

CORSO GRATUITO · 4 LEZIONI

Introduzione alla programmazione

Algoritmi e pseudocodice con carta e penna, poi Python, poi i database con MySQL.

Federico Boggia
federicoboggia.it

Si può programmare senza un computer.

Il computer per un programmatore è come il telescopio per un astronomo: gli fa vedere più lontano. Il modo di pensare viene prima.

01

Pensare prima di scrivere

Algoritmi, schemi di flusso e pseudocodice.

Federico Boggia
federicoboggia.it

LA PROGRAMMAZIONE È PIÙ VECCHIA DEI COMPUTER

1843

Ada Lovelace scrive il primo algoritmo pensato per una macchina

1936

Alan Turing definisce che cosa vuol dire «calcolare»

È considerata la prima programmatrice della storia: il suo algoritmo era per la Macchina Analitica di Babbage.

Poi arrivano le macchine e i linguaggi

ANNO	CHE COSA	PERCHÉ CONTA
1945	ENIAC	Primo computer elettronico generale programmabile: si programmava collegando cavi e interruttori
1957	FORTRAN	Primo linguaggio ad alto livello di grande successo, usato ancora oggi per i calcoli scientifici
1972	C	Il linguaggio di Dennis Ritchie, che diventa uno dei più influenti di sempre
1991	**Python**	**Lo crea Guido van Rossum, pensato per essere semplice e leggibile**
1995	JavaScript	Nasce per rendere interattive le pagine web, e oggi è ovunque

In questo corso il codice vero è scritto in Python.

Una persona con foglio e matita è una macchina di Turing.

Legge un quadretto, scrive, si sposta di uno, segue il manuale. Ogni computer fa questo, a miliardi di operazioni al secondo.

La macchina in quattro pezzi

01

Il nastro

Celle con un simbolo: 0, 1 o vuoto. È la memoria, e non finisce mai.

02

La testina

Legge e scrive sul nastro, e si sposta di una cella alla volta.

03

Gli stati

Un insieme finito: «cerco uno zero», «sto sommando», «ho finito».

04

La tabella

«Se sei nello stato X e leggi Y, scrivi Z, muoviti e passa allo stato W.»

Sommare 1 a un numero binario

Parti dalla cifra più a destra. Se leggi 1 scrivi 0 e vai a sinistra: è il riporto. Se leggi 0 scrivi 1 e ti fermi.

```
Nastro iniziale:  [1][0][1][1]   1011 = 11

Passo 1: Leggo [1], scrivo [0], vado a sinistra (riporto!)
Nastro:           [1][0][1][0]

Passo 2: Leggo [1], scrivo [0], vado a sinistra (ancora riporto!)
Nastro:           [1][0][0][0]

Passo 3: Leggo [0], scrivo [1], FERMATI
Nastro finale:    [1][1][0][0]   1100 = 12
```

CONCETTO CHIAVE

Si potrebbe costruire un computer con l’acqua che scorre nei tubi.

La programmazione è logica. Il computer fisico esegue in fretta quello che a mano richiederebbe un’eternità.

Le parole di tutti i giorni

Programma

Istruzioni che il computer esegue per svolgere un compito.

Variabile

Un contenitore con un nome, che cambia valore mentre il programma gira.

Tipo di dato

Che cosa ci sta dentro: interi, decimali, testo, vero o falso.

Ciclo

Ripete un blocco finché una condizione resta vera.

Condizione

Un’espressione vera o falsa, per decidere con if ed else.

Funzione

Un blocco riutilizzabile: riceve parametri, restituisce un valore.

E il debug: trovare e correggere gli errori.

Un algoritmo è una sequenza finita di passi precisi e non ambigui.

Una ricetta di cucina è un algoritmo, e anche le istruzioni per montare un mobile.

Le sei qualità di un buon algoritmo

Finito: termina dopo un numero finito di passi

Definito: ogni passo è preciso e non ambiguo

Input: riceve zero o più dati in ingresso

Output: produce almeno un risultato

Efficace: ogni operazione è abbastanza semplice da eseguire

Generale: funziona per tutti i casi del problema

La moka, in pseudocodice

```
INIZIO
    1. Riempi la moka con acqua fino alla valvola
    2. Inserisci il filtro nella moka
    3. Aggiungi caffè macinato nel filtro
    4. Avvita la parte superiore della moka
    5. Posiziona la moka sul fornello
    6. Accendi il fornello a fuoco medio
    7. MENTRE il caffè non è salito
        - Attendi
    8. Spegni il fornello
    9. Versa il caffè nella tazza
FINE
```

I simboli dei flowchart

Ovale

Inizio e fine dell'algoritmo.

Rettangolo

Un processo: un calcolo, un'assegnazione.

Rombo

Una decisione: sì o no, vero o falso.

Parallelogramma

Input e output: i dati che entrano, i risultati che escono.

Freccia

Il flusso, da un passo a quello dopo.

Cerchio

Un connettore, quando il diagramma è grande.

