//**Project: <u>Inventory Management System</u>**

Objective:

Develop a console-based Inventory Management System using OOP, functions, arrays, lists, and CRUD operations in C#

Requirements:

Item Class:

Create an Item class with the following attributes: ID, Name, Price, Quantity.

Inventory Class:

Include appropriate constructors, properties, and methods.

Implement an Inventory class that manages a collection of items.

Use a list to store instances of the Item class

.Include methods for.

.Adding a new item.

.Displaying all items.

.Finding an item by ID.

.Updating an item's information.

.Deleting an item.

Evaluation Criteria:

Proper use of OOP principles (classes, objects, encapsulation, inheritance if applicable). Correct implementation of the Inventory Management System functionalities.

Effective use of functions, arrays, and lists.

Well-organized and readable code with meaningful variable/method names.

Handling of edge cases and input validation.

NOTE:

Feel free to explore additional features or improvements beyond the specified requirements. The goal is to demonstrate a good understanding of OOP concepts and proficiency in using functions, arrays, lists, and loops in C#. Good luck!

TASK:

1)ITEM CLASS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Assignment2
```

```
{



    using System;

    public class Item
    {
        // Properties
        public int ID { get; set; }
        public string Name { get; set; }
        public decimal Price { get; set; }
        public int Quantity { get; set; }

        // Constructor
        public Item(int id, string name, decimal price, int quantity)
        {
            ID = id;
            Name = name;
            Price = price;
            Quantity = quantity;
        }

        // Override ToString method
        public override string ToString()
        {
            return $"ID: {ID}, Name: {Name}, Price: {Price:C}, Quantity: {Quantity}";
        }
    }
}
```

2)INVENTORY CLASS

```
using Assignment2;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Assignment2
{



    using System;
    using System.Collections.Generic;
    using System.Linq;



    public class Inventory
    {
        private List<Item> items;

        public Inventory()
        {
            items = new List<Item>();
        }

        // Method to add a new item
        public void AddItem(Item item)
        {
            items.Add(item);
            Console.WriteLine("Item added successfully.");
        }

        // Method to display all items
        public void DisplayAllItems()
        {
            if (items.Count == 0)
            {
                Console.WriteLine("No items in the inventory.");
                return;
            }

            foreach (var item in items)
            {
                Console.WriteLine(item);
            }
        }

        // Method to find an item by ID
        public Item FindItemByID(int id)
        {
            return items.FirstOrDefault(item => item.ID == id);
        }

        // Method to update an item's name
        public void UpdateItemName(int id, string newName)
        {
            var item = FindItemByID(id);
            if (item == null)
            {
                Console.WriteLine("Item not found.");
                return;
            }

            item.Name = newName;
            Console.WriteLine("Item name updated successfully.");
        }

        // Method to update an item's price
        public void UpdateItemPrice(int id, decimal newPrice)
        {
            var item = FindItemByID(id);
            if (item == null)
            {
                Console.WriteLine("Item not found.");
                return;
            }

            item.Price = newPrice;
            Console.WriteLine("Item price updated successfully.");
        }

        // Method to update an item's quantity
        public void UpdateItemQuantity(int id, int newQuantity)
        {
            var item = FindItemByID(id);
            if (item == null)
            {
                Console.WriteLine("Item not found.");
                return;
            }

            item.Quantity = newQuantity;
            Console.WriteLine("Item quantity updated successfully.");
        }

        // Method to delete an item
        public void DeleteItem(int id)
        {
            var item = FindItemByID(id);
            if (item == null)
            {
                Console.WriteLine("Item not found.");
                return;
            }

            items.Remove(item);
            Console.WriteLine("Item deleted successfully.");
        }
    }
}
```

3)MAIN program

```
using Assignment2;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Assignment2
{




    public class Program
    {

        public static void Main()
        {
            Inventory inventory = new Inventory();

            while (true)
            {
                Console.WriteLine("\nInventory Management System");
                Console.WriteLine("1. Add Item");
                Console.WriteLine("2. Display All Items");
                Console.WriteLine("3. Find Item by ID");
                Console.WriteLine("4. Update Item Name");
                Console.WriteLine("5. Update Item Price");
                Console.WriteLine("6. Update Item Quantity");
                Console.WriteLine("7. Delete Item");
                Console.WriteLine("8. Exit");
            Console.Write("Select an option: ");
            int choice = int.Parse(Console.ReadLine());

