Low-Level Design (LLD) for Product Price List Manipulation and Operations

Difficulty Level: Medium | Total Marks: 10
Standards Followed: 3 Functions | 3 Visible Test Cases

Concepts Tested

- Python list manipulation
- Mathematical operations and percentage calculations
- Conditional categorization
- Aggregation functions (sum, len)
- Looping and list transformation
- Basic data processing logic

==

Problem Statement

Design a system that manages a list of product prices and performs operations such as:

- Calculating the total revenue if all products are sold
- Applying a 10% discount to all prices and generating a new list
- Categorizing products into affordable and premium based on price

Given Input:

prices = [250, 1500, 350, 2000, 800, 1200, 450, 3000]

Category Rules:

- Affordable: price < 1000
- Premium: price >= 1000

==

Operations

1. Calculate Total Revenue

Function Prototype:
def calculate_total_revenue(self):

Expected Output:
Total Revenue: 9550

----

2. Apply Discount and Generate Discounted Price List

Function Prototype:
def apply_discount(self):

Expected Output:
Discounted Prices: [225, 1350, 315, 1800, 720, 1080, 405, 2700]

----

3. Count Products by Category

Function Prototype:

```
def count_categories(self):
```

Expected Output:

```
Affordable Count: 4, Premium Count: 4
```

==

Implementation Code

```
class ProductManager:

  def __init__(self):
    """Initialize product price list."""
    self.prices = [250, 1500, 350, 2000, 800, 1200, 450, 3000]

  def calculate_total_revenue(self):
    """Calculate and print total revenue."""
    “””Your code here”””

  def apply_discount(self):
    """Apply 10% discount and print discounted prices."""
    discounted = []
  def count_categories(self):
    """Count affordable and premium products."""
    affordable = 0
    premium = 0
    “””Your code here”””
    print(f"Affordable Count: {affordable}, Premium Count: {premium}")
```

===

Test Case Table

Test Case ID	**Test Case Description**	**Associated Function(s)**	**Marks**
TC1	Calculate total revenue	calculate_total_revenue()	3 Marks
TC2	Apply discount to all products	apply_discount()	3 Marks
TC3	Count affordable and premium products	count_categories()	4 Marks
TOTAL	**All test cases passed**	-	**10 Marks**

===

Visible Test Cases

TC1 Input:

1

total

Output:

```
Total Revenue: 9550
```

TC2 Input:

```
1
discount
```

Output:

```
Discounted Prices: [225, 1350, 315, 1800, 720, 1080, 405, 2700]
```

TC3 Input:

```
1
count
```

Output:

```
Affordable Count: 4, Premium Count: 4
```