Inventory Management Explanation

Block 1: Main Function and Initialization

```
def main():
    # Initialize inventory and categories dictionaries
    inventory = {}
    categories = {}

    print("Welcome to the Inventory Management System!")
    while True:
        # Display main menu and get user choice
        print_menu()
        choice = get_integer_input("Enter choice: ")

        # Perform actions based on user choice
        if choice == 1:
            add_inventory(inventory, categories)
        elif choice == 2:
            remove_inventory(inventory)
        elif choice == 3:
            update_inventory(inventory)
        elif choice == 4:
            search_inventory(inventory)
        elif choice == 5:
            print_inventory(inventory)
        elif choice == 6:
```

```
            manage_categories(inventory, categories)
        elif choice == 99:
            print("Goodbye! Thank you for using the Inventory Management System.")
            break
        else:
            print("Invalid choice. Please try again.")
```

Explanation:

- The `main` function serves as the entry point for the program.

- It initializes two dictionaries, `inventory` and `categories`, to store information about items and categories, respectively.

- The program enters a loop that displays the main menu and prompts the user for a choice.

- Based on the user's choice, various functions are called to perform actions such as adding, removing, updating, or searching for items, managing categories, printing the inventory, or quitting the program.

Block 2: Print Menu Function

```
# Function to print the main menu
def print_menu():
    print('\n============================')
    print('= Inventory Management Menu =')
    print('============================')
    print('(1) Add New Item to Inventory')
    print('(2) Remove Item from Inventory')
    print('(3) Update Inventory')
    print('(4) Search Item in Inventory')
```

```
    print('(5) Print Inventory Report')
    print('(6) Manage Categories')
    print('(99) Quit')
```

Explanation:

- The `print_menu` function prints the options available in the main menu.

Block 3: Get Integer Input Function

```
# Function to get integer input from the user
def get_integer_input(prompt):
    while True:
        try:
            return int(input(prompt))
        except ValueError:
            print("Please enter a valid integer.")
```

Explanation:

- The `get_integer_input` function repeatedly prompts the user for an integer input until a valid integer is entered.

Block 4: Add Inventory Function

```
# Function to add a new item to the inventory
def add_inventory(inventory, categories):
    print("\nAdding Inventory")
    print("================")
```

```
    item_description = input("Enter the name of the item: ").lower()
    item_quantity = get_integer_input("Enter the quantity of the item: ")
    item_category = input("Enter the category of the item: ").lower()

    # Ensure the category exists in the categories dictionary
    categories.setdefault(item_category, [])
    inventory[item_description] = {"quantity": item_quantity, "category": item_category}
```

Explanation:

- The `add_inventory` function adds a new item to the inventory.
- It prompts the user for the item's name, quantity, and category.
- The function ensures that the category exists in the `categories` dictionary.
- The item is then added to the `inventory` dictionary with details such as quantity and category.

Block 5: Remove Inventory Function

```
# Function to remove an item from the inventory
def remove_inventory(inventory):
    print("\nRemoving Inventory")
    print("==================")
    item_description = input("Enter the item name to remove from inventory: ").lower()
    if item_description in inventory:
        del inventory[item_description]
        print("Item removed from inventory.")
    else:
        print("Item not found in inventory.")
```

Explanation:

- The `remove_inventory` function removes an item from the inventory.

- It prompts the user for the name of the item to be removed.

- If the item is found in the `inventory` dictionary, it is removed; otherwise, a message is displayed.

Block 6: Update Inventory Function

```
# Function to update the quantity of an item in the inventory
def update_inventory(inventory):
    print("\nUpdating Inventory")
    print("==================")
    item_description = input('Enter the item to update: ').lower()
    if item_description in inventory:
        item_quantity = get_integer_input("Enter the updated quantity. Enter - for less: ")
        inventory[item_description]["quantity"] += item_quantity
    else:
        print("Item not found in inventory.")
```

Explanation:

- The `update_inventory` function updates the quantity of an existing item in the inventory.

