Fall 2025 - CS 122
Homework 2

Due: Sep. 16th , 11:59 PM
Topics: Functions, Decorators, Generators, and Modules
Total Points: 100

Overview

This assignment is designed to strengthen your understanding of key Python concepts including functions, lambda functions, decorators, generators, and proper code organization using modules. You will build two small applications: a library management system and an e-commerce price calculator.

Project Setup and Submission

1. File Structure:
You must create the following directory structure, replacing SJSUID with your own 9-digit San Jose State University ID number.

```
hw2_SJSUID/
├── question_1/
│   ├── __init__.py  (can be empty)
│   ├── library.py
│   └── utils.py
├── question_2/
│   ├── __init__.py  (can be empty)
│   └── pricing.py
└── test_homework.py
```

2. Submission:
Compress the entire hw2_SJSUID directory into a single file named hw2_SJSUID.zip and submit it.

3. Testing Your Work:
A file named test_homework.py is provided to you along with this homework document. Place it in the root of your hw2_SJSUID directory as shown above. This script contains automated tests that will help you check if your code is working correctly.

How to Run the Tests:
You do not need to understand the code inside test_homework.py (We will be covering this in future sessions). To run it, open your terminal, navigate into your hw2_SJSUID directory, and run the following command:

“python test_homework.py”

You will see output indicating which tests passed (OK) and which failed (FAIL). Your goal is to make all tests pass.

Grading Rubric

- **Automated Tests (60 points):** Your submission must pass all the tests in the provided test_homework.py file.
- **Instructor Tests (20 points):** Your code will be tested against a second set of tests (separate version of test file, not shared with students) with different inputs to check for robustness.
- **Code Quality (20 points):** Your code must be clean, well-commented, and easy to read.

Question 1: Library Management System (50 points)

Objective: Create a set of modules to manage a collection of books.

Tasks:

1. **library.py Module:**
   - Create a function add_book(library, title, author) that adds a new book dictionary {'title': title, 'author': author} to the library list.
   - Create a function remove_book(library, title) that removes a book by its title. If the book is not found, it should print "Error: Book '{title}' not found.".
   - Create a function list_books(library) that prints all books. If the library is empty, it should print "The library is empty.".
   - **Important:** You must apply the @log_operation decorator (from utils.py) to both the add_book and remove_book functions.
2. **utils.py Module:**
   - Create a decorator log_operation(func) that prints a log message Executing {function_name}... before the decorated function is executed.
   - Create a variable filter_by_author assigned to a lambda function. It should take a library and an author and return a new list of books by that author.
   - Create a generator function book_generator(library) that yields one book at a time from the library.

Question 2: E-commerce Price Calculator (50 points)

Objective: Build a system to process product prices, applying conditional discounts and taxes.

Tasks:

1. **pricing.py Module:**
   - Create a variable apply_tax assigned to a lambda function. It should take a price and return the price after applying a fixed **8% tax**.

- Create a decorator apply_discount(func) that applies a **10% discount** to the price argument *before* the wrapped function is called, but **only if the original price is greater than $100.00**.
- Create a generator function price_generator(prices, processing_function) that takes a list of prices and a function, applies the function to each price, and yields the result.