La Logica di Programmazione

🟦 Definizione

La **logica di programmazione** è la capacità di **analizzare un problema** e **trovare una soluzione** seguendo una serie di **passaggi logici**, ordinati e coerenti, che possono essere tradotti in un linguaggio di programmazione.

> “ È la base del pensiero algoritmico: il modo con cui un programmatore “pensa” prima di scrivere codice. ”

⚙ Obiettivi della logica di programmazione

- Comprendere **come funziona un problema** prima di risolverlo.
- Suddividere il problema in **sotto-problemi più semplici**.
- Definire un **algoritmo**, cioè una sequenza finita di istruzioni che portano alla soluzione.
- Scrivere **codice chiaro, corretto e riutilizzabile**.

🧩 Concetti fondamentali

Concetto	Descrizione	Esempio
Sequenza	Le istruzioni vengono eseguite una dopo l'altra	`a = b + c; d = a * 2;`
Selezione (condizione)	Il flusso varia in base a una condizione logica	`if (x > 0) { ... } else { ... }`
Iterazione (ciclo)	Un gruppo di istruzioni si ripete fino a una condizione	`for (int i=0; i<10; i++) { ... }`
Astrazione	Suddivisione del problema in funzioni o moduli	Funzione `calcolaMedia()`

🧮 Esempio pratico

Problema: Calcolare la media di 3 numeri.

Algoritmo (in pseudocodice):

```
leggi numero1
leggi numero2
leggi numero3
somma = numero1 + numero2 + numero3
media = somma / 3
stampa media
```

Logica: sequenza → calcolo → output.
Questo stesso algoritmo può essere tradotto in **qualsiasi linguaggio** (Java, Python, PHP...).

🔍 Errori comuni

- Scrivere codice senza aver progettato la soluzione.
- Mancanza di condizioni di controllo (if, while).
- Non considerare casi limite (es. divisione per zero).
- Creare codice duplicato o poco leggibile.

🚀 Perché è importante

- Permette di **ragionare in modo strutturato**.
- Riduce il rischio di errori e bug.
- Rende più **facile apprendere nuovi linguaggi**.
- È **fondamentale per ogni programmatore**, indipendentemente dal linguaggio o dall’ambito (web, database, mobile, AI...).

🧠 Mini-esercizi

1. Scrivi un algoritmo per determinare se un numero è pari o dispari.
2. Crea un diagramma di flusso per calcolare la somma dei numeri da 1 a 100.
3. Descrivi in parole tue la differenza tra “sequenza” e “iterazione”.
4. Spiega perché la logica è importante anche prima di scrivere codice.

💬 In sintesi

> La logica di programmazione è **pensare come un computer**: analizzare, scomporre, ordinare e risolvere problemi con metodo.

Slide 4: Variabili e Tipi di Dati 📦

Cosa sono le variabili?

Le **variabili** sono **contenitori** che memorizzano informazioni che possono essere utilizzate e modificate nel programma.

Analogia: Come una scatola con un'etichetta che contiene qualcosa all'interno.

Dichiarazione di variabili

```python
# Python
nome = "Mario"
età = 30
altezza = 1.75
è_studente = True
```

```javascript
// JavaScript
let nome = "Mario";
const età = 30;
var altezza = 1.75;
let è_studente = true;
```

Tipi di Dati Fondamentali

1 Numeri (Numbers)

Interi (int/integer):

```
età = 25
anno = 2024
temperatura = -5
punteggio = 0
```

Decimali (float/double):

```
prezzo = 19.99
pi_greco = 3.14159
temperatura = 36.6
peso = 72.5
```

Operazioni numeriche:

```
somma = 10 + 5          # 15
differenza = 20 - 8     # 12
prodotto = 6 * 7        # 42
divisione = 100 / 4     # 25.0
potenza = 2 ** 3        # 8
resto = 17 % 5          # 2
```

2️⃣ Stringhe (String)

Testo tra virgolette:

```
nome = "Anna"
cognome = 'Rossi'
messaggio = "Benvenuto nel corso!"
indirizzo = "Via Roma, 123"
```

Operazioni con stringhe:

```python
# Concatenazione
nome_completo = "Anna" + " " + "Rossi"  # "Anna Rossi"

# Ripetizione
risata = "ha" * 3  # "hahaha"

# Lunghezza
lunghezza = len("Ciao")  # 4

# Maiuscole/minuscole
testo = "python"
print(testo.upper())  # "PYTHON"
print(testo.capitalize())  # "Python"

# Estrazione caratteri
prima_lettera = nome[0]  # "A"
```

3️⃣ Booleani (Boolean)

Vero o Falso:

```
è_maggiorenne = True
ha_patente = False
è_studente = True
corso_completato = False
```

Operazioni logiche:

```
# AND - entrambi devono essere veri
può_guidare = è_maggiorenne and ha_patente

# OR - almeno uno deve essere vero
ha_sconto = è_studente or età < 18

# NOT - inverte il valore
non_è_maggiorenne = not è_maggiorenne
```