Prova tu: il flowchart per entrare al cinema, maggiorenni o accompagnati.

Prima lo pseudocodice, poi il codice.

In aula c’è sempre chi va dritto al codice, si impianta a metà e torna indietro a scrivere lo pseudocodice.

Le strutture di base

01

Sequenza

Le istruzioni una dopo l’altra, in ordine.

02

Selezione

SE età >= 18 ALLORA «Puoi votare», ALTRIMENTI «Non puoi ancora votare».

03

Iterazione

MENTRE una condizione è vera, oppure PER i DA 1 A 10: si ripete.

Dallo pseudocodice a Python

Il fattoriale: $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$.

```
def calcola_fattoriale(n):
    if n < 0:
        print("Errore: il numero deve essere positivo")
        return None
    if n == 0 or n == 1:
        risultato = 1
    else:
        risultato = 1
        for i in range(2, n + 1):
            risultato = risultato * i
    print(f"Il fattoriale di {n} è: {risultato}")
    return risultato
```

02

I dati e le funzioni

Variabili, tipi, operatori e parametri, in pseudocodice e in Python.

Federico Boggia
federicoboggia.it

Una variabile è una scatola con un’etichetta.

Il nome resta, il contenuto può cambiare mentre il programma gira. Il tipo dice che cosa ci può stare dentro.

I tipi principali

TIPO	CHE COS'È	ESEMPI	SI USA PER
int	Numeri senza decimali	7, -3, 0, 1000	contatori, età, quantità
float	Numeri con la virgola	3.14, -0.5, 2.718	misure, percentuali
str	Sequenza di caratteri	"Ciao", "Roma", "123"	testi, nomi, messaggi
bool	Vero o falso	True, False	condizioni, stato
list	Sequenza ordinata	[1, 2, 3], ["a", "b"]	collezioni
dict	Coppie chiave e valore	{"nome": "Mario", "eta": 30}	strutture complesse

Python è dinamicamente tipizzato

```
numero_studenti = 25          # int
temperatura_media = 18.5      # float
nome_citta = "Livorno"        # str
iscritto = True               # bool
voti = [18, 24, 27, 30]       # list
persona = {"nome": "Anna", "eta": 22}  # dict

x = 10       # x è un int
x = "ciao"   # ora x è una stringa (possibile ma sconsigliato)
```

Nomi descrittivi, in minuscolo: totale_vendite è meglio di tv.

Sette operatori per fare i conti

OP.	NOME	ESEMPIO	RISULTATO
+	Addizione	7 + 3	10
-	Sottrazione	7 - 3	4
*	Moltiplicazione	7 * 3	21
/	Divisione	7 / 3	2.333...
//	Divisione intera	7 // 3	2
%	**Modulo, cioè il resto**	**7 % 3**	**1**
**	Potenza	2 ** 3	8

Il modulo dice se un numero è pari: n % 2 == 0.

Un uguale assegna.
Due uguali chiedono.

x = 5 mette 5 dentro x. x == 5 chiede se x vale 5. È lo scambio che in aula vedo più spesso nelle prime lezioni.

Tre operatori logici

01

and

Vero se tutte e due le condizioni sono vere. (5 > 3) and (10 > 7) dà True.

02

or

Vero se almeno una è vera. (5 > 10) or (3 < 7) dà True.

03

not

Inverte il valore. not (5 > 10) dà True.

Parametri e valori di default

Una funzione ha un nome, dei parametri in ingresso e un valore che restituisce.

```
def calcola_sconto(prezzo, percentuale):
    sconto = prezzo * (percentuale / 100)
    prezzo_finale = prezzo - sconto
    return prezzo_finale

def potenza(base, esponente=2):
    return base ** esponente

prezzo_scontato = calcola_sconto(100, 20)  # 80.0
quadrato = potenza(5)                       # 25 (usa default)
cubo = potenza(5, 3)                        # 125
```

Dove si vede una variabile

SCOPE GLOBALE

Definita fuori dalle funzioni.

Si vede ovunque nel programma.

SCOPE LOCALE

Definita dentro una funzione.

Si vede solo lì: fuori, print(valore_locale) dà errore.

La regola d’oro: parametri espliciti al posto delle variabili globali.

03

I database

MySQL: database, tabelle, tipi e vincoli.

Federico Boggia
federicoboggia.it

Come è fatto MySQL

01

Il server

Il motore che gestisce tutti i database e risponde alle richieste.

02

Il database

Il contenitore delle tabelle di un progetto.

03

La tabella

Righe e colonne: le colonne dicono il tipo, le righe portano i valori.

04

La query

L'istruzione SQL con cui la tua app parla col server.

Si comincia con XAMPP: gratuito, con dentro MySQL, Apache e phpMyAdmin.

Il primo database

```
# Da terminale/prompt dei comandi
mysql -u root -p

-- Specificare il charset (importante per caratteri accentati)
CREATE DATABASE scuola
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;

SHOW DATABASES;
USE scuola;
```

In produzione la password di root si imposta sempre, ed è forte.

