$

```
rate=_last-x_firstt_last-t_first
```

Tasso di variazione medio (esempio 5.1)

$$1\ \text{cfs} = 0.0283168\ \text{m}^3/\text{s}$$

```
1\ cfs=0.0283168\ m^3/s
```

Conversione di unità (esempio 5.2)

Esempi svolti passo passo

Esempio 5.1 Analizza le ultime cinque righe del CSV delle medie annue di Mauna Loa della NOAA e trova il tasso medio di aumento.

```
import csv, io
text = """year,mean,unc
2021,416.41,0.12
2022,418.53,0.12
2023,421.08,0.12
2024,424.61,0.12
2025,427.35,0.12
"""
rows = list(csv.DictReader(io.StringIO(text)))
years = [int(r["year"]) for r in rows]
ppm = [float(r["mean"]) for r in rows]
print(years[0], years[-1], ppm[-1])
print(round((ppm[-1] - ppm[0]) / (years[-1] - years[0]), 2), "ppm per year")
```

1. io.StringIO permette a csv di leggere una stringa come se fosse un file.
2. Converti i campi di testo in int e float.

Output

```
2021 2025 427.35
2.73 ppm per year
```

Risposta: 2.73 ppm all'anno dal 2021 al 2025.

Esempio 5.2 Scrivi un piccolo CSV di portate USGS, rileggilo e converti in m³/s.

```
with open("flows.csv", "w") as f:
    f.write("date,cfs\n2025-06-01,44700\n2025-06-02,40200\n2025-06-03,29100\n")
with open("flows.csv") as f:
    next(f)                      # skip the header
    for line in f:
        date, cfs = line.strip().split(",")
        print(date, round(int(cfs) * 0.0283168, 1), "m3/s")
```

1. next(f) salta la riga di intestazione.
2. strip rimuove il ritorno a capo; split separa i campi.

Output

```
2025-06-01 1265.8 m3/s
2025-06-02 1138.3 m3/s
2025-06-03 824.0 m3/s
```

Risposta: 44700 cfs ≈ 1265.8 m^3/s.

Esempio 5.3 Leggi righe con commenti e valori errati senza bloccarti.

```
lines = ["# comment", "year,mean", "2024,424.61", "2025,bad", "2026,"]
good = []
for line in lines:
    if line.startswith("#") or line.startswith("year"):
        continue
    year, value = line.split(",")
    try:
        good.append((int(year), float(value)))
    except ValueError:
        print("skipping", repr(line))
print(good)
```

1. continue salta i commenti e l'intestazione.
2. float("bad") e float("") sollevano ValueError, che viene intercettato.

Output

```
skipping '2025,bad'
skipping '2026,'
[(2024, 424.61)]
```

Risposta: Viene tenuta solo la riga valida; quelle errate vengono segnalate.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Come si legge l'intero file NOAA in Colab?

   Usa urllib.request.urlopen(url) o scaricalo, poi salta le righe che iniziano con #.

2. Cosa restituisce csv.DictReader per ogni riga?

   Un dict con chiavi pari ai nomi delle colonne e valori stringa.

3. Perché usare with open(...)?

   Il file viene chiuso automaticamente, anche in caso di errore.

Errori concettuali comuni

Mito: I numeri in un CSV sono numeri.

In realtà: Vengono letti come stringhe finché non vengono convertiti.

Mito: Le righe errate rendono inutile il dataset.

In realtà: Saltale e registrale, poi riporta quante sono.

Mito: Aprire un file in modalità "w" aggiunge in coda.

In realtà: "w" sovrascrive; "a" aggiunge in coda.

Flashcard

with open()

Apre un file e lo chiude automaticamente.

CSV

Valori separati da virgole.

csv.DictReader

Righe come dict.

try / except

Gestire gli errori.

ValueError

Conversione non valida.

Modalità "w" / "a"

Sovrascrivere / aggiungere.

Quiz di autoverifica

1. float("3.5") dà...

"3.5"

3.5

3

un errore

3.5. Converte testo in float.

2. float("") solleva...

TypeError

ValueError

KeyError

niente

ValueError. Una stringa vuota non è un numero.

3. La modalità di file "a"...

sovrascrive

aggiunge in coda

legge

cancella

aggiunge in coda. Aggiunta.

4. Le chiavi di csv.DictReader vengono da...

i numeri di riga

la riga di intestazione

il nome del file

a caso

la riga di intestazione. Prima riga.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

NOAA, NASA e USGS condividono le loro misure come file CSV gratuiti. Saperli leggere da soli significa poter verificare i titoli sui dati originali.

Su Saving Planets: Analisi II

Fonti: NOAA GML: Mauna Loa CO_2 annual means (CSV) (https://gml.noaa.gov/webdata/ccgg/trends/co2/co2_annmean_mlo.csv) · Documentazione Python: il modulo csv (https://docs.python.org/3/library/csv.html)

Risorse di apprendimento gratuite

- Documentazione Python: il modulo csv (https://docs.python.org/3/library/csv.html): Leggere e scrivere CSV.
- NOAA GML: Mauna Loa CO_2 annual means (CSV) (https://gml.noaa.gov/webdata/ccgg/trends/co2/co2_annmean_mlo.csv): Il file reale.
- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Sezione 7.2: file.

☐ **Ho finito questa unità**

NumPy e Matplotlib

Calcolare su interi array in un colpo solo con NumPy, stimare tendenze e fare grafici chiari con Matplotlib.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Creare array NumPy e usare l'aritmetica vettorializzata.
- Usare mean, diff, maschere booleane e polyfit.
- Tracciare dati con assi etichettati e titoli.
- Salvare figure su file.

Concetti chiave

- Un array NumPy contiene numeri di un solo tipo e calcola su tutti insieme: cfs * 0.0283168 converte ogni valore.
- np.diff dà le differenze successive; una maschera booleana come cfs > 20000 seleziona o conta valori.
- np.polyfit(x, y, 1) adatta una retta e restituisce pendenza e intercetta.
- pyplot di Matplotlib disegna figure: fig, ax = plt.subplots(); ax.plot(x, y); etichetta gli assi con le unità; fig.savefig salva l'immagine.
- Un buon grafico ha assi etichettati con unità, un titolo e una fonte dei dati citata.

