🧠 Company Performance Case Study

Title: Evaluate Best-Performing Stocks and Industries
Audience: Python Academy Students (beginners learning control flow, functions, loops, and data structures)
Goal: Build a console-based Python program that analyzes stock performance over time.

🎯 Learning Objectives

By completing this project, students will learn to:

- Parse tabular (CSV or text) data and validate inputs.
- Write functions to compute **percentage returns** and perform **aggregations**.
- Use **loops**, **lists**, and **dictionaries** to collect and summarize data.
- Apply **sorting** and **grouping** logic to rank and analyze stocks.
- Present results in a **clean console report** and optionally export them to a CSV file.

📄 Dataset Schema

Your program should accept a **CSV file** (or input lines) with the following columns:

Column	Description	Type
Stock	Ticker or company name	string
Sector	Industry or sector name	string
PriceStart	Price at start of period	float
PriceEnd	Price at end of period	float

Notes:

- PriceStart and PriceEnd are **required** and must be **> 0**.
- Handle invalid or missing values gracefully (skip or flag).
- Dataset should contain **10–50 rows**.

📊 Sample Dataset (stocks_sample.csv)

Stock,Sector,PriceStart,PriceEnd

AAPL,Technology,120.50,175.20

MSFT,Technology,210.10,280.50

WMT,Consumer Staples,140.00,150.00

TSLA,Automotive,600.00,880.00

JPM,Financials,95.00,105.00

BABA,Consumer Discretionary,180.00,160.00

RELIANCE,Energy,2200.00,2400.00

TCS,Technology,3100.00,3500.00

HDFC,Financials,1500.00,1650.00

BAJAJ-AUTO,Automotive,4800.00,5200.00

🧩 Core Tasks

1. Read Input Data

- Read the CSV file
- Validate each row:
  - Check that PriceStart and PriceEnd are numeric and > 0.
- Skip or flag invalid rows (don't crash the program).

---

2. Compute Returns

For each valid stock:

$$\text{Return (\%)} = \frac{(\text{PriceEnd} - \text{PriceStart})}{\text{PriceStart}} \times 100$$

- Store results as structured dictionaries:

{'Stock': 'AAPL', 'Sector': 'Technology', 'PriceStart': 120.5, 'PriceEnd': 175.2, 'Return': 45.39}

---

3. Rank Stocks

- Sort all stocks **by Return (%) descending**.
- Display the **Top 5** best-performing stocks.

---

4. Aggregate by Sector

- Compute:
  - **Average return per sector**
  - **Number of stocks per sector**
- Identify the **best-performing sector** (highest average return).

---

5. Output Results

Print a clear, formatted console report:

- **All Stock Details:**
  Stock | Sector | PriceStart | PriceEnd | Return(%)
- **Top-5 Stocks** by return
- **Per-Sector Summary:**
- Sector | Avg Return(%) | Count
- Highlight the **Best Sector**
- (Optional) Export results to stock_returns.csv

⚙️ **Suggested Program Structure**

Function	Description
read_csv_safe(filepath)	Reads CSV, validates data, and returns a list of valid rows (dicts).
compute_return(row)	Computes and returns the percentage return for one stock.
process_all(rows)	Applies compute_return to all rows, returns list of results.
aggregate_by_sector(results)	Computes average return and count per sector.
print_report(results, sector_summary)	Prints formatted console tables.
export_csv(results, filename)	Writes results to an output CSV file.

Error Handling

- Use try/except for:
  - ValueError (invalid numeric conversion)
  - FileNotFoundError (missing CSV file)
- Skip invalid rows but continue processing.
