Lezione 5: Diventiamo Sviluppatori

Creare una Libreria per il Calcolo e il Disegno Circuitale

Laboratorio di Sistemi e Automazione

1 Obiettivo: Automatizzare il Lavoro

Nelle lezioni precedenti abbiamo scritto codice "usa e getta". Oggi faremo un salto di qualità: creeremo un nostro **programma modulare** (una piccola "libreria") che fa due cose automaticamente:

1. Calcola che batteria serve per alimentare un carico (Legge di Ohm).
2. Disegna lo schema elettrico corrispondente usando Python.

Procederemo per piccoli passi (Step-by-Step).

2 Passo 1: La Funzione Matematica

Invece di riscrivere la formula $V = R \cdot I$ ogni volta, creiamo una **Funzione**. In Python, una funzione è come un "componente integrato": ha dei pin di ingresso (argomenti) e un pin di uscita (return).

Attenzione all'Indentazione!

In Python gli spazi sono fondamentali. Il codice dentro la funzione deve essere spostato a destra (tasto TAB). Se non lo fate, otterrete `IndentationError`.

```python
# Definizione della funzione (Il nostro componente software)
def calcola_batteria(corrente_mA, resistenza_kOhm):
    """
    Input: Corrente in mA, Resistenza in kOhm
    Output: Tensione in Volt
    """

    # Calcolo la tensione (Nota: i "kilo" e i "milli" si annullano)
    tensione = corrente_mA * resistenza_kOhm

    # Restituisco il risultato (Pin di uscita)
    return tensione
```

Listing 1: Definizione della funzione di calcolo

Ora, ogni volta che scriveremo `calcola_batteria(2, 1)`, Python ci risponderà `2`.

3 Passo 2: Il Disegno Tecnico (Schemdraw)

Python può disegnare circuiti per noi usando una libreria chiamata **Schemdraw**. Poiché non è installata di base su Colab, la prima riga del codice deve scaricarla.

```
# 1. Installiamo la libreria (Comando di sistema con !)
!pip install schemdraw

# 2. Importiamo i moduli necessari
import schemdraw
import schemdraw.elements as elm

# 3. Creiamo la funzione che disegna
def disegna_circuito(tensione, resistenza):
    print(f"Sto disegnando il circuito: {tensione}V su {resistenza}kOhm...")

    # Apriamo la "tela" da disegno
    with schemdraw.Drawing() as d:
        # Aggiungiamo la batteria (SourceV)
        d.add(elm.SourceV().label(f'{tensione}V'))

        # Aggiungiamo il resistore a destra
        # NOTA: Usiamo rf'...' per evitare warning sul simbolo Omega
        d.add(elm.Resistor().right().label(rf'{resistenza} k$\Omega$'))

        # Chiudiamo il circuito tornando all'origine
        d.add(elm.Line().down()) # Scendiamo
        d.add(elm.Line().left()) # Torniamo a sinistra (chiuso!)
```

Listing 2: Setup e Funzione di Disegno

4 Passo 3: Mettiamo tutto insieme

Ora creiamo il "Main Program". L'utente inserisce i dati e il sistema calcola la batteria necessaria e genera lo schema.

```
# --- INPUT UTENTE ---
I_target = 5.0  # Vogliamo 5 mA
R_carico = 2.2  # Resistore da 2.2 kOhm

# --- FASE 1: CALCOLO ---
# Richiamiamo la nostra funzione
V_necessaria = calcola_batteria(I_target, R_carico)

print(f"Per avere {I_target} mA su {R_carico} kOhm...")
print(f"Serve una batteria da: {V_necessaria} Volt")

# --- FASE 2: DISEGNO ---
# Richiamiamo la funzione grafica
disegna_circuito(V_necessaria, R_carico)
```

Listing 3: Il Programma Finale

5 Cosa otterrete su Colab

Eseguendo il codice, Python genererà automaticamente un'immagine simile a questa:

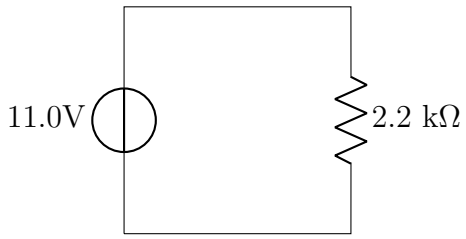

Fig. 1: Schema elettrico generato automaticamente dal codice.

→ LABORATORIO PRATICO

Diventa uno sviluppatore Python.
Clicca qui per aprire il file `CorsoPython_5.ipynb`:

[APRI GENERATORE SCHEMI]

→ CONTINUA IL PERCORSO

Finora abbiamo visto solo linee rette (Resistori).
Ma cosa succede se il componente **non è lineare**?

Vai alla Lezione 6: Il Diodo e la Non-Linearità