📚 Module 23: Set Operations - Complete Notes

🎯 What You'll Learn

Master **union, intersection, difference** — combining and comparing collections.

📖 Concept Explained

```
A = {1, 2, 3, 4}
B = {3, 4, 5, 6}

UNION (A ∪ B): All unique from both
    {1, 2, 3, 4, 5, 6}

INTERSECTION (A ∩ B): Common only
    {3, 4}

DIFFERENCE (A − B): In A but not B
    {1, 2}

SYMMETRIC DIFF (A △ B): In one but not both
    {1, 2, 5, 6}
```

💻 Programming Connection

Code Examples

```python
# Example 1: All Set Operations

A = {1, 2, 3, 4}
B = {3, 4, 5, 6}

print(f"Union: {A | B}")         # {1, 2, 3, 4, 5, 6}
print(f"Intersection: {A & B}") # {3, 4}
print(f"Difference A-B: {A - B}")  # {1, 2}
print(f"Difference B-A: {B - A}")  # {5, 6}
print(f"Symmetric: {A ^ B}")    # {1, 2, 5, 6}
```

```python
# Example 2: Compare Two Lists

def compare_lists(expected, actual):
    """Find common, missing, and extra items"""
    expected_set = set(expected)
    actual_set = set(actual)

    return {
        "common": expected_set & actual_set,
```

```
        "missing": expected_set - actual_set,
        "extra": actual_set - expected_set
    }

result = compare_lists(['a', 'b', 'c'], ['b', 'c', 'd'])
print(result)
# {'common': {'b', 'c'}, 'missing': {'a'}, 'extra': {'d'}}
```

```
# Example 3: Filter Allowed Fields

allowed = {"name", "email", "phone"}

def filter_fields(data):
    """Keep only allowed fields"""
    return {k: v for k, v in data.items() if k in allowed}

input_data = {"name": "John", "email": "j@t.com", "password": "123"}
print(filter_fields(input_data))
# {'name': 'John', 'email': 'j@t.com'}
```

🧪 SDET/Testing Application

```
# SDET Scenario: Compare Test Runs

def compare_test_runs(baseline_tests, current_tests):
    """Analyze changes between test runs"""
    baseline = set(baseline_tests)
    current = set(current_tests)

    return {
        "unchanged": baseline & current,
        "removed": baseline - current,
        "added": current - baseline,
        "total_change": len(baseline ^ current)
    }

baseline = ["test_a", "test_b", "test_c"]
current = ["test_b", "test_c", "test_d"]
print(compare_test_runs(baseline, current))
# {'unchanged': {'test_b', 'test_c'}, 'removed': {'test_a'}, 'added': {'test_d'}}
```

🔑 Key Takeaways

✅ **Union (|)** — All unique from both
✅ **Intersection (&)** — Only common
✅ **Difference (-)** — In first, not second
✅ **Symmetric (^)** — In one OR other, not both

💾 *Save as:* `Module_23_Set_Operations.md`