Unit testing of the task 3.2

**Test display\_csv\_as\_table Function**

**Test Invalid Input: Integer as File Name**

**Test Description**: This test case ensures that the function throws a TypeError when the file name provided is an integer.

**Test Input**: csv\_file\_name=123

**Expected Result**: A TypeError with the message: "File name must be a string".

**Actual Result**:

The function raised an OSError: [WinError 6] invalid handle, indicating an issue when trying to open a file handle with an integer as the file name.

The regex matching for the error message "File name must be a string" did not match the expected string (expected str, bytes or os.PathLike object, not int).

**Status**: **Failed**

**Error**: The error raised by open() was an OSError instead of the expected TypeError.

**Test Invalid Input: Float as File Name**

**Test Description**: This test case ensures that the function throws a TypeError when the file name provided is a float.

**Test Input**: csv\_file\_name=45.67

**Expected Result**: A TypeError with the message: "File name must be a string".

**Actual Result**:

The function raised a TypeError with the message: "expected str, bytes or os.PathLike object, not float".

However, the regex pattern "File name must be a string" did not match the actual exception message, which caused the test to fail.

**Status**: **Failed**

**Error**: The error message did not match the expected string.

**Test Invalid Input: List as File Name**

**Test Description**: This test case ensures that the function throws a TypeError when the file name provided is a list.

**Test Input**: csv\_file\_name=["file1.csv", "file2.csv"]

**Expected Result**: A TypeError with the message: "File name must be a string".

**Actual Result**:

The function raised a TypeError with the message: "expected str, bytes or os.PathLike object, not list".

The regex pattern "File name must be a string" did not match the actual exception message.

**Status**: **Failed**

**Error**: The error message did not match the expected string.

**Test Invalid Input: None as File Name**

**Test Description**: This test case ensures that the function throws a TypeError when the file name provided is None.

**Test Input**: csv\_file\_name=None

**Expected Result**: A TypeError with the message: "File name must be a string".

**Actual Result**:

The function raised a TypeError with the message: "expected str, bytes or os.PathLike object, not NoneType".

The regex pattern "File name must be a string" did not match the actual exception message.

**Status**: **Failed**

**Error**: The error message did not match the expected string.

Additional Test Results:

**Test Case: File Not Found**

**Test Description**: Test for the case when the file does not exist.

**Test Result**: Passed as expected with a FileNotFoundError.

**Test Case: Empty File**

**Test Description**: Test for the case when the file is empty.

**Test Result**: Passed. The function correctly raised a StopIteration exception due to no data being present in the file.

**Test Case: Invalid CSV File**

**Test Description**: Test for an invalid file path, ensuring an OSError is raised.

**Test Result**: Passed. An OSError was raised as expected.

**Test Case: Header Only**

**Test Description**: Test when the file contains only the header row.

**Test Result**: Passed. The function correctly displayed the header as a table.

**Test Case: Single Data Row**

**Test Description**: Test when the file contains a single data row.

**Test Result**: Passed. The function correctly displayed both the header and data row.

**Test Case: Multiple Data Rows**

**Test Description**: Test when the file contains multiple data rows.

**Test Result**: Passed. The function correctly displayed multiple rows in the table format.

**Test Case: Mixed Data Types**

**Test Description**: Test when the file contains various data types (strings and numbers).

**Test Result**: Passed. The function handled mixed data types correctly.

**Test Case: Special Characters in Product Name**

**Test Description**: Test when the product name contains special characters (e.g., commas).

**Test Result**: Passed. The function displayed the product name with special characters properly.

**Test Case: Product Name with Spaces**

**Test Description**: Test when the product name contains spaces.

**Test Result**: Passed. The function correctly handled spaces in product names.

**Test Case: Integer Product Price**

**Test Description**: Test when the product price is an integer.

**Test Result**: Passed. The function displayed the integer price correctly.

**Test Case: Floating Point Product Price**

**Test Description**: Test when the product price is a floating-point number.

**Test Result**: Passed. The function displayed the floating-point price correctly.

**Test Case: Large Numbers**

**Test Description**: Test when the file contains large numbers.

**Test Result**: Passed. The function handled large numbers correctly.

**Test Case: Empty Product Name**

**Test Description**: Test when the product name is empty.

**Test Result**: Passed. The function displayed the empty product name correctly.

**Test display\_filtered\_table Function**

test\_valid\_csv\_file\_name\_string:

Purpose: Ensure the function correctly handles valid CSV data and displays all products.

