Day 1: Workshop

พื้นฐาน **Automated Testing** และการตั้งค่า

---

1. Workshop: Introduction to Automated Testing

- **กิจกรรม**:
    - สร้างโปรเจกต์ Python เพื่ออธิบายความสำคัญของ Automated Testing
    - เขียนสคริปต์ Python ที่รันคำสั่ง `assert` เพื่อเปรียบเทียบผลลัพธ์ที่คาดหวังกับผลลัพธ์จริง

1. Simple Assertion
2. Mathematical Functions (Addition, Multiplication, Division)
3. List Operations (Add/Remove Items)
4. String Manipulation (Capitalize, Reverse String)
5. Date and Time Operations
6. Validation Functions (Email Validation)
7. Word Operations (Count Words, Check for Vowels)
8. List Indexing
9. Max/Min Operations
10. Comparison Functions (Greater, Equal)
11. Substring Search
12. File Operations (Read/Write)
13. Unit Conversion (Celsius/Fahrenheit)
14. Number Operations (Even/Odd)
15. Splitting Strings
16. Financial Calculations (Interest Calculation)
17. Error Handling (Custom Errors)
18. Password Generation
19. Character Count
20. Sorting

ตัวอย่างที่ 1# Example: Simple Assertion (File Ex1)

```
def add(a, b):
    return a + b

try:
    assert add(2, 3) == 6, "Test Failed"
except AssertionError as e:
    print("AssertionError:", e)
```

Aj.Tar(SE)

ตัวอย่างที่ **2:** การทดสอบฟังก์ชันการคำนวณ **(Mathematical Functions)** (File Ex2)

```
# ตัวอย่างที่ 2: การทดสอบฟังก์ชันการคำนวณ (Mathematical Functions) (File Ex2)

def multiply(a, b):
    return a * b

def divide(a, b):
    if b == 0:
        raise ValueError("Division by zero is not allowed")
    return a / b

# ทดสอบฟังก์ชัน multiply
print("\nTesting multiply function:")
try:
    assert multiply(2, 3) == 6, "Test Passed"
    print("Test 1 Passed: multiply(2, 3) = 6")
except AssertionError:
    print("Test 1 Failed")

try:
    assert multiply(0, 10) == 0, "Test Passed"
    print("Test 2 Passed: multiply(0, 10) = 0")
except AssertionError:
    print("Test 2 Failed")

# ทดสอบฟังก์ชัน divide
print("\nTesting divide function:")
try:
    assert divide(10, 2) == 5, "Test Passed"
    print("Test 3 Passed: divide(10, 2) = 5")
except AssertionError:
    print("Test 3 Failed")

try:
    divide(10, 0)
except ValueError as e:
    try:
        assert str(e) == "Division by zero is not allowed", "Test Passed"
        print("Test 4 Passed: divide(10, 0) raised 'Division by zero is not allowed'")
    except AssertionError:
        print("Test 4 Failed: Exception message mismatch")
```

Aj. Tatar(SE)

ตัวอย่างที่ **3:** การทดสอบฟังก์ชันจัดการรายการ **(List Operations)**

```
#ตัวอย่างที่ 3: การทดสอบฟังก์ชันจัดการรายการ (List Operations)


def add_to_list(lst, item):
    lst.append(item)
    return lst

def remove_from_list(lst, item):
    if item in lst:
        lst.remove(item)
    return lst

print("\nList Operations Tests:")
my_list = [1, 2, 3]
# ทดสอบฟังก์ชัน add_to_list
try:
    assert add_to_list(my_list, 4) == [1, 2, 3, 4]
    print("Test 1 Passed: Added 4 to [1, 2, 3]")
except AssertionError:
    print("Test 1 Failed")

# ทดสอบฟังก์ชัน remove_from_list
try:
    assert remove_from_list(my_list, 2) == [1, 3, 4]
    print("Test 2 Passed: Removed 2 from [1, 3, 4]")
except AssertionError:
    print("Test 2 Failed")
```

ตัวอย่างที่ **4:** การทดสอบฟังก์ชันแปลงข้อความ **(String Manipulation)**

```
#ตัวอย่างที่ 4: การทดสอบฟังก์ชันแปลงข้อความ (String Manipulation)

def capitalize_word(word):
    return word.capitalize()

def reverse_string(string):
    return string[::-1]

print("\nString Manipulation Tests:")
# ทดสอบฟังก์ชัน capitalize_word
try:
    assert capitalize_word("hello") == "Hello"
    print("Test 1 Passed: Capitalized 'hello'")
except AssertionError:
    print("Test 1 Failed")
```