4 Liste/Array

Collezione ordinata di elementi:

```python
# Python - Liste
frutti = ["mela", "banana", "arancia", "pera"]
numeri = [1, 2, 3, 4, 5]
misto = [1, "ciao", True, 3.14]

# Accesso agli elementi
primo_frutto = frutti[0]  # "mela"
ultimo = frutti[-1]  # "pera"

# Modifica
frutti[1] = "kiwi"  # Sostituisce "banana"
frutti.append("uva")  # Aggiunge alla fine

# Operazioni
lunghezza = len(frutti)  # Numero elementi
```

5 Dizionari/Oggetti

Coppie chiave-valore:

```python
# Python - Dizionario
studente = {
    "nome": "Mario",
    "età": 22,
    "corso": "Informatica",
    "media_voti": 27.5,
    "è_attivo": True
}

# Accesso ai valori
nome_studente = studente["nome"]  # "Mario"
età = studente.get("età")  # 22

# Modifica
studente["età"] = 23
studente["email"] = "mario@email.com"
```

Conversione tra tipi (Type Casting)

```python
# Da stringa a numero
età_stringa = "25"
età_numero = int(età_stringa)  # 25

# Da numero a stringa
punteggio = 100
punteggio_stringa = str(punteggio)  # "100"

# Da stringa a float
prezzo = float("19.99")  # 19.99

# Da numero a booleano
vero = bool(1)   # True
falso = bool(0)  # False
```

💡 Best Practices

- **Nomi descrittivi**: `nome_studente` invece di `x`
- **Convenzioni**: in Python usa `snake_case`, in JavaScript `camelCase`
- **Costanti**: usa MAIUSCOLE per valori che non cambiano (`PI_GRECO = 3.14`)

Slide 5: Istruzioni di Controllo 

Cos'è il flusso di controllo?

Il **controllo del flusso** permette di determinare quali istruzioni eseguire e in che ordine, basandosi su condizioni specifiche.

Analogia: Come un semaforo che decide se puoi proseguire o devi fermarti.

1. Istruzioni Condizionali: IF-ELSE

Sintassi base

```python
# Python
if condizione:
    # codice da eseguire se vero
    print("La condizione è vera")
```

IF-ELSE completo

```python
età = 18

if età >= 18:
    print("Sei maggiorenne")
    print("Puoi guidare")
else:
    print("Sei minorenne")
    print("Non puoi ancora guidare")
```

```javascript
// JavaScript
let età = 18;

if (età >= 18) {
    console.log("Sei maggiorenne");
    console.log("Puoi guidare");
} else {
```

IF-ELIF-ELSE (Condizioni multiple)

```
voto = 28

if voto >= 30:
    print("Eccellente! 🌟")
    risultato = "Lode"
elif voto >= 24:
    print("Molto bene! 👍")
    risultato = "Buono"
elif voto >= 18:
    print("Sufficiente ✓")
    risultato = "Sufficiente"
else:
    print("Non superato ✗")
    risultato = "Insufficiente"

print(f"Risultato: {risultato}")
```

Operatori di confronto

```
# Uguaglianza
5 == 5    # True (uguale)
5 != 3    # True (diverso)

# Confronto
10 > 5    # True (maggiore)
3 < 7     # True (minore)
5 >= 5    # True (maggiore o uguale)
4 <= 3    # False (minore o uguale)
```

Operatori logici

```python
# AND - entrambe le condizioni devono essere vere
età = 25
ha_patente = True

if età >= 18 and ha_patente:
    print("Puoi noleggiare un'auto")

# OR - almeno una condizione deve essere vera
è_weekend = True
è_festivo = False

if è_weekend or è_festivo:
    print("Niente lavoro oggi!")

# NOT - inverte la condizione
è_piovoso = False

if not è_piovoso:
    print("Andiamo al parco!")
```

Condizioni annidate

```
età = 20
ha_documento = True
ha_soldi = True

if età >= 18:
    print("Maggiorenne ✓")

    if ha_documento:
        print("Documento presente ✓")

        if ha_soldi:
            print("Può comprare il biglietto! 🎟️")
        else:
            print("Fondi insufficienti ✗")
    else:
        print("Documento mancante ✗")
else:
    print("Minorenne ✗")
```

2. Switch/Match (Selezione multipla)

Python 3.10+ (Match-Case)

```python
giorno = "lunedì"

match giorno:
    case "lunedì":
        print("Inizio settimana 💼")
        attività = "Riunione"
    case "mercoledì":
        print("Metà settimana 📊")
        attività = "Report"
    case "venerdì":
        print("Fine settimana! 🎉")
        attività = "Aperitivo"
    case "sabato" | "domenica":
        print("Weekend! 🏖️")
        attività = "Relax"
    case _:
        print("Giorno lavorativo standard")
        attività = "Lavoro"
```