Expected Outcome: The function should display the header followed by each product's details.

Result: Passed.

test\_display\_filtered\_table\_with\_valid\_search:

Purpose: Verify that the function filters results correctly based on a valid search term.

Expected Outcome: Only the products matching the search term should be displayed.

Result: Passed.

test\_display\_filtered\_table\_with\_empty\_search:

Purpose: Test the function's behavior when an empty search term is provided.

Expected Outcome: Only the header should be displayed as no products match the empty search.

Result: Passed.

test\_display\_filtered\_table\_single\_product:

Purpose: Ensure that when only one product is present in the data, it is displayed correctly.

Expected Outcome: The single product should be shown along with the header.

Result: Passed.

test\_display\_filtered\_table\_same\_product\_name:

Purpose: Test how the function handles multiple entries of the same product name with different prices or quantities.

Expected Outcome: All instances of the same product should be displayed.

Result: Passed.

test\_display\_filtered\_table\_special\_characters:

Purpose: Verify that the function handles products with special characters in their names.

Expected Outcome: Special characters should be correctly displayed along with the product data.

Result: Passed.

test\_display\_filtered\_table\_no\_search\_target:

Purpose: Check the behavior when no search target is specified (i.e., all products should be displayed).

Expected Outcome: The header followed by all product rows should be shown.

Result: Passed.

test\_display\_filtered\_table\_with\_search\_target:

Purpose: Ensure that the function displays the correct products when a search term is provided.

Expected Outcome: Only the products that match the search term should be displayed.

Result: Passed.

test\_display\_filtered\_table\_no\_match:

Purpose: Verify the function's behavior when no products match the search term.

Expected Outcome: Only the header should be displayed if no products match.

Result: Passed.

test\_display\_filtered\_table\_case\_insensitive:

Purpose: Ensure case-insensitive matching for search terms.

Expected Outcome: The function should find and display products regardless of case (e.g., "apple" should match "Apple").

Result: Passed.

test\_invalid\_csv\_file\_name\_int:

Purpose: Test how the function handles invalid CSV file name types (integer).

Expected Outcome: The function should raise an OSError when the file name is an integer.

Result: Passed.

test\_invalid\_csv\_file\_name\_float:

Purpose: Check how the function reacts to a float being passed as a file name.

Expected Outcome: The function should raise a TypeError.

Result: Passed.

test\_invalid\_csv\_file\_name\_list:

Purpose: Ensure the function raises an error when a list is provided as the file name.

Expected Outcome: The function should raise a TypeError.

Result: Passed.

test\_invalid\_csv\_file\_name\_string:

Purpose: Test how the function handles an invalid file path (non-existent file).

Expected Outcome: The function should raise a FileNotFoundError.

Result: Passed.

**Test search\_and\_purchase\_product() Function**

Function Purpose:  
The function facilitates the product search and purchase process in an online shopping system. It allows users to search products based on different criteria, view product details, and proceed with the purchase.

Core Features:

User login: Simulates user authentication before initiating the search and purchase process.

Product search: Users can search for products by:

Exact product name

Fuzzy name pattern

Price range

Stock range

Initial letter of the product name

Exact price

Purchase process: After the search, the user can confirm the purchase of selected items.

Mocked Dependencies:

UserInterface.get\_user\_input: Mocks user input for search criteria (e.g., name, price, stock) and purchase confirmation.

product\_data.get\_csv\_data: Mocks retrieval of product data in CSV format.

product\_search.display\_csv\_as\_table: Mocks the display of the product data as a table.

product\_data.get\_products: Mocks retrieving the list of available products.

user\_login.login: Mocks the user login process.

user\_data.UserDataManager.load\_users: Mocks loading of user data.

checkout\_process.checkout\_and\_payment: Mocks the checkout and payment operations.

Test Cases:

Query all products: Ensure the function displays all products correctly.

Search by exact product name: Verify that the system handles exact name queries and shows relevant results.

Search by stock range: Check the system's ability to filter products based on user-defined stock ranges.

Search by price range: Test that the system correctly filters products based on a specified price range.

Fuzzy search by name: Ensure that the system supports fuzzy searches (e.g., searching for "Laptop" matches "laptop").

Search by name initial: Test search functionality when querying by the first letter of the product name.

Search by exact price: Confirm that the system can filter products by an exact price.

Handle invalid price input: Validate that the system gracefully handles invalid price inputs, prompting for correction if necessary.

**Some errors and the last function did not complete all test cases, since recheck with teacher the right output we want to get is late so they will be finished after submission.**