AJ.Tatar(SE)

```
try:
    assert capitalize_word("python") == "Python"
    print("Test 2 Passed: Capitalized 'python'")
except AssertionError:
    print("Test 2 Failed")

# ทดสอบฟังก์ชัน reverse_string
try:
    assert reverse_string("abcd") == "dcba"
    print("Test 3 Passed: Reversed 'abcd' to 'dcba'")
except AssertionError:
    print("Test 3 Failed")

try:
    assert reverse_string("12345") == "54321"
    print("Test 4 Passed: Reversed '12345' to '54321'")
except AssertionError:
    print("Test 4 Failed")
```

ตัวอย่างที่ 5: การทดสอบฟังก์ชันการจัดการวันที่ (Date and Time)

```
# ตัวอย่างที่ 5: การทดสอบฟังก์ชันการจัดการวันที่ (Date and Time)
from datetime import datetime

def get_day_of_week(date_string):
    # Convert string to date
    date_object = datetime.strptime(date_string, '%Y-%m-%d')
    return date_object.strftime('%A')

print("\nDate and Time Tests:")
# ทดสอบฟังก์ชัน get_day_of_week
try:
    assert get_day_of_week("2023-01-01") == "Sunday"
    print("Test 1 Passed: 2023-01-01 is Sunday")
except AssertionError:
    print("Test 1 Failed")

try:
    assert get_day_of_week("2025-01-03") == "Friday"
    print("Test 2 Passed: 2025-01-03 is Friday")
except AssertionError:
    print("Test 2 Failed")
```

AI-Tatar(SE)

ตัวอย่างที่ **6:** การทดสอบฟังก์ชันการตรวจสอบข้อมูล **(Validation Functions)**

```
#ตัวอย่างที่ 6: การทดสอบฟังก์ชันการตรวจสอบข้อมูล (Validation Functions)


def is_email_valid(email):
    return "@" in email and "." in email.split("@")[-1]

print("\nValidation Tests:")
# ทดสอบฟังก์ชัน is_email_valid
try:
    assert is_email_valid("test@example.com") == True
    print("Test 1 Passed: Valid email 'test@example.com'")
except AssertionError:
    print("Test 1 Failed")

try:
    assert is_email_valid("invalid-email") == False
    print("Test 2 Passed: Invalid email 'invalid-email'")
except AssertionError:
    print("Test 2 Failed")
```

ตัวอย่างที่ **7:** การทดสอบฟังก์ชันสำหรับคำศัพท์ **(Word Operations)**

```
#ตัวอย่างที่ 7: การทดสอบฟังก์ชันสำหรับคำศัพท์ (Word Operations)

def word_count(sentence):
    return len(sentence.split())

def starts_with_vowel(word):
    return word[0].lower() in 'aeiou'

print("\nWord Operations Tests:")
# ทดสอบฟังก์ชัน word_count
try:
    assert word_count("Hello world!") == 2
    print("Test 1 Passed: 'Hello world!' has 2 words")
except AssertionError:
    print("Test 1 Failed")

# ทดสอบฟังก์ชัน starts_with_vowel
try:
    assert starts_with_vowel("Apple") == True
    print("Test 2 Passed: 'Apple' starts with a vowel")
except AssertionError:
    print("Test 2 Failed")
```

Aj.Tatar(SE)

```
try:
    assert starts_with_vowel("Banana") == False
    print("Test 3 Passed: 'Banana' does not start with a vowel")
except AssertionError:
    print("Test 3 Failed")
```

ตัวอย่างที่ 8: การทดสอบฟังก์ชันการจัดการดัชนีในรายการ (List Indexing)

```
#ตัวอย่างที่ 8: การทดสอบฟังก์ชันการจัดการดัชนีในรายการ (List Indexing)

def get_item_at_index(lst, index):
    if index < 0 or index >= len(lst):
        raise IndexError("Index out of range")
    return lst[index]

print("\nList Indexing Tests:")
# ทดสอบฟังก์ชัน get_item_at_index
my_list = [10, 20, 30, 40]
try:
    assert get_item_at_index(my_list, 2) == 30
    print("Test 1 Passed: Item at index 2 is 30")
except AssertionError:
    print("Test 1 Failed")

try:
    get_item_at_index(my_list, 5)
except IndexError as e:
    assert str(e) == "Index out of range"
    print("Test 2 Passed: Index out of range exception raised")
```