            switch (choice)
            {
                case 1:
                    AddNewItem(inventory);
                    break;
                case 2:
                    inventory.DisplayAllItems();
                    break;
                case 3:
                    FindItem(inventory);
                    break;
                case 4:
                    UpdateItemName(inventory);
                    break;
                case 5:
                    UpdateItemPrice(inventory);
                    break;
                case 6:
                    UpdateItemQuantity(inventory);
                    break;
                case 7:
                    DeleteExistingItem(inventory);
                    break;
                case 8:
                    return;
                default:
                    Console.WriteLine("Invalid choice. Please try again.");
                    break;
            }
        }
    }

    private static void AddNewItem(Inventory inventory)
    {
        Console.Write("Enter Item ID: ");
        int id = int.Parse(Console.ReadLine());
        Console.Write("Enter Item Name: ");
        string name = Console.ReadLine();
        Console.Write("Enter Item Price: ");
        decimal price = decimal.Parse(Console.ReadLine());
        Console.Write("Enter Item Quantity: ");
        int quantity = int.Parse(Console.ReadLine());

        Item item = new Item(id, name, price, quantity);
        inventory.AddItem(item);
    }

    private static void FindItem(Inventory inventory)
    {
        Console.Write("Enter Item ID: ");
        int id = int.Parse(Console.ReadLine());

        Item item = inventory.FindItemByID(id);
        if (item != null)
        {
            Console.WriteLine(item);
        }
        else
        {
            Console.WriteLine("Item not found.");
        }
        }

        private static void UpdateItemName(Inventory inventory)
        {
            Console.Write("Enter Item ID to update: ");
            int id = int.Parse(Console.ReadLine());

            Console.Write("Enter new Item Name: ");
            string newName = Console.ReadLine();

            inventory.UpdateItemName(id, newName);
        }

        private static void UpdateItemPrice(Inventory inventory)
        {
            Console.Write("Enter Item ID to update: ");
            int id = int.Parse(Console.ReadLine());

            Console.Write("Enter new Item Price: ");
            decimal newPrice = decimal.Parse(Console.ReadLine());

            inventory.UpdateItemPrice(id, newPrice);
        }

        private static void UpdateItemQuantity(Inventory inventory)
        {
            Console.Write("Enter Item ID to update: ");
            int id = int.Parse(Console.ReadLine());

            Console.Write("Enter new Item Quantity: ");
            int newQuantity = int.Parse(Console.ReadLine());

            inventory.UpdateItemQuantity(id, newQuantity);
        }

        private static void DeleteExistingItem(Inventory inventory)
        {
            Console.Write("Enter Item ID to delete: ");
            int id = int.Parse(Console.ReadLine());
            inventory.DeleteItem(id);
        }
    }

}
```

OUTPUT:

```
Inventory Management System

1. Add Item

2. Display All Items

3. Find Item by ID

4. Update Item Name

5. Update Item Price

6. Update Item Quantity
```

```
7. Delete Item
8. Exit
Select an option: 1
Enter Item ID: 1
Enter Item Name: JAM
Enter Item Price: 20
Enter Item Quantity: 10
Item added successfully.

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 1
Enter Item ID: 2
Enter Item Name: BREAD
Enter Item Price: 40
Enter Item Quantity: 20
Item added successfully.

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 2
```

```
ID: 1, Name: JAM, Price: ? 20.00, Quantity: 10
ID: 2, Name: BREAD, Price: ? 40.00, Quantity: 20

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 3
Enter Item ID: 2
ID: 2, Name: BREAD, Price: ? 40.00, Quantity: 20

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 4
Enter Item ID to update: 1
Enter new Item Name: CREAM
Item name updated successfully.

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
```

```
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 2
ID: 1, Name: CREAM, Price: ? 20.00, Quantity: 10
ID: 2, Name: BREAD, Price: ? 40.00, Quantity: 20

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 5
Enter Item ID to update: 2
Enter new Item Price: 50
Item price updated successfully.

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 2
ID: 1, Name: CREAM, Price: ? 20.00, Quantity: 10
ID: 2, Name: BREAD, Price: ? 50.00, Quantity: 20
```

```
Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 7
Enter Item ID to delete: 2
Item deleted successfully.

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option: 2
ID: 1, Name: CREAM, Price: ? 20.00, Quantity: 10

Inventory Management System
1. Add Item
2. Display All Items
3. Find Item by ID
4. Update Item Name
5. Update Item Price
6. Update Item Quantity
7. Delete Item
8. Exit
Select an option:8  // Here it gets exited
```