FOUNDATION LESSON

WEEK 2 · LESSON 14

Accumulate safely

Accumulation patterns / Conditional logic inside loops

ACCUMULATE SAFELY

MENTAL MODEL

Accumulate safely

Initialise totals and counts before the loop; update them for each qualifying item.
Guard operations that need a non-empty sequence.
Initialising a maximum to zero fails for an all-negative list. Seed it from the first item after an empty-data check.

TOTAL AND COUNT

CODE WALKTHROUGH W02-L14-S03

```
01  values = [10, 20, 30, 40]
02  total = 0
03  count = 0
04  for value in values:
05      total += value
06      count += 1
07  print(total, count)
```

WHAT TO TRACK

Total and count

PREDICT THE RESULT

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

TOTAL AND COUNT

CODE WALKTHROUGH W02-L14-S04

```
01  values = [10, 20, 30, 40]
02  total = 0
03  count = 0
04  for value in values:
05      total += value
06      count += 1
07  print(total, count)
```

WHAT TO TRACK

Total and count

VALIDATE THE RESULT

```
100 4
```

INCLUSIVE FLAG RULE

CODE WALKTHROUGH W02-L14-S05

```
01  amounts = [100, 2500, 500, 4000]
02  flagged = 0
03  for amount in amounts:
04      if amount >= 2000:
05          flagged += 1
06  print(flagged)
```

WHAT TO TRACK

Inclusive flag rule

PREDICT THE RESULT

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

INCLUSIVE FLAG RULE

CODE WALKTHROUGH W02-L14-S06

```
01  amounts = [100, 2500, 500, 4000]
02  flagged = 0
03  for amount in amounts:
04      if amount >= 2000:
05          flagged += 1
06  print(flagged)
```

WHAT TO TRACK

Inclusive flag rule

VALIDATE THE RESULT

2

NEGATIVE MAXIMUM

CODE WALKTHROUGH W02-L14-S07

```
01  values = [-5, -2, -9]
02  if values:
03      largest = values[0]
04      for value in values:
05          if value > largest:
06              largest = value
07      print(largest)
08  else:
09      print("no maximum: empty input")
```

WHAT TO TRACK

Negative maximum

PREDICT THE RESULT

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

NEGATIVE MAXIMUM

CODE WALKTHROUGH W02-L14-S08

```
values = [-5, -2, -9]
if values:
    largest = values[0]
    for value in values:
        if value > largest:
            largest = value
    print(largest)
else:
    print("no maximum: empty input")
```

WHAT TO TRACK

Negative maximum

VALIDATE THE RESULT

-2

EMPTY MAXIMUM

CODE WALKTHROUGH W02-L14-S09

```
01  values = []
02  if values:
03      largest = values[0]
04      for value in values:
05          if value > largest:
06              largest = value
07      print(largest)
08  else:
09      print("no maximum: empty input")
```

WHAT TO TRACK

Empty maximum

PREDICT THE RESULT

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

EMPTY MAXIMUM

CODE WALKTHROUGH W02-L14-S10

```
01  values = []
02  if values:
03      largest = values[0]
04      for value in values:
05          if value > largest:
06              largest = value
07      print(largest)
08  else:
09      print("no maximum: empty input")
```

WHAT TO TRACK

Empty maximum

VALIDATE THE RESULT

no maximum: empty input

GROUP TOTALS

CODE WALKTHROUGH W02-L14-S11

```
01  records = [{"route": "GR", "amount": 100}, {"route": "CY", "amo
↪   unt": 50}, {"route": "GR", "amount": 20}]
02  totals = {}
03  for record in records:
04      route = record["route"]
05      totals[route] = totals.get(route, 0) + record["amount"]
06  for route in sorted(totals):
07      print(route, totals[route])
```

WHAT TO TRACK

Group totals

PREDICT THE RESULT

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

GROUP TOTALS

CODE WALKTHROUGH W02-L14-S12

```
records = [{"route": "GR", "amount": 100}, {"route": "CY", "amo
unt": 50}, {"route": "GR", "amount": 20}]
totals = {}
for record in records:
    route = record["route"]
    totals[route] = totals.get(route, 0) + record["amount"]
for route in sorted(totals):
    print(route, totals[route])
```

WHAT TO TRACK

Group totals

VALIDATE THE RESULT

```
CY 50
GR 120
```

INDEPENDENT PREDICTION

LEARNER PRACTICE

Independent prediction

Change one input in accumulate safely 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.

ACCUMULATE SAFELY

Recap and next step

Initialise totals and counts before the loop; update them for each qualifying item. Guard operations that need a non-empty sequence. Initialising a maximum to zero fails for an all-negative list. Seed it from the first item after an empty-data check.

UNDERSTAND APPLY VALIDATE EXPLAIN