Equazioni chiave

$$y \approx mx + b$$

Retta dei minimi quadrati con np.polyfit

$$V = \sum Q_i \, \Delta t$$

`V= Q_i Δ t`

Volume dalle portate giornaliere (esempio 6.2)

Esempi svolti passo passo

Esempio 6.1 Usa NumPy sulle medie annue di Mauna Loa 2021–2025: differenze, media e tendenza.

```
import numpy as np
years = np.array([2021, 2022, 2023, 2024, 2025])
```

```
ppm = np.array([416.41, 418.53, 421.08, 424.61, 427.35])
print(np.diff(ppm))
print(round(ppm.mean(), 3), round(np.diff(ppm).mean(), 3))
slope, intercept = np.polyfit(years, ppm, 1)
print(round(slope, 3))
```

1. np.diff dà quattro aumenti annui.
2. La pendenza di polyfit è il tasso di miglior adattamento.

Output

```
[2.12 2.55 3.53 2.74]
421.596 2.735
2.796
```

Risposta: Aumento stimato ≈ 2.80 ppm all'anno nel 2021–2025.

Esempio 6.2 Converti dieci giorni di portata del Potomac in m³/s, conta i giorni di piena e trova il volume totale.

```
import numpy as np
cfs = np.array([44700, 40200, 29100, 22700, 17300, 14100, 12600, 12200, 14300, 22600])
m3s = cfs * 0.0283168
print(np.round(m3s[:3], 1))
print(int((cfs > 20000).sum()), "days above 20,000 cfs")
print(round(m3s.sum() * 86400 / 1e9, 3), "km3 in 10 days")
```

1. Moltiplica tutto l'array in un colpo.
2. (cfs > 20000).sum() conta i valori True.
3. Somma di m³/s × 86400 s al giorno.

Output

```
[1265.8 1138.3  824. ]
5 days above 20,000 cfs
0.562 km3 in 10 days
```

Risposta: Cinque giorni sopra 20000 cfs; in dieci giorni sono passati circa 0.56 km³ d'acqua.

Esempio 6.3 Traccia una selezione di anomalie annue NASA GISTEMP e salva la figura.

```
import matplotlib
matplotlib.use("Agg")
import matplotlib.pyplot as plt
years = [1880, 1951, 1980, 2000, 2016, 2020, 2023, 2024, 2025]
anom = [-0.17, -0.07, 0.26, 0.39, 1.01, 1.01, 1.17, 1.28, 1.18]
fig, ax = plt.subplots()
ax.plot(years, anom, "o-")
ax.set_xlabel("Year"); ax.set_ylabel("Anomaly (C)")
ax.set_title("NASA GISTEMP v4, selected years")
fig.savefig("gistemp.png")
print(len(ax.lines[0].get_xdata()), "points saved")
```

1. matplotlib.use("Agg") permette al codice di salvare immagini senza schermo (Colab mostra comunque i grafici in linea).
2. Etichetta assi e titolo; salva con savefig.

Output

```
9 points saved
```

Risposta: Un grafico a linee salvato di 9 punti; apri gistemp.png per vederlo.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. np.array([1, 2, 3]) * 2 dà?

   array([2, 4, 6]).

2. Come selezioni i valori sopra 1.0 in un array a?

   a[a > 1.0].

3. Quale chiamata aggiunge un'etichetta all'asse?

   ax.set_xlabel("Year").

Errori concettuali comuni

Mito: Array NumPy e liste si sommano allo stesso modo.

In realtà: [1, 2] + [3, 4] concatena; gli array si sommano elemento per elemento.

Mito: Una retta adattata dimostra una causa.

In realtà: Riassume una tendenza; le cause richiedono una comprensione fisica.

Mito: I grafici senza unità vanno bene.

In realtà: Indica sempre le unità e cita la fonte.

Flashcard

np.array

Array numerico veloce.

Vettorializzato

Calcoli su interi array.

Maschera booleana

a[a > x].

np.polyfit

Adattamento polinomiale ai minimi quadrati.

plt.subplots

Crea una figura e gli assi.

savefig

Salva un grafico.

Quiz di autoverifica

1. np.diff([1, 4, 9]) =

[3, 5]

[1, 4, 9]

[0, 3, 5]

14

[3, 5]. Differenze successive.

2. np.polyfit(x, y, 1) restituisce...

la media

pendenza e intercetta

un grafico

il massimo

pendenza e intercetta. Adattamento di grado 1.

3. (a > 0).sum() conta...

i valori positivi

tutti i valori

gli zeri

niente

i valori positivi. True conta come 1.

4. fig.savefig("x.png")...

stampa

salva la figura

la cancella

mostra il codice

salva la figura. Scrive un file immagine.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

L'analisi GISTEMP della NASA è pubblicata con i suoi dati, così chiunque può riprodurre la serie della temperatura globale. Tracciarla da sé, con le unità indicate e la fonte, è il modo onesto di condividere i dati sul clima.

Su Saving Planets: Dati climatici su questo sito: CO2M

Fonti: NASA GISS: GISTEMP v4 surface temperature analysis (https://data.giss.nasa.gov/gistemp/) · NumPy: the absolute basics for beginners (https://numpy.org/doc/stable/user/absolute_beginners.html)

Risorse di apprendimento gratuite

- NumPy: the absolute basics for beginners (https://numpy.org/doc/stable/user/absolute_beginners.html): Array per principianti.
- Matplotlib: pyplot tutorial (https://matplotlib.org/stable/tutorials/pyplot.html): Basi della grafica.
- NASA GISTEMP v4 global means (CSV) (https://data.giss.nasa.gov/gistemp/tabledata_v4/GLB.Ts+dSST.csv): I dati reali.

- [ ] **Ho finito questa unità**

Debug e test

Trovare i bug in modo sistematico, leggere i traceback e scrivere test che intercettino gli errori prima che contino.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Leggere un traceback e individuare la riga che fallisce.
- Usare print e asserzioni per circoscrivere i bug.
- Sollevare errori chiari per input non validi.
- Scrivere test unitari con assert e pytest.

