FOUNDATION LESSON

WEEK 4 · LESSON 10

Filter and sort an explicit report

Filtering retains labels; sorting and resetting are separate choices

THE IDEA BEHIND THE OPERATION

MENTAL MODEL

The idea behind the operation

Filtering retains labels; sorting and resetting are separate choices

COMBINE BUSINESS CONDITIONS

CODE WALKTHROUGH W04-L10-S03

```
01  import pandas as pd
02  df = pd.DataFrame({"product":["A","B","C"],
03                     "units":[5,0,8], "price":[10.0,4.0,12.0]})
04  mask = (df["units"] > 0) & (df["price"] <= 10)
05  print(df.loc[mask, ["product","units"]].to_string(index=False))
```

WHAT TO TRACK

Require positive units and price at most EUR 10.

PREDICT THE RESULT

Predict the output.
Explain each operation.
Pause before continuing.

COMBINE BUSINESS CONDITIONS — RESULT

CODE WALKTHROUGH W04-L10-S04

```
01  import pandas as pd
02  df = pd.DataFrame({"product":["A","B","C"],
03                     "units":[5,0,8], "price":[10.0,4.0,12.0]})
04  mask = (df["units"] > 0) & (df["price"] <= 10)
05  print(df.loc[mask, ["product","units"]].to_string(index=False))
```

WHAT TO TRACK

Require positive units and price at most EUR 10.

VALIDATE THE RESULT

```
product  units
      A      5
```

SORT, RESET AND ASSIGN DELIBERATELY

CODE WALKTHROUGH W04-L10-S05

```
01  import pandas as pd
02  df = pd.DataFrame({"product":["B","A","C"], "units":[5,5,2]}, i
↪   ndex=[7,3,9])
03  report = df.sort_values(["units","product"], ascending=[False,T
↪   rue])
04  print("Original labels:", report.index.tolist())
05  report = report.reset_index(drop=True)
06  report.loc[report["units"] == 2, "units"] = 0
07  print(report.to_string(index=False))
```

WHAT TO TRACK

Quantity descending; product ascending breaks ties.

PREDICT THE RESULT

Predict the output.
Explain each operation.
Pause before continuing.

SORT, RESET AND ASSIGN DELIBERATELY — RESULT

CODE WALKTHROUGH W04-L10-S06

```
01  import pandas as pd
02  df = pd.DataFrame({"product":["B","A","C"], "units":[5,5,2]}, i
↪   ndex=[7,3,9])
03  report = df.sort_values(["units","product"], ascending=[False,T
↪   rue])
04  print("Original labels:", report.index.tolist())
05  report = report.reset_index(drop=True)
06  report.loc[report["units"] == 2, "units"] = 0
07  print(report.to_string(index=False))
```

WHAT TO TRACK

Quantity descending; product ascending breaks ties.

VALIDATE THE RESULT

```
Original labels: [3, 7, 9]
product  units
      A      5
      B      5
      C      0
```

UNKNOWN FILTER DECISIONS NEED A POLICY

CODE WALKTHROUGH W04-L10-S07

```
01  import pandas as pd
02  units = pd.Series([5, None, 0], dtype="Int64")
03  mask = (units > 0).fillna(False)
04  print("Missing quantities:", int(units.isna().sum()))
05  print("Kept:", units[mask].tolist())
```

WHAT TO TRACK

Unknown quantities are excluded from this report and counted.

PREDICT THE RESULT

Predict the output.
Explain each operation.
Pause before continuing.

UNKNOWN FILTER DECISIONS NEED A POLICY — RESULT

CODE WALKTHROUGH W04-L10-S08

```
01  import pandas as pd
02  units = pd.Series([5, None, 0], dtype="Int64")
03  mask = (units > 0).fillna(False)
04  print("Missing quantities:", int(units.isna().sum()))
05  print("Kept:", units[mask].tolist())
```

WHAT TO TRACK

Unknown quantities are excluded from this report and counted.

VALIDATE THE RESULT

```
Missing quantities: 1
Kept: [5]
```

GUIDED MODIFICATION

LEARNER PRACTICE

Guided modification

Filter products with units at least three, sort units descending and break ties by product code ascending. Reset the display index and print the result.

PREDICT · MODIFY · RUN

01 Restate the supplied rules.

02 Predict the result.

03 Modify and run the code.

04 Compare actual output.

05 Explain any difference.

INDEPENDENT APPLICATION

LEARNER PRACTICE

Independent application

Create an independent copy of rows with positive units. Set units below three to three inside the copy using loc. Print both original and adjusted tables to prove the original is unchanged.

PREDICT · MODIFY · RUN

01 Restate the supplied rules.

02 Predict the result.

03 Modify and run the code.

04 Compare actual output.

05 Explain any difference.

RECAP AND NEXT STEP

Recap and next step

State filter boundaries and missing policies, sort with tie-breakers and reset labels deliberately. Use one-step assignment. Next, we create calculated columns and parse dates with an explicit format.

UNDERSTAND APPLY VALIDATE EXPLAIN