AI LAB
ASSIGNMENT 3

by

AMMAR JAMIL (01-134231-010)

Submitted to: DR ARSHAD FARHAD
Date: 2/16/2025

DEPARTMENT OF COMPUTER SCIENCE

BAHRIA UNIVERSITY ISLAMABAD E-8

Task 1: Student Grade Management System

Create a program that manages student grades with validation and analysis.

- **Ask the user to enter 5 grades (0-100) with validation**
- **Calculate and display the average grade**
- **Find and display the highest and lowest grades**
- **Count how many grades are above a user-specified threshold**

Hint: Use list comprehension for input collection, built-in functions like max(), min(), and sum(), and implement input validation with while loops and try-except blocks.

SOL

```
try:
    grades = [int(input("Enter grade (0-100): ")) for i in range(5)]

    if any(grade < 0 or grade > 100 for grade in grades):
        raise ValueError("Please Enter valid grades.")

    average = sum([grade for grade in grades]) / len(grades)
    highest = max([grade for grade in grades])
    lowest = min([grade for grade in grades])

    threshold = int(input("Enter threshold (0-100): "))
    if threshold < 0 or threshold > 100:
        print("Please Enter a valid threshold.")
    else:
        above_threshold = len([grade for grade in grades if grade > threshold])

        print("Average grade:", round(average, 2))
        print("Highest grade:", highest)
        print("Lowest grade:", lowest)
```

```
        print("Grades above threshold:", above_threshold)

except ValueError as err:
    print(err)
```

```
C:\Users\Ammar Jamil\Desktop\python>"C:/Users/Ammar Jamil/AppData/Local/Programs/Python/Python313/pytho
Enter grade (0-100): 78
Enter grade (0-100): 67
Enter grade (0-100): 88
Enter grade (0-100): 65
Enter grade (0-100): 67
Enter threshold (0-100): 79
Average grade: 73.0
Highest grade: 88
Lowest grade: 65
Grades above threshold: 1

C:\Users\Ammar Jamil\Desktop\python>
```

Task 2: Shopping Cart with Inventory Validation

Build a shopping cart system with product inventory and validation.

- **Predefine a dictionary of products with prices and stock quantities**
- **Allow users to add/remove items with quantity validation**
- **Calculate total cost with tax**
- **Apply and validate discount codes**

Hint: Use dictionary for product inventory, implement input validation for quantities, use dictionary methods like get() and update(), and create functions for cart operations.

SOL

```
product_inventory = {
    "apple": {"price": 50, "stock": 10},
    "banana": {"price": 30, "stock": 15},
    "milk": {"price": 200, "stock": 5}
}
```

```
shopping_cart = {}

tax_rate = 0.1
valid_discount_codes = {"10": 0.1, "20": 0.2}

def add_item(product_inventory, shopping_cart):
    selected_product = input("Enter product name: ")
    quantity = int(input("Enter quantity: "))

    if (
        selected_product in product_inventory
        and 0 < quantity <= product_inventory[selected_product]["stock"]
    ):
        shopping_cart[selected_product] = (
            shopping_cart.get(selected_product, 0) + quantity
        )
        product_inventory[selected_product]["stock"] -= quantity
        print("Added " + str(quantity) + " " + selected_product + "(s) to your cart.")
    else:
        print("Invalid product or quantity!")

def remove_item(product_inventory, shopping_cart):
    selected_product = input("Enter product name: ")
    quantity = int(input("Enter quantity: "))

    if (
        selected_product in shopping_cart
        and 0 < quantity <= shopping_cart[selected_product]
    ):
        shopping_cart[selected_product] -= quantity
```

```
            product_inventory[selected_product]["stock"] += quantity

            if shopping_cart[selected_product] == 0:
                del shopping_cart[selected_product]

            print("Removed " + str(quantity) + " " + selected_product + "(s) from your cart.")
        else:
            print("Invalid product or quantity!")

    def checkout(product_inventory, shopping_cart, tax_rate, valid_discount_codes):
        if not shopping_cart:
            print("Your cart is empty!")
            return False

        subtotal = sum(
            product_inventory[item]["price"] * qty
            for item, qty in shopping_cart.items()
        )

        discount_code = input("Enter discount code (press Enter to skip): ")
        discount_amount = 0

        if discount_code in valid_discount_codes:
            discount_amount = subtotal * valid_discount_codes[discount_code]
            subtotal -= discount_amount
            print("Discount applied! You saved Rs." + str(round(discount_amount, 2)))
        elif discount_code != "":
            print("Invalid discount code! No discount applied.")

        tax_amount = subtotal * tax_rate
```

```
    total_with_tax = subtotal + tax_amount

    print("Total : Rs." + str(round(subtotal + discount_amount, 2)))
    if discount_amount > 0:
        print("Discount: -Rs." + str(round(discount_amount, 2)))
    print("Tax (10%): Rs." + str(round(tax_amount, 2)))
    print("Total to Pay: Rs." + str(round(total_with_tax, 2)))

    return True


while True:
    available_products = {
        product: str(info['price']) + " (stock: " + str(info['stock']) + ")"
        for product, info in product_inventory.items()
    }
    print("\nAvailable Products:", available_products)
    print("Your Cart:", shopping_cart if shopping_cart else "Empty")

    action = input("\nPress :a for add, r for remove , c for checkout and q to quit the program: ").lower()

    if action == "a":
        add_item(product_inventory, shopping_cart)

    elif action == "r":
        remove_item(product_inventory, shopping_cart)

    elif action == "c":
        if checkout(product_inventory, shopping_cart, tax_rate, valid_discount_codes):
            break
```

```
elif action == "q":
    print("Thank you for shopping with us!")
    break

else:
    print("Invalid choice, please try again.")
```