- It prompts the user for the item to update, and if it exists in the `inventory`, the quantity is modified based on user input.

Block 7: Search Inventory Function

```
# Function to search for an item in the inventory
def search_inventory(inventory):
    print('\nSearching Inventory')
    print('==================')
    item_description = input('Enter the name of the item: ').lower()
    item_info = inventory.get(item_description)
    if item_info:
        print('Item:     ', item_description)
        print('Quantity: ', item_info["quantity"])
        print('Category: ', item_info["category"])
    else:
        print('Item not found in inventory.')
```

Explanation:

- The `search_inventory` function allows the user to search for an item in the inventory.

- It prompts the user for the item's name, and if found in the `inventory`, details such as quantity and category are displayed.

Block 8: Print Inventory Function

```
# Function to print the entire inventory
def print_inventory(inventory):
    print('\nCurrent Inventory')
    print('-----------------')
    for item_description, item_info in inventory.items():
```

```
        print('Item:     ', item_description)
        print('Quantity: ', item_info["quantity"])
        print('Category: ', item_info["category"])
```

Explanation:

- The `print_inventory` function prints the entire content of the inventory.

- It iterates through the items in the `inventory` dictionary and prints details such as item name, quantity, and category.

Block 9: Manage Categories Function

```
# Function to manage categories (add, remove, view)
def manage_categories(inventory, categories):
    while True:
        print('\nCategory Management')
        print('====================')
        print('(1) Add Category')
        print('(2) Remove Category')
        print('(3) View Categories')
        print('(4) View Items by Category')
        print('(5) Return to Main Menu')
        category_choice = get_integer_input('Enter choice: ')

        if category_choice == 1:
            add_category(categories)
        elif category_choice == 2:
```

```
            remove_category(categories)
        elif category_choice == 3:
            view_categories(categories)
        elif category_choice == 4:
            view_items_by_category(inventory, categories)
        elif category_choice == 5:
            # Return to the main menu
            break
        else:
            print("Invalid choice. Please try again.")
```

Explanation:

The manage_categories function provides options for managing categories (add, remove, view).

It uses a loop to repeatedly display the category management menu until the user chooses to return to the main menu.

Depending on the user's choice, various functions are called to perform actions related to categories.

Block 10: Remove Category Function

```
# Function to remove a category
def remove_category(categories):
    category_name = input('Enter the name of the category to remove: ').lower()
    if category_name in categories:
        del categories[category_name]
        print('Category removed: ', category_name)
    else:
```

```
    print('Category not found.')
```

Explanation:

The remove_category function removes a category from the categories dictionary.

It prompts the user for the name of the category to be removed, and if found, it is deleted.

Block 11: Add Category Function

```
# Function to add a new category
def add_category(categories):
    category_name = input('Enter the name of the category: ').lower()
    categories.setdefault(category_name, [])
    print('Category added: ', category_name)
```

Explanation:

The add_category function adds a new category to the categories dictionary.

It prompts the user for the name of the category, and it is added to the dictionary.

Block 12: View Categories Function

```
# Function to view all categories
def view_categories(categories):
    print('\nCategories:')
    for category in categories:
        print(category)
```

Explanation:

The view_categories function prints all existing categories.

Block 13: View Items by Category Function

```
# Function to view all items in a specific category

def view_items_by_category(inventory, categories):

    category_name = input('Enter the name of the category: ').lower()

    if category_name in categories:

        print('\nItems in category: ', category_name)

        for item_description, item_info in inventory.items():

            if item_info['category'] == category_name:

                print('Item:     ', item_description)

                print('Quantity: ', item_info['quantity'])
```

Explanation:

The view_items_by_category function allows the user to view all items in a specific category.

It prompts the user for the category name, and if it exists, all items in that category are printed with their details.

Block 14: Call the Main Function

```
# Call the main function to start the program

main()
```

Explanation:

This block calls the main function, serving as the starting point for the entire program.

Each of these blocks has a specific role in the program, contributing to the overall functionality of the Inventory Management System. If you have any specific questions or need further clarification on any part, feel free to ask!