FOUNDATION LESSON

WEEK 2 · LESSON 17

Batch processing and comprehensions

Dictionary and set comprehensions / Week 2 batch transaction processing

BATCH PROCESSING AND COMPREHENSIONS

MENTAL MODEL

Batch processing and comprehensions

Set comprehensions deduplicate; dictionary comprehensions create key-value mappings.
Combine validation, iteration and accumulation in a small transaction batch.
Do not rely on set iteration order. Exclude invalid amounts before adding to a total
and count flagged transactions with the intended boundary.

CLEAN UNIQUE NAMES

CODE WALKTHROUGH W02-L17-S03

```
01  names = [" alice ", "BOB", "Alice"]
02  clean = {name.strip().title() for name in names}
03  lookup = {name: len(name) for name in sorted(clean)}
04  print(sorted(clean))
05  print(lookup)
```

WHAT TO TRACK

Clean unique names

PREDICT THE RESULT

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

CLEAN UNIQUE NAMES

CODE WALKTHROUGH W02-L17-S04

```
01  names = [" alice ", "BOB", "Alice"]
02  clean = {name.strip().title() for name in names}
03  lookup = {name: len(name) for name in sorted(clean)}
04  print(sorted(clean))
05  print(lookup)
```

WHAT TO TRACK

Clean unique names

VALIDATE THE RESULT

```
['Alice', 'Bob']
{'Alice': 5, 'Bob': 3}
```

VALIDATE BATCH TOTALS

CODE WALKTHROUGH W02-L17-S05

```
transactions = [120, 2500, 0, -5, 75, 3200]
valid = []
flagged = 0
total = 0
for amount in transactions:
    if amount <= 0:
        continue
    valid.append(amount)
    total += amount
    if amount >= 2000:
        flagged += 1
print(valid)
print(total, flagged)
```

WHAT TO TRACK

Validate batch totals

PREDICT THE RESULT

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

VALIDATE BATCH TOTALS

CODE WALKTHROUGH W02-L17-S06

```
transactions = [120, 2500, 0, -5, 75, 3200]
valid = []
flagged = 0
total = 0
for amount in transactions:
    if amount <= 0:
        continue
    valid.append(amount)
    total += amount
    if amount >= 2000:
        flagged += 1
print(valid)
print(total, flagged)
```

WHAT TO TRACK

Validate batch totals

VALIDATE THE RESULT

```
[120, 2500, 75, 3200]
5895 2
```

SEMANTIC SET CHECK

CODE WALKTHROUGH W02-L17-S07

```
01  names = [" alice ", "BOB", "Alice"]
02  clean = {name.strip().title() for name in names}
03  lookup = {name: len(name) for name in sorted(clean)}
04  print(clean == {"Alice", "Bob"})
05  print(lookup == {"Alice": 5, "Bob": 3})
```

WHAT TO TRACK

Semantic set check

PREDICT THE RESULT

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

SEMANTIC SET CHECK

CODE WALKTHROUGH W02-L17-S08

```
01  names = [" alice ", "BOB", "Alice"]
02  clean = {name.strip().title() for name in names}
03  lookup = {name: len(name) for name in sorted(clean)}
04  print(clean == {"Alice", "Bob"})
05  print(lookup == {"Alice": 5, "Bob": 3})
```

WHAT TO TRACK

Semantic set check

VALIDATE THE RESULT

```
True
True
```

INDEPENDENT PREDICTION

LEARNER PRACTICE

Independent prediction

Change one input in batch processing and comprehensions and test a boundary or edge case.

01 Predict the exact output.
02 Change only one input.
03 Run in a fresh Python process.
04 Explain any difference.

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.

BATCH PROCESSING AND COMPREHENSIONS

Recap and next step

Set comprehensions deduplicate; dictionary comprehensions create key-value mappings. Combine validation, iteration and accumulation in a small transaction batch. Do not rely on set iteration order. Exclude invalid amounts before adding to a total and count flagged transactions with the intended boundary.

UNDERSTAND APPLY VALIDATE EXPLAIN