Concetti chiave

- Leggi i messaggi di errore dal basso: l'ultima riga indica il tipo di errore, le righe sopra mostrano dove è avvenuto.
- Fai il debug in modo scientifico: formula un'ipotesi, verificala con un input piccolo, cambia una cosa sola.
- I casi limite (liste vuote, zero, numeri negativi, dati mancanti) causano la maggior parte dei bug.
- Un assert dichiara ciò che deve essere vero; un assert che fallisce ferma il programma sul bug.
- I test sono piccole funzioni che verificano risposte note; pytest esegue automaticamente ogni funzione chiamata test_....

Equazioni chiave

$$\left|x_{computed} - x_{expected}\right| < \varepsilon$$

Test con tolleranza per i float

Esempi svolti passo passo

Esempio 7.1 Trova il bug in una funzione di media chiamata su una lista vuota.

```
def mean(xs):
    return sum(xs) / len(xs)

try:
    print(mean([]))
except ZeroDivisionError as e:
    print("bug found:", e)
```

1. Dividere per len([]) = 0 solleva ZeroDivisionError.
2. Intercettarlo mostra il messaggio.

Output

```
bug found: division by zero
```

Risposta: La funzione deve gestire un input vuoto.

Esempio 7.2 Correggila con un errore chiaro e aggiungi test con asserzioni.

```
def mean(xs):
    if not xs:
        raise ValueError("mean() of empty list")
    return sum(xs) / len(xs)

assert mean([1, 2, 3]) == 2
assert abs(mean([0.1, 0.2]) - 0.15) < 1e-12
try:
    mean([])
except ValueError:
    print("all tests passed")
```

1. Solleva ValueError con un messaggio in caso di input vuoto.
2. I risultati float sono confrontati con una tolleranza.

Output

```
all tests passed
```

Risposta: Tutti i test passano.

Esempio 7.3 Scrivi un test in stile pytest per f_to_c usando punti noti.

```
def f_to_c(f):
    return (f - 32) * 5 / 9

def test_f_to_c():
    assert f_to_c(32) == 0
    assert f_to_c(212) == 100
    assert abs(f_to_c(-40) - (-40)) < 1e-12
```

```
test_f_to_c()
print("ok")
```

1. Coppie note: 32 °F = 0 °C, 212 °F = 100 °C, −40 = −40.
2. In un file, pytest troverebbe test_f_to_c automaticamente; qui la chiamiamo noi.

Output

```
ok
```

Risposta: La funzione supera tutti e tre i controlli.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Cosa significa di solito «IndexError: list index out of range»?

   Hai usato un indice ≥ len(list), spesso un errore di uno.

2. Scrivi un assert che x sia positivo.

   assert x > 0, "x must be positive".

3. Indica tre casi limite per una funzione che fa la media dei dati.

   Lista vuota, un solo valore e valori mancanti o non numerici.

Errori concettuali comuni

Mito: Se gira, è corretto.
In realtà: Il codice può girare e dare risposte sbagliate; testalo su risultati noti.

Mito: Confronta i float con ==.
In realtà: Usa una tolleranza.

Mito: I test servono solo ai grandi progetti.
In realtà: Piccoli test fanno risparmiare tempo fin dal primo giorno.

Flashcard

Traceback	**Caso limite**
Rapporto d’errore; si legge dal basso.	Input insolito al confine.

assert

Si ferma se una condizione è falsa.

Test unitario

Verifica una funzione su risposte note.

pytest

Esecutore di test.

raise

Segnala un errore deliberatamente.

Quiz di autoverifica

1. La riga più utile di un traceback di solito è...

la prima

l'ultima

quella in mezzo

nessuna

l'ultima. Indica l'errore.

2. pytest esegue le funzioni chiamate...

run_*

test_*

check_*

main

test_*. Convenzione.

3. Un assert che fallisce solleva...

ValueError

AssertionError

niente

TypeError

AssertionError. AssertionError.

4. Un buon primo passo di debug è…

riscrivere tutto

riprodurre con un input piccolo

aggiungere print a caso ovunque

ignorarlo

riprodurre con un input piccolo. Riduci il problema.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

Il codice scientifico che orienta le politiche deve essere verificato. Questo corso mette in pratica ciò che predica: ogni esempio di codice di questa pagina viene eseguito dal nostro script di build, e il suo output deve corrispondere esattamente prima della pubblicazione.

Su Saving Planets: Corsi

Fonti: Documentazione di pytest (https://docs.pytest.org/) · Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/)

Risorse di apprendimento gratuite

- Documentazione di pytest (https://docs.pytest.org/): Scrivere test.
- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Sezione 8: errori ed eccezioni.

Guarda la lezione

Lecture 7 testing debugging exceptions and assertions (https://ocw.mit.edu/courses/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016/resources/lecture-7-testing-debugging-exceptions-and-assertions/) (in inglese)

MIT 6.0001 Introduction to Computer Science and Programming in Python (Fall 2016), MIT OpenCourseWare. Licenza: CC BY-NC-SA 4.0 (https://creativecommons.org/licenses/by-nc-sa/4.0/) · Condizioni d'uso (https://ocw.mit.edu/pages/privacy-and-terms-of-use/). Link fornito solo per lo studio gratuito; Saving Planets non è affiliato a questa università né approvato da essa.

- [ ] **Ho finito questa unità**

Algoritmi e complessità

Ricerca, ricerca delle radici e come cresce il tempo di esecuzione con la dimensione dell'input: ragionare in O grande.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Implementare la ricerca lineare e la ricerca binaria.
- Usare la bisezione per risolvere equazioni.
- Stimare la complessità: O(1), O(log n), O(n), O(n²).
- Scegliere le strutture dati per la velocità.

Concetti chiave

- Un algoritmo è un metodo preciso passo per passo. Il suo costo è come cresce il numero di passi con la dimensione n dell'input.
- La ricerca lineare controlla gli elementi uno per uno: O(n). La ricerca binaria dimezza una lista ordinata a ogni passo: O(log n).
- La bisezione trova una radice di f su [a, b], dove f cambia segno, dimezzando l'intervallo; dopo n passi l'intervallo è (b − a)/2ⁿ.
- Cicli annidati su n elementi sono O(n²): vanno bene per 1000 elementi, senza speranza per un milione.
- Le ricerche in dizionari e insiemi sono O(1) in media, quelle nelle liste O(n).

