Титов Павел Сергеевич

Лабораторная работа №1

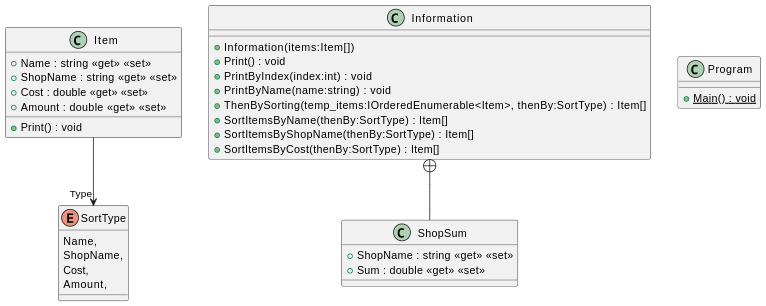
Продвинутый уровень Вариант 14.

Описать класс «товар», содержащий следующие закрытые поля:

* название товара;
* название магазина, в котором продается товар;
* стоимость товара в рублях;
* количество товара.

Предусмотреть свойства для получения состояния объекта.

Описать класс «Справка», содержащий закрытый массив товаров. Обеспечить следующие возможности:

* вывод информации о товаре по номеру с помощью индекса;
* вывод на экран информации о товаре, название которого введено с клавиатуры, если таких товаров нет, выдать соответствующее сообщение;
* сортировку товаров по названию магазина, по наименованию и по цене;
* перегруженную операцию сложения товаров, выполняющую сложение их общей стоимости с группировкой по магазинам.

namespace lab1;

class Item

{

public string Name { get; set; }

public string ShopName { get; set; }

public double Cost { get; set; }

public double Amount { get; set; }

public SortType Type { get; set; }

public void Print()

{

Console.WriteLine($"Info for: {Name}");

Console.WriteLine($"ShopName: {ShopName}");

Console.WriteLine($"Cost: {Cost}");

Console.WriteLine($"Amount: {Amount}");

Console.WriteLine();

}

}

namespace lab1;

enum SortType

{

Name,

ShopName,

Cost,

Amount

}

class Information

{

public class ShopSum

{

public string ShopName { get; set; }

public double Sum { get; set; }

}

private Item[] \_items;

public Item[] GetItems { get => \_items; }

public Item this[int index]

{

get => \_items[index];

set => \_items[index] = value;

}

public Information(Item[] items)

{

\_items = items;

}

public void Print()

{

foreach (var item in \_items)

{

item.Print();

}

}

public void PrintByIndex(int index)

{

if (index >= \_items.Length)

throw new IndexOutOfRangeException(message: $"Максимальное кол-во элементов: {\_items.Length}");

\_items[index].Print();

}

public void PrintByName(string name)

{

var currentItem = Array.Find(\_items, el => el.Name == name);

if (currentItem is null)

{

throw new ArgumentException($"Товара {name} не существует!");

}

currentItem.Print();

}

public Item[] ThenBySorting(IOrderedEnumerable<Item> temp\_items, SortType thenBy)

{

switch (thenBy)

{

case SortType.Cost:

return temp\_items.ThenBy(el => el.Cost).ToArray();

case SortType.ShopName:

return temp\_items.ThenBy(el => el.ShopName).ToArray();

case SortType.Amount:

return temp\_items.ThenBy(el => el.Amount).ToArray();

default:

return temp\_items.ToArray();

}

}

public Item[] SortItemsByName(SortType thenBy = SortType.Name) => ThenBySorting(\_items.OrderBy(el => el.Name), thenBy);

public Item[] SortItemsByShopName(SortType thenBy = SortType.ShopName) => ThenBySorting(\_items.OrderBy(el => el.ShopName), thenBy).ToArray();

public Item[] SortItemsByCost(SortType thenBy = SortType.Cost) => ThenBySorting(\_items.OrderBy(el => el.Cost), thenBy).ToArray();

public static ShopSum[] operator +(Information h1, Information h2)

{

Item[] allItems = h1.GetItems.Union(h2.GetItems).ToArray();

ShopSum[] shops = { };

foreach (var item in allItems)

{

var currentItem = shops.FirstOrDefault(x => x.ShopName.Equals(item.ShopName));

if (currentItem is null)

shops = shops.Append(new ShopSum() { ShopName = item.ShopName, Sum = item.Cost }).ToArray();

else

currentItem.Sum += item.Cost;

}

return shops;

}

}

namespace lab1;

class Program

{

public static void Main()

{

*// Товары*

Item banana = new Item() { Name = "Банан", ShopName = "Абсолют", Cost = 1000, Amount = 2 };

Item banana2 = new Item() { Name = "Банан", ShopName = "Монетка", Cost = 105, Amount = 2 };

Item apple = new Item() { Name = "Яблоко", ShopName = "Монетка", Cost = 200, Amount = 1 };

Item apple2 = new Item() { Name = "Яблоко", ShopName = "Монетка", Cost = 300, Amount = 1 };

*// Справки*

Information info1 = new Information(new Item[] { banana });

Information info2 = new Information(new Item[] { banana, banana2, apple });

*// вывод информации о товаре по номеру с помощью индекса;*

try

{

info1.PrintByIndex(0);

info1.PrintByIndex(100);

}

catch (IndexOutOfRangeException e)

{

Console.WriteLine(e.Message);

}

*// вывод на экран информации о товаре, название которого введено с клавиатуры, если таких товаров нет, выдать соответствующее сообщение;*

try

{

info1.PrintByName("Яблоко");

info1.PrintByName("Ананас");

}

catch (ArgumentException e)

{

Console.WriteLine(e.Message);

}

*// сортировку товаров по названию магазина, по наименованию и по цене;*

Console.WriteLine("Sort example Name");

info2.SortItemsByName(SortType.Cost).ToList().ForEach(shop => shop.Print());

Console.WriteLine("Sort example ShopName");

info2.SortItemsByShopName(SortType.Amount).ToList().ForEach(shop => shop.Print());

Console.WriteLine("Sort example Cost");

info2.SortItemsByCost(SortType.ShopName).ToList().ForEach(shop => shop.Print());

*// перегруженную операцию сложения товаров, выполняющую сложение их общей стоимости с группировкой по магазинам.*

var shops = info1 + info2;

shops.ToList().ForEach(shop => Console.WriteLine($"{shop.ShopName} - {shop.Sum}"));

}

}