Aj. Tatar(SE)

```
#ตัวอย่างที่ 9: การทดสอบฟังก์ชันค้นหาสูงสุดและต่ำสุด (Max/Min Operations)


def find_max(lst):
    if not lst:
        raise ValueError("List is empty")
    return max(lst)

def find_min(lst):
    if not lst:
        raise ValueError("List is empty")
    return min(lst)

print("\nMax/Min Operations Tests:")
# ทดสอบฟังก์ชัน find_max
numbers = [1, 2, 3, 4, 5]
try:
    assert find_max(numbers) == 5
    print("Test 1 Passed: Max of [1, 2, 3, 4, 5] is 5")
except AssertionError:
    print("Test 1 Failed")

# ทดสอบฟังก์ชัน find_min
try:
    assert find_min(numbers) == 1
    print("Test 2 Passed: Min of [1, 2, 3, 4, 5] is 1")
except AssertionError:
    print("Test 2 Failed")
```

ตัวอย่างที่ 10: การทดสอบฟังก์ชันเปรียบเทียบค่า (Comparison Functions)

```
#ตัวอย่างที่ 10: การทดสอบฟังก์ชันเปรียบเทียบค่า (Comparison Functions)

def is_greater(a, b):
    return a > b

def is_equal(a, b):
    return a == b

print("\nComparison Functions Tests:")
# ทดสอบฟังก์ชัน is_greater
try:
    assert is_greater(10, 5) == True
    print("Test 1 Passed: 10 is greater than 5")
except AssertionError:
```

Aj.Tatar(SE)

```
    print("Test 1 Failed")

# ทดสอบฟังก์ชัน is_equal
try:
    assert is_equal(10, 10) == True
    print("Test 2 Passed: 10 is equal to 10")
except AssertionError:
    print("Test 2 Failed")
```

ตัวอย่างที่ 11: การทดสอบฟังก์ชันค้นหาข้อความ (Substring Search)

```
#ตัวอย่างที่ 11: การทดสอบฟังก์ชันค้นหาข้อความ (Substring Search)
def contains_substring(string, substring):
    return substring in string

print("\nSubstring Search Tests:")
try:
    assert contains_substring("Hello, world!", "world") == True
    print("Test 1 Passed: 'world' found in 'Hello, world!'")
except AssertionError:
    print("Test 1 Failed")

try:
    assert contains_substring("Hello, world!", "Python") == False
    print("Test 2 Passed: 'Python' not found in 'Hello, world!'")
except AssertionError:
    print("Test 2 Failed")
```

Aj.Tatar(SE)

ตัวอย่างที่ **12:** การทดสอบฟังก์ชันการจัดการไฟล์ **(File Operations)**

```
#ตัวอย่างที่ 12: การทดสอบฟังก์ชันการจัดการไฟล์ (File Operations)
def read_file(file_path):
    with open(file_path, 'r') as file:
        return file.read()

def write_file(file_path, content):
    with open(file_path, 'w') as file:
        file.write(content)

print("\nFile Operations Tests:")
test_file = "test_file.txt"
try:
    write_file(test_file, "Hello, Automated Testing!")
    content = read_file(test_file)
    assert content == "Hello, Automated Testing!"
    print("Test 1 Passed: File read and write successful")
except AssertionError:
    print("Test 1 Failed")
finally:
    import os
    if os.path.exists(test_file):
        os.remove(test_file)
```

ตัวอย่างที่ **13:** การทดสอบฟังก์ชันแปลงหน่วย **(Unit Conversion)**

```
#ตัวอย่างที่ 13: การทดสอบฟังก์ชันแปลงหน่วย (Unit Conversion)

def celsius_to_fahrenheit(celsius):
    return (celsius * 9/5) + 32

def fahrenheit_to_celsius(fahrenheit):
    return (fahrenheit - 32) * 5/9

print("\nUnit Conversion Tests:")
try:
    assert celsius_to_fahrenheit(0) == 32
    print("Test 1 Passed: 0°C = 32°F")
except AssertionError:
    print("Test 1 Failed")

try:
    assert fahrenheit_to_celsius(32) == 0
    print("Test 2 Passed: 32°F = 0°C")
except AssertionError:
    print("Test 2 Failed")
```

Ai Tatar(SE)

ตัวอย่างที่ **14:** การทดสอบฟังก์ชันจัดการตัวเลข **(Number Operations)**

```
#ตัวอย่างที่ 14: การทดสอบฟังก์ชันจัดการตัวเลข (Number Operations)

def is_even(number):
    return number % 2 == 0

def is_odd(number):
    return number % 2 != 0

print("\nNumber Operations Tests:")
try:
    assert is_even(4) == True
    print("Test 1 Passed: 4 is even")
except AssertionError:
    print("Test 1 Failed")

try:
    assert is_odd(7) == True
    print("Test 2 Passed: 7 is odd")
except AssertionError:
    print("Test 2 Failed")
```