Equazioni chiave

$$\text{steps}_{binary} \approx \lceil \log_2 n \rceil$$

Passi della ricerca binaria

$$n = \left\lceil \log_2 \frac{b - a}{\text{tol}} \right\rceil$$

`n=_2-atol`

Iterazioni della bisezione

Esempi svolti passo passo

Esempio 8.1 Cerca l'ultimo elemento di una lista ordinata di un milione di numeri pari, in entrambi i modi.

```
def linear_search(xs, target):
    for i, x in enumerate(xs):
        if x == target:
            return i
    return -1

def binary_search(xs, target):
    lo, hi, steps = 0, len(xs) - 1, 0
    while lo <= hi:
        steps += 1
        mid = (lo + hi) // 2
        if xs[mid] == target:
            return mid, steps
        if xs[mid] < target:
            lo = mid + 1
        else:
            hi = mid - 1
    return -1, steps

data = list(range(0, 2_000_000, 2))
print(linear_search(data, 1_999_998))
print(binary_search(data, 1_999_998))
```

1. La ricerca lineare controlla 1000000 elementi per arrivare all'indice 999999.
2. La ricerca binaria richiede solo 20 passi.

Output

```
999999
(999999, 20)
```

Risposta: 20 passi contro un milione: $\log_2(10^6) \approx 20$.

Esempio 8.2 Trova $\sqrt{2}$ per bisezione su $x^2 - 2$ da $[0, 2]$ fino a 10^{-6}.

```
def bisect_root(f, a, b, tol=1e-6):
    n = 0
    while b - a > tol:
        m = (a + b) / 2
        if f(a) * f(m) <= 0:
            b = m
        else:
            a = m
        n += 1
    return (a + b) / 2, n
```

```
root, n = bisect_root(lambda x: x**2 - 2, 0, 2)
print(round(root, 6), n)
```

1. Ogni passo dimezza l'intervallo.
2. $2/2^{21} < 10^{-6}$ dopo 21 passi.

Output

```
1.414214 21
```

Risposta: 1.414214 dopo 21 iterazioni.

Esempio 8.3 Confronta la crescita di $\log_2 n$ e n^2.

```
import math
for n in (10, 1_000, 1_000_000):
    print(n, math.ceil(math.log2(n)), n * n)
```

1. math.ceil(math.log2(n)) è il numero di passi della ricerca binaria.

Output

```
10 4 100
1000 10 1000000
1000000 20 1000000000000
```

Risposta: Per $n = 10^6$, $\log_2 n$ vale 20 mentre n^2 vale 10^{12}.

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Quanti passi di ricerca binaria per 1 miliardo di elementi ordinati?

   $\lceil \log_2 10^9 \rceil = 30$.

2. Qual è la complessità della somma di una lista?

   $O(n)$.

3. Bisezione su $[0, 1]$ fino a 10^{-3}: quanti passi?

   $\lceil \log_2 1000 \rceil = 10$.

Errori concettuali comuni

Mito: Computer più veloci rendono irrilevante la complessità.

In realtà: Un metodo $O(n^2)$ su un milione di elementi richiede 10^{12} passi; nessun computer lo risolve.

Mito: La ricerca binaria funziona su qualsiasi lista.

In realtà: Solo su dati ordinati.

Mito: La O grande dà il tempo esatto.

In realtà: Descrive la crescita, ignorando le costanti.

Flashcard

Algoritmo

Metodo preciso passo per passo.

Big-O

Tasso di crescita del costo.

Ricerca lineare

O(n).

Ricerca binaria

O(log n), dati ordinati.

Bisezione

Ricerca di radici per dimezzamento.

$O(n^2)$

Cicli annidati.

Quiz di autoverifica

1. La ricerca binaria è...

O(n)

O(log n)

$O(n^2)$

O(1)

O(log n). Dimezza a ogni passo.

2. La bisezione richiede...

una derivata

un cambio di segno su [a, b]

dati ordinati

interi

un cambio di segno su [a, b]. Teorema dei valori intermedi.

3. Un ciclo dentro un ciclo su n elementi è...

O(n)

$O(n^2)$

O(log n)

O(1)

$O(n^2)$. n × n.

4. Una ricerca media in un dict è...

O(n)

O(1)

$O(n^2)$

O(log n)

O(1). Hashing.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

Algoritmi efficienti permettono di elaborare la marea di dati di osservazione della Terra, dai satelliti a migliaia di idrometri. Scegliere O(n log n) invece di $O(n^2)$ può trasformare una settimana di calcolo

in secondi, e meno calcolo significa anche meno energia.

Su Saving Planets: Analisi I

Fonti: USGS Water Services (API dei dati) (https://waterservices.usgs.gov/) · MIT OpenCourseWare 6.100L: Introduction to CS and Programming Using Python (https://ocw.mit.edu/courses/6-100l-introduction-to-cs-and-programming-using-python-fall-2022/)

Risorse di apprendimento gratuite

- MIT OpenCourseWare 6.100L: Introduction to CS and Programming Using Python (https://ocw.mit.edu/courses/6-100l-introduction-to-cs-and-programming-using-python-fall-2022/): Lezioni su ricerca, bisezione e complessità.
- Harvard CS50's Introduction to Programming with Python (https://cs50.harvard.edu/python/): Esercizi di problem solving.

☐ **Ho finito questa unità**

Simulazione numerica

Far avanzare nel tempo modelli fisici: un proiettile, un'orbita e una popolazione, il cuore di ogni modello climatico e ingegneristico.

Obiettivi di apprendimento

Alla fine di questa unità saprai:

- Usare il passo temporale (Euler ed Euler-Cromer) per simulare il moto.
- Verificare le simulazioni con risposte esatte.
- Simulare un'orbita circolare e una popolazione logistica.
- Esplorare come la dimensione del passo influisce sull'accuratezza.

Concetti chiave

- Una simulazione aggiorna lo stato (posizione, velocità, popolazione) a piccoli passi temporali dt usando i tassi di variazione: x_new = x + v·dt.
- Euler-Cromer aggiorna prima la velocità, poi la posizione con la nuova velocità; mantiene le orbite stabili molto meglio del semplice Euler.
- Valida sempre: confronta con una soluzione esatta in un caso in cui esiste (senza resistenza dell'aria, orbita circolare, formula logistica).
- Un dt più piccolo è più accurato ma più lento; dimezza dt e guarda se la risposta cambia.
- I modelli meteorologici e climatici sono enormi versioni di questi cicli, che fanno avanzare le equazioni della fisica su una griglia globale.