JavaScript (Switch)

```
const giorno = "lunedì";
let attività;

switch(giorno) {
    case "lunedì":
        console.log("Inizio settimana 💼");
        attività = "Riunione";
        break;
    case "mercoledì":
        console.log("Metà settimana 📊");
        attività = "Report";
        break;
    case "venerdì":
        console.log("Fine settimana! 🎉");
        attività = "Aperitivo";
        break;
    case "sabato":
    case "domenica":
        console.log("Weekend! 🏖️");
        attività = "Relax";
        break;
    default:
        console.log("Giorno lavorativo standard");
        attività = "Lavoro";
}
```

3. Ciclo FOR (Iterazione definita)

Iterare su un range

```python
# Stampare numeri da 0 a 4
for i in range(5):
    print(f"Numero: {i}")

# Output:
# Numero: 0
# Numero: 1
# Numero: 2
# Numero: 3
# Numero: 4

# Range con inizio e fine
for i in range(1, 6):
    print(i)  # 1, 2, 3, 4, 5

# Range con step (passo)
for i in range(0, 10, 2):
    print(i)  # 0, 2, 4, 6, 8
```

Iterare su liste

```python
frutti = ["mela", "banana", "arancia", "pera"]

# Metodo 1: Iterazione diretta
for frutto in frutti:
    print(f"Mi piace la {frutto}")

# Metodo 2: Con indice
for i in range(len(frutti)):
    print(f"{i+1}. {frutti[i]}")

# Metodo 3: Con enumerate
for indice, frutto in enumerate(frutti, start=1):
    print(f"{indice}. {frutto}")
```

JavaScript FOR loop

```
// For tradizionale
for (let i = 0; i < 5; i++) {
    console.log(`Numero: ${i}`);
}

// For...of (su array)
const frutti = ["mela", "banana", "arancia"];
for (const frutto of frutti) {
    console.log(`Mi piace la ${frutto}`);
}

// forEach
frutti.forEach((frutto, indice) => {
    console.log(`${indice + 1}. ${frutto}`);
});
```

4. Ciclo WHILE (Iterazione indefinita)

Sintassi base

```
# Conta fino a 5
contatore = 1

while contatore <= 5:
    print(f"Contatore: {contatore}")
    contatore += 1

# Output:
# Contatore: 1
# Contatore: 2
# Contatore: 3
# Contatore: 4
# Contatore: 5
```

Esempio pratico: Menu interattivo

```
scelta = ""

while scelta != "q":
    print("\n=== MENU ===")
    print("1. Nuova partita")
    print("2. Carica partita")
    print("3. Opzioni")
    print("q. Esci")

    scelta = input("Scegli un'opzione: ")

    if scelta == "1":
        print("Avvio nuova partita...")
    elif scelta == "2":
        print("Caricamento partita...")
    elif scelta == "3":
        print("Apertura opzioni...")
    elif scelta == "q":
        print("Arrivederci!")
    else:
        print("Opzione non valida!")
```

WHILE con condizioni complesse

```
tentativo = 0
max_tentativi = 3
password_corretta = False

while tentativo < max_tentativi and not password_corretta:
    password = input("Inserisci la password: ")

    if password == "python123":
        print("✓ Accesso consentito!")
        password_corretta = True
    else:
        tentativo += 1
        rimanenti = max_tentativi - tentativo
        if rimanenti > 0:
            print(f"✗ Password errata. {rimanenti} tentativi rimasti.")
        else:
            print("✗ Accesso negato. Troppi tentativi falliti.")
```

5. Controllo del Ciclo: BREAK e CONTINUE

BREAK - Interrompe il ciclo

```python
# Cerca un numero in una lista
numeri = [1, 5, 8, 12, 15, 20, 25]
target = 15

for numero in numeri:
    print(f"Controllo: {numero}")
    if numero == target:
        print(f"✓ Trovato {target}!")
        break  # Esce dal ciclo
    print("Non è quello che cerco...")

# Output si ferma quando trova 15
```

CONTINUE - Salta all'iterazione successiva

```python
# Stampa solo numeri pari
for i in range(1, 11):
    if i % 2 != 0:  # Se dispari
        continue  # Salta questa iterazione
    print(f"{i} è pari")

# Output: 2, 4, 6, 8, 10
```

Esempio combinato

```python
# Trova numeri divisibili per 3 ma non per 6
for numero in range(1, 21):
    if numero % 3 != 0:
        continue  # Salta se non divisibile per 3

    if numero % 6 == 0:
        print(f"{numero} è divisibile per 6, mi fermo!")
        break

    print(f"{numero} è divisibile per 3 ma non per 6")
```

💡 Best Practices

- **Evita cicli infiniti**: assicurati che la condizione diventi False
- **Usa FOR quando conosci** il numero di iterazioni
- **Usa WHILE quando dipende** da una condizione variabile
- **Indentazione corretta**: cruciale in Python!
- **Commenta logica complessa**: rendi il codice leggibile