ตัวอย่างที่ **15:** การทดสอบฟังก์ชันแยกคำ **(Splitting Strings)**

```
#ตัวอย่างที่ 15: การทดสอบฟังก์ชันแยกคำ (Splitting Strings)
def split_words(sentence):
    return sentence.split()

print("\nSplitting Strings Tests:")
try:
    assert split_words("Hello world!") == ["Hello", "world!"]
    print("Test 1 Passed: 'Hello world!' split correctly")
except AssertionError:
    print("Test 1 Failed")
```

AJ Tatar(SE)

ตัวอย่างที่ **16:** การทดสอบฟังก์ชันคำนวณทางการเงิน **(Financial Calculations)**

```
#ตัวอย่างที่ 16: การทดสอบฟังก์ชันคำนวณทางการเงิน (Financial Calculations)

def calculate_interest(principal, rate, time):
    return (principal * rate * time) / 100

print("\nFinancial Calculations Tests:")
try:
    assert calculate_interest(1000, 5, 2) == 100
    print("Test 1 Passed: Interest for 1000 at 5% for 2 years is 100")
except AssertionError:
    print("Test 1 Failed")
```

ตัวอย่างที่ **17:** การทดสอบฟังก์ชันสร้างรหัสผ่าน **(Password Generation)**

```
#ตัวอย่างที่ 17: การทดสอบฟังก์ชันสร้างรหัสผ่าน (Password Generation)
import random
import string

def generate_password(length):
    if length < 6:
        raise ValueError("Password must be at least 6 characters")
    return ''.join(random.choices(string.ascii_letters + string.digits, k=length))

print("\nPassword Generation Tests:")
try:
    password = generate_password(8)
    assert len(password) == 8
    print(f"Test 1 Passed: Password '{password}' generated with length 8")
except AssertionError:
    print("Test 1 Failed")

try:
    generate_password(4)
except ValueError as e:
    assert str(e) == "Password must be at least 6 characters"
    print("Test 2 Passed: Error for short password length")
```

A.Tatar(SE)

ตัวอย่างที่ **18:** การทดสอบฟังก์ชันนับตัวอักษรในข้อความ **(Character Count)**

```
#ตัวอย่างที่ 18: การทดสอบฟังก์ชันนับตัวอักษรในข้อความ (Character Count)

def count_characters(string):
    return len(string)

print("\nCharacter Count Tests:")
try:
    assert count_characters("Hello") == 5
    print("Test 1 Passed: 'Hello' has 5 characters")
except AssertionError:
    print("Test 1 Failed")

try:
    assert count_characters("") == 0
    print("Test 2 Passed: Empty string has 0 characters")
except AssertionError:
    print("Test 2 Failed")
```

ตัวอย่างที่ **19:** การทดสอบฟังก์ชันจัดเรียงลำดับ **(Sorting)**

```
#ตัวอย่างที่ 19: การทดสอบฟังก์ชันจัดเรียงลำดับ (Sorting)
def sort_list(lst):
    return sorted(lst)

print("\nSorting Tests:")
try:
    assert sort_list([3, 1, 2]) == [1, 2, 3]
    print("Test 1 Passed: [3, 1, 2] sorted to [1, 2, 3]")
except AssertionError:
    print("Test 1 Failed")

try:
    assert sort_list([]) == []
    print("Test 2 Passed: Empty list remains empty")
except AssertionError:
    print("Test 2 Failed")
```

Ai.Tatar(SE)

ตัวอย่างที่ **20:** การทดสอบฟังก์ชันตรวจสอบคำ **Palindrome (Palindrome Check)**

```
# ตัวอย่างที่ 20: การทดสอบฟังก์ชันตรวจสอบคำ Palindrome (Palindrome Check)
def is_palindrome(word):
    # เปรียบเทียบคำกับคำที่กลับด้าน
    return word == word[::-1]

print("\nPalindrome Check Tests:")
# ทดสอบฟังก์ชัน is_palindrome
try:
    assert is_palindrome("madam") == True
    print("Test 1 Passed: 'madam' is a palindrome")
except AssertionError:
    print("Test 1 Failed")

try:
    assert is_palindrome("hello") == False
    print("Test 2 Passed: 'hello' is not a palindrome")
except AssertionError:
    print("Test 2 Failed")
```

Aj.Tatar(SE)