Equazioni chiave

$$v_{k+1} = v_k + a_k\,\Delta t, \quad x_{k+1} = x_k + v_{k+1}\,\Delta t$$

```
v_k+1=v_k+a_k Δ t, x_k+1=x_k+v_k+1 Δ t
```

Passo di Euler-Cromer

$$R = \frac{v_0^2 \sin 2\theta}{g}$$

```
R=(v_0^22θ)/(g)
```

Gittata di un proiettile senza resistenza

$$\vec{a} = -\frac{GM}{r^3}\vec{r}$$

```
a=-(GM)/(r^3) r
```

Gravità per un'orbita

$$P_{k+1} = P_k + h\,r P_k \left(1 - \frac{P_k}{K}\right)$$

P_k+1=P_k+h rP_k(1-_kK)

Modello logistico, passo di Euler

Esempi svolti passo passo

Esempio 9.1 Simula un proiettile a 20 m/s lanciato a 45° e confronta con la gittata esatta.

```
import math
v0, angle, g, dt = 20.0, math.radians(45), 9.81, 0.001
x, y = 0.0, 0.0
vx, vy = v0 * math.cos(angle), v0 * math.sin(angle)
while y >= 0:
    vy -= g * dt
    x += vx * dt
    y += vy * dt
print(round(x, 2), "m simulated;", round(v0**2 / g, 2), "m exact")
```

1. Fai avanzare la velocità, poi la posizione, finché non atterra.
2. Gittata esatta = v_0^2 sin 90°/g.

Output

```
40.77 m simulated; 40.77 m exact
```

Risposta: Entrambi danno 40.77 m a due decimali con dt = 0.001 s.

Esempio 9.2 Simula un'orbita a 400 km sopra la Terra con passi di 1 secondo.

```
import math
GM = 3.986e14
r0 = 6.771e6
x, y = r0, 0.0
vx, vy = 0.0, math.sqrt(GM / r0)
dt, t = 1.0, 0
period = 2 * math.pi * math.sqrt(r0**3 / GM)
while t < period:
    r3 = (x * x + y * y) ** 1.5
    vx -= GM * x / r3 * dt          # Euler-Cromer: update velocity first
```

```
    vy -= GM * y / r3 * dt
    x += vx * dt
    y += vy * dt
    t += dt
print(round(period / 60, 1), "min;", round(math.hypot(x, y) / 1000), "km from Earth's center
```

1. Parti da r_0 con la velocità circolare $\sqrt{(GM/r_0)}$.
2. Aggiorna la velocità dalla gravità, poi la posizione.

Output

```
92.4 min; 6771 km from Earth's center
```

Risposta: Dopo un periodo (92.4 minuti) il veicolo è ancora a 6771 km dal centro della Terra: l'orbita resta circolare.

Esempio 9.3 Fai evolvere una popolazione logistica di pesci ($r = 0.5$, $K = 1000$, $P_0 = 100$) per quattro anni.

```
r, K, P, h = 0.5, 1000, 100.0, 1.0
for year in range(1, 5):
    P += h * r * P * (1 - P / K)
    print(year, round(P, 1))
```

1. Ogni anno aggiunge $rP(1 - P/K)$.

Output

```
1 145.0
2 207.0
3 289.1
4 391.8
```

Risposta: 145.0, 207.0, 289.1, 391.8, come nel laboratorio A di Analisi II (lì 391.9, una differenza di arrotondamento).

Esercizi

Svolgi ciascuno prima su carta, poi controlla.

1. Cosa succede all'errore dell'esempio 9.1 se dt è 0.1 s?

   L'atterraggio viene rilevato tardi, quindi l'errore sulla gittata cresce; prova e confronta.

2. Perché il semplice Euler fa allargare un’orbita a spirale?

   Aggiunge un po’ di energia a ogni passo; Euler-Cromer la conserva quasi.

3. Aggiungi una resistenza −k v v□ all’esempio 9.1. Cosa succede all’angolo migliore?

   Scende sotto 45° (laboratorio D).

Errori concettuali comuni

Mito: Una simulazione che gira è corretta.
In realtà: Valida su casi esatti e verifica che dimezzare dt cambi appena la risposta.

Mito: Un dt più piccolo è sempre meglio.
In realtà: Costa tempo e può aggiungere errore di arrotondamento; scegli dt con test di convergenza.

Mito: I modelli predicono esattamente il futuro.
In realtà: Sono approssimazioni le cui ipotesi vanno dichiarate.

Flashcard

Passo temporale dt

Dimensione di ogni aggiornamento.

Metodo di Euler

x += v·dt.

Euler-Cromer

Prima la velocità, poi la posizione.

Validazione

Verifica su risposte note.

Convergenza

La risposta smette di cambiare quando dt si riduce.

Stato

Variabili che descrivono il sistema ora.

Quiz di autoverifica

1. Euler-Cromer aggiorna...

prima la posizione

prima la velocità, poi la posizione con la nuova velocità

solo la velocità

niente

prima la velocità, poi la posizione con la nuova velocità. Stabile per le orbite.

2. Per validare una simulazione...

fidarsi

confrontarla con una soluzione esatta

rendere dt enorme

togliere la fisica

confrontarla con una soluzione esatta. Casi noti.

3. Gittata esatta a 20 m/s, 45°, g = 9.81:

20.4 m

40.8 m

81.5 m

9.8 m

40.8 m. 400/9.81.

4. Dimezzare dt senza vedere cambiamenti significa...

un bug

il risultato è andato a convergenza

il modello è sbagliato

niente

il risultato è andato a convergenza. Test di convergenza.

Rifai questo quiz

Come questo aiuta a salvare il pianeta

I modeli di popolazione come l'esempio 9.3 fanno parte del modo in cui gli scienziati della pesca valutano gli stock e fissano limiti di cattura sostenibili.

La stessa idea dei passi temporali, ingrandita enormemente, è il modo in cui si costruiscono i modelli meteorologici e climatici.

