📚 Module 4: Basic Operations - Complete Notes

🎯 What You'll Learn

In this module, you'll master **addition, subtraction, and the patterns they create** — accumulators, counters, running totals, and delta tracking.

📖 Concept Explained (Like a YouTube Video)

The Four Core Patterns

```
┌──────────────────────────────────────────────────────┐
│  PATTERN          │  WHAT IT DOES                    │
├──────────────────────────────────────────────────────┤
│  Accumulator      │  Add up values into a total      │
│  Counter          │  Count how many times something happens│
│  Running Balance  │  Track value after each change   │
│  Delta            │  Measure the difference between values│
└──────────────────────────────────────────────────────┘
```

💻 Programming Connection

Code Examples

```
# Example 1: Accumulator (Sum of All)
def calculate_total(prices):
    total = 0
    for price in prices:
        total += price
    return total
```

```
# Example 2: Counter
def count_above_threshold(values, threshold):
    count = 0
    for value in values:
        if value > threshold:
            count += 1
    return count
```

```
# Example 3: Running Balance
def track_transactions(initial_balance, transactions):
    balance = initial_balance
    history = []
    for amount in transactions:
        balance += amount
```

```
        history.append(balance)
    return history
```

```
# Example 4: Delta (Change Measurement)
def calculate_change(before, after):
    delta = after - before
    percent_change = (delta / before) * 100 if before != 0 else 0
    return {"delta": delta, "percent": percent_change}
```

🧪 SDET/Testing Application

```
# SDET Scenario: Test Results Counter

def analyze_test_results(results):
    counts = {"pass": 0, "fail": 0, "skip": 0}
    for result in results:
        counts[result["status"]] += 1
    total = sum(counts.values())
    pass_rate = (counts["pass"] / total * 100) if total > 0 else 0
    return {**counts, "pass_rate": f"{pass_rate:.2f}%"}
```

🔑 Key Takeaways

✅ **Accumulator** — Start at 0, add values in loop
✅ **Counter** — Start at 0, add 1 when condition is true
✅ **Delta** — `new - old` gives you the change
✅ **Always initialize before the loop**

Next: Module 5 - Advanced Operations (Multiplication, Division, Remainder)! 🚀