**Assignment Report: Inventory Management System**

Arshpreet Singh Sidhu (Student ID: 815805)

**Deadline: 11:59PM on 1 March 2024**

**1. Introduction:**

This assignment is about the development of a Python inventory management program. The program offers functionalities such as adding items to the inventory, removing items from the inventory, and displaying the current inventory through a menu-based interface.

**2. Problem Statement:**

You are tasked with creating a Python program to manage a small inventory for a store. Your program should allow users to perform various actions such as adding items to the inventory, removing items from the inventory, and displaying the current inventory. Error handling for invalid inputs and edge cases is also required.

**3. Methodology:**

- **User Interaction Design**: The program features a user-friendly interface with clear prompts and a menu-based system, ensuring easy navigation and minimizing user errors. This design prioritizes efficiency and enhances the overall user experience in managing the inventory.



- **Modularization**: The code is modularized using functions to handle each functionality separately, ensuring a clear and organized structure.



- **Error Handling**: Appropriate error-handling techniques are implemented to gracefully manage invalid inputs and edge cases, ensuring the robustness of the program. For instance, In function for adding items to inventory, conditional statements are used to verify the range of quantity inputted by the user, while try-except blocks are implemented to handle cases where the user enters incorrect input types such as strings or floats for value of “quantity”.



- **Pattern Matching with “*match case*”**: The match case statement is used to handle different cases based on user input, instead of traditional if-else statements. The use of match case increases the clarity and consistency of the code, making it easier to understand and maintain. It also contributes the Pythonic principle of readability and simplicity. For instance, when the user selects an option from the menu, instead of using multiple if-else statements to check each possible choice, a match case statement can be used to handle each option more efficiently.



**4. Implementation:**

The implementation involves the use of functions to add items, remove items, display the inventory, and handle user inputs. Error handling mechanisms are integrated to manage exceptions and ensure smooth execution of the program.

**5. Results:**

The program manages the inventory effectively, allowing users to add, remove, and view items in the inventory. Error handling enhances the stability of the program even when user enters unexpected inputs.

**6. Screenshots:**

**Figure 1: Screenshot of the source code of the program.
**

Figure 1: Screenshot of the source code of the program.

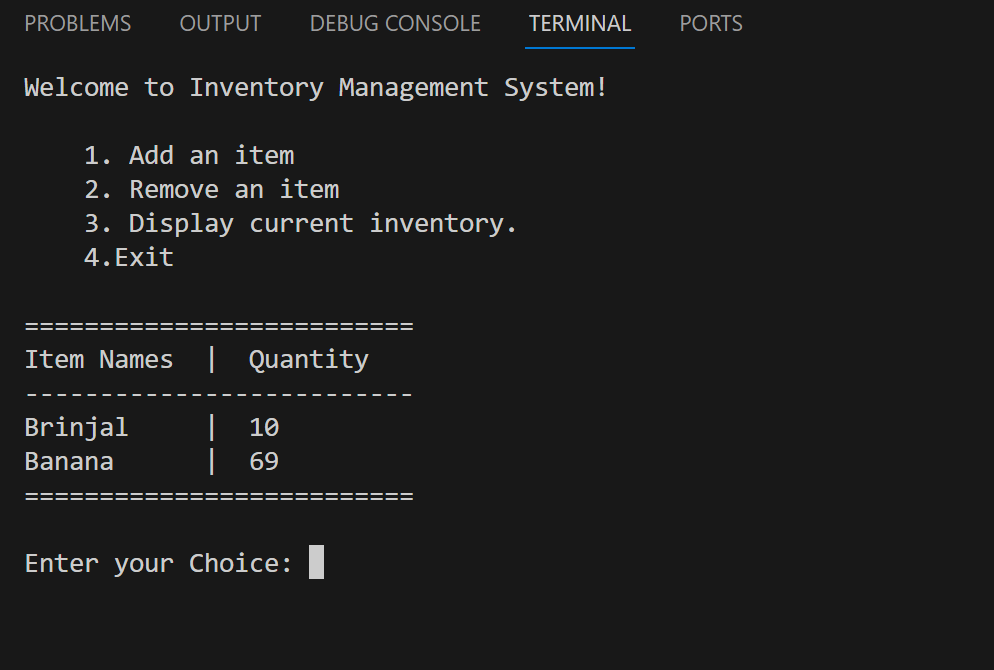


Figure 2: Screenshot of the execution of the program.

**7. Efficiency Approaches:**

- **Use of While Loops**: While loops are used to create a menu-based interface that allows users to interact with the inventory system repeatedly until they choose to exit. This ensures a continuous user experience without the need to restart the program after each operation.

- **Match Case for User Input**: The `match case` statement is employed to handle user input effectively. It provides a concise and readable way to match different user choices and execute corresponding actions. By using `match case`, the code becomes more structured and easier to maintain compared to traditional `if-else` statements.

- **Case Insensitivity**: Case insensitivity is used to make the program more user-friendly. User input for item names is converted to lowercase before saving to make sure that the program can process all type in casing, increasing the usability of the program.

- **String Formatting Operations for Display**: String formatting operations are used to present the inventory in a well-organized and readable format. The `%-12s` format specifier is used to align item names and quantities neatly in columns, increasing the visual presentation of the inventory.

- **Use of clear\_screen()**: The `clear\_screen()` function is utilized to clear the terminal screen before displaying the menu or executing any operation. It uses command according to platform of user to clear terminal’s screen. This ensures a clean and clutter-free interface, preventing previous outputs from cluttering the display and providing a more pleasant user experience.

**8. References:**

Online resources during the development of this assignment are: -

1. Official Python Documentation
2. W3Schools
3. Lecture Slides, provided by *Jeeho Ryoo*