Su Saving Planets: Analisi II, laboratorio A · Guida alle orbite

Fonti: NOAA Fisheries: valutazioni delle popolazioni (https://www.fisheries.noaa.gov/topic/population-assessments) · MIT OpenCourseWare 6.100L: Introduction to CS and Programming Using Python (https://ocw.mit.edu/courses/6-100l-introduction-to-cs-and-programming-using-python-fall-2022/)

Risorse di apprendimento gratuite

- MIT OpenCourseWare 6.100L: Introduction to CS and Programming Using Python (https://ocw.mit.edu/courses/6-100l-introduction-to-cs-and-programming-using-python-fall-2022/): Lezioni su simulazione e modellazione.
- NumPy: the absolute basics for beginners (https://numpy.org/doc/stable/user/absolute_beginners.html): Vettorizzare le simulazioni.

☐ **Ho finito questa unità**

Esercizi

Questi esercizi stanno dentro ogni unità. Falli su carta, poi apri la risposta.

1. Primi passi: variabili, tipi ed espressioni
   4 esercizi
2. Controllo del flusso: condizioni e cicli
   3 esercizi
3. Funzioni
   3 esercizi
4. Liste, dizionari e stringhe
   3 esercizi
5. File e dati CSV
   3 esercizi
6. NumPy e Matplotlib
   3 esercizi
7. Debug e test
   3 esercizi
8. Algoritmi e complessità

3 esercizi

9. Simulazione numerica
   3 esercizi

Laboratori di programmazione

Quattro laboratori di programmazione usano dati pubblici reali e simulazione: la serie di CO_2 di Mauna Loa, la serie globale delle temperature della NASA, una stazione idrometrica dell'USGS e un proiettile con resistenza dell'aria.

Il codice iniziale di ogni laboratorio funziona così com'è; i risultati stampati sono stati ottenuti eseguendolo. Poi scarica il dataset completo ed estendi l'analisi.

Riferimento per la sicurezza: Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/)

Laboratorio A: la serie di CO_2 di Mauna Loa (NOAA)

SETTIMANA 6

Obiettivo: Carica il CSV delle medie annuali di CO_2 della NOAA, calcola l'aumento totale e il tasso di crescita recente, e traccia la curva.

$$\text{rate} = \text{slope of } \text{np.polyfit}(\text{year}, \text{ppm}, 1)$$

```
rate=slope of np.polyfit(year, ppm, 1)
```

Tasso di crescita da un fit lineare

Procedimento

1. Esegui il codice iniziale (alcune righe reali del file della NOAA).
2. Scarica il CSV completo dalla NOAA (link sotto), salta le righe #, e carica tutti gli anni.
3. Traccia i ppm in funzione dell'anno con assi etichettati e la fonte.
4. Adatta rette a 1960–1969 e 2016–2025. Come è cambiato il tasso?

```
import csv, io
import numpy as np
text = """year,mean,unc
1959,315.98,0.12
1960,316.91,0.12
2020,414.21,0.12
2021,416.41,0.12
2022,418.53,0.12
2023,421.08,0.12
2024,424.61,0.12
2025,427.35,0.12
"""
rows = list(csv.DictReader(io.StringIO(text)))
yr = np.array([int(r["year"]) for r in rows])
ppm = np.array([float(r["mean"]) for r in rows])
print(round(ppm[-1] - ppm[0], 2), "ppm since 1959")
recent = yr >= 2020
slope = np.polyfit(yr[recent], ppm[recent], 1)[0]
print(round(slope, 2), "ppm per year, 2020-2025")
```

Output

```
111.37 ppm since 1959
2.65 ppm per year, 2020-2025
```

Tabella dei dati

Anno	CO_2 medio annuo (ppm)
1959	315.98
1960	316.91
2020	414.21
2023	421.08
2024	424.61
2025	427.35

Valori reali dal file delle medie annuali di Mauna Loa di NOAA GML, consultato il 24 ott. 2026.

Calcolo di esempio

1. 427.35 − 315.98 = 111.37 ppm dal 1959.
2. Tasso di best fit 2020–2025 ≈ 2.65 ppm all'anno.

> **Sicurezza**
>
> - Nessuna, oltre al normale uso del computer.

Alternativa a casa o virtuale

- Colab o JupyterLite in un browser.

NOAA GML: Mauna Loa CO_2 annual means (CSV) (https://gml.noaa.gov/webdata/ccgg/trends/co2/co2_annmean_mlo.csv) · Google Colab (notebook gratuiti) (https://colab.research.google.com/)

Legame con il pianeta: È la serie dietro la curva di Keeling, la più lunga misura continua dell'aumento della CO_2. NOAA GML: Trends in atmospheric CO_2, data (https://gml.noaa.gov/ccgg/trends/data.html)

☐ **Ho completato questo laboratorio (reale o virtuale)**

Laboratorio B: la temperatura globale con NASA GISTEMP

SETTIMANA 8

Obiettivo: Analizza le anomalie annuali della temperatura globale della NASA: medie, l'anno più caldo e la tendenza.

$$\Delta T \text{ per decade} = 10 \times \text{slope}$$

```
Δ T\ per decade=10×slope
```

Tendenza per decennio

Procedimento

1. Esegui il codice iniziale con le anomalie annuali (J–D) 2016–2025.
2. Scarica il CSV globale completo di GISTEMP; salta la prima riga di titolo; gestisci i valori mancanti ***.
3. Traccia 1880–2025 e calcola la tendenza dal 1975.
4. Confronta un fit su 10 anni con uno su 50 anni e spiega perché le tendenze brevi sono più rumorose.

```
import numpy as np
years = np.arange(2016, 2026)
anom = np.array([1.01, 0.91, 0.85, 0.98, 1.01, 0.85, 0.89, 1.17, 1.28, 1.18])
print(round(anom.mean(), 3))
print(int(years[anom.argmax()]), anom.max())
print(round(np.polyfit(years, anom, 1)[0] * 10, 2), "C per decade (10-year fit)")
```

Output

```
1.013
2024 1.28
0.32 C per decade (10-year fit)
```

Tabella dei dati

Anno	Anomalia (°C rispetto al 1951–1980)
2016	1.01
2020	1.01
2023	1.17
2024	1.28
2025	1.18

Valori reali da NASA GISTEMP v4 (GLB.Ts+dSST, colonna J–D), consultati il 24 ott. 2026. La NASA aggiorna l'analisi, quindi download successivi possono differire leggermente.