Il database è il contenitore.
La tabella ci sta dentro.

Me lo chiedono sempre in aula. Il database scuola tiene gli studenti, i docenti e le classi, ognuno nella sua tabella.

La prima tabella

id è la chiave primaria: due studenti con lo stesso id non possono esistere.
AUTO_INCREMENT lo numera da sé, NOT NULL rende il campo obbligatorio.

```
CREATE TABLE studenti (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(50) NOT NULL,
    cognome VARCHAR(50) NOT NULL,
    eta INT,
    classe VARCHAR(10)
);
```

SCEGLIERE IL TIPO GIUSTO

I soldi vanno in DECIMAL, mai in FLOAT.

FLOAT è un decimale approssimato, buono per le coordinate GPS. Un prezzo vuole DECIMAL(10,2): dieci cifre in tutto, due dopo la virgola.

I vincoli tengono puliti i dati

UNIQUE vieta i doppioni, CHECK controlla una condizione, DEFAULT mette un valore quando manca.

```
CREATE TABLE utenti (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(30) NOT NULL UNIQUE,
    email VARCHAR(100) NOT NULL UNIQUE,
    password_hash VARCHAR(255) NOT NULL,
    eta INT CHECK (eta >= 13),
    attivo BOOLEAN DEFAULT 1,
    data_registrazione TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

04

Fare domande ai dati

SELECT, JOIN, INSERT, UPDATE, DELETE, e gli indici.

Federico Boggia
federicoboggia.it

Le quattro query di base

01

INSERT

Aggiunge righe nuove.

02

SELECT

Legge: filtra, ordina, raggruppa.

03

UPDATE

Cambia i valori di righe che esistono già.

04

DELETE

Toglie righe dalla tabella.

Più i JOIN, che mettono insieme i dati di più tabelle.

Una SELECT, per esteso

Dopo FROM ogni clausola è facoltativa: WHERE filtra, GROUP BY raggruppa, ORDER BY ordina, LIMIT taglia.

```
SELECT colonna1, colonna2, ...
FROM nome_tabella
WHERE condizione
GROUP BY colonna
HAVING condizione_gruppo
ORDER BY colonna ASC|DESC
LIMIT numero;
```

WHERE filtra le righe, HAVING filtra i gruppi.

WHERE lavora prima del raggruppamento, HAVING dopo, e può usare COUNT, SUM e AVG: «solo i corsi con più di 10 studenti».

Due modi di unire le tabelle

INNER JOIN

Solo le righe che si trovano in tutte e due.

Gli studenti che hanno almeno un esame, accanto alle materie e ai voti.

LEFT JOIN

Tutte le righe della tabella di sinistra.

Tutti gli studenti, anche quelli che non hanno ancora un esame.

Con gli alias, studenti AS s, la query si legge meglio.

INSERT: più righe in una volta

Con l'elenco delle colonne scrivi solo i valori che servono. Senza, li devi dare tutti.

```
INSERT INTO studenti (nome, cognome, eta, corso)
VALUES
    ('Mario', 'Rossi', 22, 'Informatica'),
    ('Laura', 'Bianchi', 21, 'Matematica'),
    ('Giuseppe', 'Verdi', 23, 'Fisica');
```

UPDATE senza WHERE cambia tutte le righe.

Capita quasi sempre a qualcuno. Prima si scrive la SELECT con la stessa WHERE: le righe che tornano sono quelle che cambierai.

Svuotare: DELETE o TRUNCATE

DELETE

Toglie le righe che scegli con WHERE.

Più lento e più flessibile. Senza WHERE le toglie tutte, e senza un backup non si torna indietro.

TRUNCATE

Svuota tutta la tabella.

Più veloce, senza WHERE, e azzera anche il contatore di AUTO_INCREMENT.

Gli indici: dritti alla riga

Sulle colonne che usi spesso in WHERE, JOIN e ORDER BY

Sulle colonne con molti valori diversi

Sulle chiavi esterne, per i JOIN

Non sulle tabelle sotto le 1.000 righe

Non sulle colonne che cambiano spesso: rallentano INSERT e UPDATE

Senza indice MySQL legge tutta la tabella, con l'indice ci arriva dritto.

Tre grafici, tre query

01

Barre

Confronti fra categorie: COUNT o SUM con GROUP BY.

02

Linea

L’andamento nel tempo: YEAR, MONTH e GROUP BY sulla data.

03

Torta

Le parti di un intero: COUNT diviso il totale, in percentuale.

Un grafico vuole i dati già aggregati: una riga per ogni barra.



CORSO GRATUITO · 4 LEZIONI

Prima i passi su carta, poi il codice e le query.

Quattro lezioni gratuite con gli esercizi, e un test finale per vedere dove sei arrivato.

Introduzione alla programmazione federicoboggia.it/corsi/prog

Tutti i corsi gratuiti federicoboggia.it/corsi

Federico Boggia
federicoboggia.it