Corso Pratico Python

[Modulo 1- 2026]

Esercitazione Pratica

Esercizio: Simulatore Termico Primavera 2026

Richiedente: Dott. Roberto Felici [CNR-ISM] • Insegnante: Dott. Scigè J. Liu [INAF-IAPS]

Simulatore Termico Primavera 2026

Obiettivo: Creare un algoritmo che emuli il passaggio di una giornata tipo, campionando temperature simulate.

Specifiche del Modello:

- **Tempo Reale:** Il programma deve girare per **25 secondi** totali [Esecuzione].
- **Frequenza:** Un campionamento ogni **0.5 secondi** (funzione time.sleep).
- **Simulazione Temporale:** Coprire l'arco di 24 ore (dalle 00:00 alle 24:00) rapportate ai 25 secondi di esecuzione.
- **Dati:** Generare una temperatura "quasi random" tra **15°C e 25°C** (tipica primaverile).
- **Output richiesto:** Print formattata con datetime:
- Formato: %Y%m%d %H:%M:00 + Valore Temperatura.

```python
import time
import random
from datetime import datetime, timedelta

# Parametri di simulazione
start_time = datetime(2026, 3, 21, 0, 0)
secondi_simulati_per_step = (24 * 3600) / 50 # 50 step in 25 sec

for i in range(50):
    # Calcolo tempo simulato
    current_sim = start_time + timedelta(seconds=i * secondi_simulati_per_step)

    # Generazione temperatura (random range 15-25)
    temp = random.uniform(15, 25)

    # Print formattata (Punto 4.1.2)
    print(f"{current_sim.strftime('%Y%m%d %H:%M:00')} | Temp: {temp:.2f}°C")

    # Pausa reale (0.5 secondi)
    time.sleep(0.5)
```

Simulatore Termico Primavera 2026

Obiettivo: Creare un algoritmo che emuli il passaggio di una giornata tipo, campionando temperature simulate.

```python
import time
import random
from datetime import datetime, timedelta

# Configurazione file e parametri
file_name = "simulazione_termica.csv"
start_time = datetime(2026, 3, 21, 0, 0)
step_totali = 50
delta_simulato = (24 * 3600) / step_totali

# Apertura file in modalità scrittura ('w')
with open(file_name, "w", encoding="utf-8") as f:
    # Scrittura Intestazione CSV
    f.write("Timestamp,Temperatura_C\n")

    for i in range(step_totali):
        # Logica temporale e simulazione
        curr_sim = start_time + timedelta(seconds=i * delta_simulato)
        temp = random.uniform(15, 25)

        # Formattazione riga [cite: 46]
        riga = f"{curr_sim.strftime('%Y%m%d %H:%M:00')},{temp:.2f}\n"

        # Output doppio: Console + Disco [cite: 45, 57]
        print(riga.strip())
        f.write(riga)

        time.sleep(0.5) # Ritardo reale 500ms

print(f"\nSimulazione completata. Dati salvati in: {file_name}")
```

```python
import matplotlib.pyplot as plt

# Liste per il plotting
tempi_plot = []
temperature_plot = []

# --- Dentro il ciclo for dell'esercizio precedente ---
    tempi_plot.append(curr_sim)
    temperature_plot.append(temp)
# ----------------------------------------------------

# Creazione del grafico (Punto 3.1)
plt.figure(figsize=(10, 6))
plt.plot(tempi_plot, temperature_plot, color='orange', linewidth=2, label='Temp.

# Personalizzazione e Label (Punto 3.2)
plt.title("Simulazione Termica Giornaliera - Primavera 2026", fontsize=14)
plt.xlabel("Ora del Giorno", fontsize=12)
plt.ylabel("Temperatura (°C)", fontsize=12)
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()

# Ottimizzazione asse delle date
plt.gcf().autofmt_xdate()

# Salvataggio e visualizzazione [cite: 56]
plt.savefig("andamento_termico.png")
plt.show()
```

Simulatore Termico Primavera 2026

Riflessioni:

a) Separazione in programmi che lavorano «indipendentemente»

b) Conservazione dei Plots in «giornale-storico-stampabile»

c) Conservazione dei dati in «giornale-storico-binario»
   a) OpenData
   b) Formati Speciali

d) «Armonizzazione» → «Integrazione»

Generazione (Logica)
→ **Output** (Console)
→**Archiviazione** (CSV)
→ **Visualizzazione** (Plot).