Calcolo di esempio

1. Media 2016–2025 = 1.013 °C.
2. Un fit su dieci anni dà circa 0.32 °C per decennio, ma dieci anni sono pochi e rumorosi; usa periodi più lunghi per la tendenza di lungo termine.

> **Sicurezza**
>
> - Nessuna, oltre al normale uso del computer.

Alternativa a casa o virtuale

- Colab o JupyterLite.

NASA GISTEMP v4 global means (CSV) (https://data.giss.nasa.gov/gistemp/tabledata_v4/GLB.Ts+dSST.csv) · NASA GISS: GISTEMP v4 surface temperature analysis (https://data.giss.nasa.gov/gistemp/)

Legame con il pianeta: Riprodurre da sé la serie della NASA è la migliore risposta a «come lo sappiamo?» NASA GISS: GISTEMP v4 surface temperature analysis (https://data.giss.nasa.gov/gistemp/)

- [ ] **Ho completato questo laboratorio (reale o virtuale)**

Laboratorio C: un fiume attraverso una stazione dell’USGS

SETTIMANA 9

Obiettivo: Analizza la portata giornaliera della stazione USGS sul fiume Potomac a Little Falls: unità, statistiche, recessione e volume.

$$V = \sum_{i} Q_i \times 86{,}400 \text{ s}$$

V=_iQ_i×86,400\ s

Volume dalle portate medie giornaliere

Procedimento

1. Esegui il codice iniziale con dieci giorni reali (1–10 giugno 2025).
2. Richiedi un anno completo a USGS Water Services (format=rdb) per il sito 01646500 e il parametro 00060, e analizza il file separato da tabulazioni.
3. Traccia la portata giornaliera, trova il picco e calcola il volume annuo in km³.
4. Scrivi test per la tua funzione di conversione di unità.

```
import numpy as np
cfs = np.array([44700, 40200, 29100, 22700, 17300, 14100, 12600, 12200, 14300, 22600])
m3s = cfs * 0.0283168
print(round(m3s.mean(), 1), round(np.median(m3s), 1))
drops = np.diff(cfs)
print(int(drops.min()), "cfs biggest one-day drop")
print(round(m3s.sum() * 86400 / 1e6, 1), "million m3 in 10 days")
```

Output

```
650.7 564.9
-11100 cfs biggest one-day drop
562.2 million m3 in 10 days
```

Tabella dei dati

Data (2025)	Portata media giornaliera (cfs)
1 giugno	44,700
3 giugno	29,100
5 giugno	17,300
8 giugno	12,200

Data (2025)	Portata media giornaliera (cfs)
10 giugno	22,600

Valori giornalieri reali e approvati dal sito USGS 01646500, consultati il 24 ott. 2026.

Calcolo di esempio

1. Media 650.7 m³/s, mediana 564.9 m³/s (la media è alzata dal picco del 1 giugno).
2. Il maggior calo in un giorno è stato di 11100 cfs; in dieci giorni sono passati circa 562 milioni di m³.

> **Sicurezza**
>
> - Nessuna, oltre al normale uso del computer.

Alternativa a casa o virtuale

- Colab o JupyterLite.

USGS: Potomac River near Washington, DC (Little Falls), site 01646500 (https://waterdata.usgs.gov/monitoring-location/USGS-01646500/) · USGS Water Services (API dei dati) (https://waterservices.usgs.gov/)

Legame con il pianeta: Stazioni idrometriche come questa servono per l'approvvigionamento di acqua potabile, gli allarmi alluvione e i deflussi ecologici. USGS: Potomac River near Washington, DC (Little Falls), site 01646500 (https://waterdata.usgs.gov/monitoring-location/USGS-01646500/)

☐ **Ho completato questo laboratorio (reale o virtuale)**

Laboratorio D: proiettile con resistenza dell’aria

SETTIMANA 14

Obiettivo: Estendi la simulazione del proiettile con resistenza quadratica e trova il miglior angolo di lancio all’aumentare della resistenza.

$$\vec{a} = -g\hat{y} - k\,|\vec{v}|\,\vec{v}$$

```
a=-g y-k | v| v
```

Resistenza quadratica (k per metro)

Procedimento

1. Esegui il codice iniziale, che prova angoli interi da 20° a 60°.
2. Verifica che k = 0 dia 45° e la gittata esatta.
3. Dimezza dt e conferma che i risultati cambiano appena.
4. Traccia la gittata in funzione dell’angolo per ogni k.

```
import math
def fly(v0, deg, k=0.0, dt=0.001, g=9.81):
    vx, vy = v0 * math.cos(math.radians(deg)), v0 * math.sin(math.radians(deg))
    x = y = 0.0
    while y >= 0:
        v = math.hypot(vx, vy)
        vx -= k * v * vx * dt
        vy -= (g + k * v * vy) * dt
        x += vx * dt
        y += vy * dt
    return x
for k in (0.0, 0.01, 0.05):
    best = max(range(20, 61), key=lambda a: fly(20, a, k))
    print(k, best, round(fly(20, best, k), 1))
```

Output

```
0.0 45 40.8
0.01 43 31.4
0.05 39 17.8
```

Tabella dei dati

Resistenza k (1/m)	Angolo migliore	Gittata (m)
0	45°	40.8

Resistenza k (1/m)	Angolo migliore	Gittata (m)
0.01	43°	31.4
0.05	39°	17.8

Calcolato dal codice iniziale (v_0 = 20 m/s, dt = 0.001 s).

Calcolo di esempio

1. Senza resistenza l'angolo migliore è 45° e la gittata 40.8 m (esatta 40.77 m).
2. La resistenza accorcia la gittata e abbassa l'angolo migliore.

> **Sicurezza**
>
> - Nessuna, oltre al normale uso del computer.

Alternativa a casa o virtuale

- Colab o JupyterLite.

Google Colab (notebook gratuiti) (https://colab.research.google.com/) · JupyterLite: Python nel browser (Pyodide) (https://jupyter.org/try-jupyter/lab/)

Legame con il pianeta: La resistenza è la stessa fisica che determina il consumo di carburante di un aereo (Introduzione all'ingegneria aerospaziale).

☐ **Ho completato questo laboratorio (reale o virtuale)**

Serie di esercizi per gli esami

Cronometrati: circa 10 minuti per problema. Poi confronta con le soluzioni complete.

Esercizi di preparazione all'esame intermedio (unità 1–5)

1. Cosa stampa print(7 // 2, 7 % 2)?

    1. Divisione intera e resto.

    Risposta: 3 1

2. Scrivi un ciclo che sommi da 1 a 100.

    1. total = 0; for n in range(1, 101): total += n.

    Risposta: 5050

3. Scrivi stall_speed(W, rho, S, cl_max=1.6) e valutala a 10791 N, 1.227 kg/m³, 16.2 m².

    1. (2W/(ρSC))**0.5.

    Risposta: 26.0 m/s

4. Dato co2 = [418.53, 421.08, 424.61, 427.35], calcola le variazioni anno per anno.

    1. [b - a for a, b in zip(co2, co2[1:])].

    Risposta: [2.55, 3.53, 2.74]

5. Perché float("") richiede try/except quando si leggono dati?

    1. Solleva ValueError.

    Risposta: Per saltare o registrare le righe errate invece di andare in crash.

Serie di esercizi finale (tutte le unità)

1. Converti 44700 cfs in m³/s.

    1. × 0.0283168.

    Risposta: ≈ 1,265.8 m³/s

2. Quanti passi di ricerca binaria per un milione di elementi ordinati?

    1. $\lceil \log_2 10^6 \rceil$.

    Risposta: 20

3. Quanti passi di bisezione portano [0, 2] sotto 10^{-6}?

    1. $2/2^n < 10^{-6}$.

    Risposta: 21

4. Gittata esatta di un proiettile a 20 m/s a 45° (g = 9.81)?

    1. v_0^2/g.

    Risposta: 40.77 m

5. Quattro passi di Euler di P += 0.5P(1 − P/1000) partendo da 100?

    1. 145.0, 207.0, 289.1, …

    Risposta: 391.8

6. Scrivi un test per f_to_c.

1. assert f_to_c(212) == 100, e altri.

Risposta: Coppie note: 32 → 0, 212 → 100, −40 → −40.

Formulario

$$7//2 = 3,\ 7\,\%\,2 = 1,\ 2^{10} = 1024$$

$$\bar{x} = \tfrac{1}{n}\sum x_i$$

$$\text{rate} = \Delta y / \Delta x$$

$$y \approx mx + b \ (\text{np.polyfit})$$

$$1\ \text{cfs} = 0.0283168\ \text{m}^3/\text{s}$$

$$\text{binary search} \approx \lceil \log_2 n \rceil$$

$$n_{bisect} = \lceil \log_2((b-a)/\text{tol}) \rceil$$

$$v\ +\!= a\,\Delta t,\ x\ +\!= v\,\Delta t$$

$$R = v_0^2 \sin 2\theta / g$$

$$P\ +\!= hrP(1 - P/K)$$

Costanti

Costante	Valore
1 piede cubo al secondo (cfs)	0.0283168 m³/s
Secondi in un giorno	86,400
CO_2 di Mauna Loa, media annua 1959	315.98 ppm
CO_2 di Mauna Loa, media annua 2025	427.35 ppm
Anomalia GISTEMP 2024 (rispetto al 1951–1980)	1.28 °C
GM della Terra	3.986 × 10¹⁴ m³/s²
g	9.81 m/s²

Fonte: NOAA GML: Trends in atmospheric CO_2, data (https://gml.noaa.gov/ccgg/trends/data.html)

Strumenti e riferimenti gratuiti

Esegui Python gratis nel browser e consulta la documentazione qui.

- Google Colab (notebook gratuiti) (https://colab.research.google.com/): Notebook gratuiti con NumPy e Matplotlib preinstallati.

- JupyterLite: Python nel browser (Pyodide) (https://jupyter.org/try-jupyter/lab/): Python nel browser, senza account, basato su Pyodide.
- Il tutorial ufficiale di Python (https://docs.python.org/3/tutorial/): Il tutorial di riferimento.
- MIT OpenCourseWare 6.100L: Introduction to CS and Programming Using Python (https://ocw.mit.edu/courses/6-100l-introduction-to-cs-and-programming-using-python-fall-2022/): Il corso introduttivo completo del MIT.
- Harvard CS50's Introduction to Programming with Python (https://cs50.harvard.edu/python/): Il corso Python gratuito di Harvard.

Glossario

Algoritmo
Metodo preciso passo per passo.

Array
Contenitore NumPy per calcoli numerici veloci.

Asserzione
Controllo che ferma il programma se è falso.

Big-O
Come cresce il costo con la dimensione dell'input.

Bisezione
Ricerca di radici dimezzando un intervallo.

CSV
File di valori separati da virgole.

Dizionario
Associazione chiave → valore.

Caso limite
Input insolito a un estremo.

Euler-Cromer
Passo temporale stabile: prima la velocità.

Eccezione
Errore sollevato durante l'esecuzione.

Funzione
Blocco di codice con nome e riutilizzabile.

Lista
Sequenza ordinata e mutabile.

Ciclo
Esecuzione ripetuta.

Notebook
Documento di celle di codice eseguibili.

Pyodide
Python compilato per girare in un browser web.

Simulazione
Far avanzare un modello nel tempo.

Traceback
Rapporto che indica dove si è verificato un errore.

Test unitario
Piccolo controllo di una singola funzione.

Variabile
Nome legato a un valore.

f-string
Letterale di stringa formattata.

Griglia di autovalutazione

Non si assegnano voti né crediti. Usala per valutare onestamente a che punto sei in ogni unità.

Livello	Come si presenta
4 · Padronanza	Scrive funzioni chiare e testate; valida i risultati con risposte note; spiega le scelte.
3 · Competente	I programmi funzionano, con bug occasionali trovati con i test.
2 · In sviluppo	Sa modificare esempi ma fatica a scrivere da zero.
1 · Principiante	Copia codice senza capirlo; ripassare l'unità.

Pesi suggeriti se vuoi un punteggio

Parte	Peso
Esercizi	20%
Laboratori di programmazione	35%
Serie di esercizi di metà corso	20%
Serie di esercizi finale	25%

Punta al livello 3 o superiore in ogni unità prima di passare al corso successivo.

Corsi correlati

- Disegno tecnico / CAD ENGR 120

- Analisi III MATH 210
- Chimica generale I + Laboratorio CHEM 110 + CHEM 110L
- Tutti i corsi