МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет информационных технологий и робототехники (ФИТР)

Кафедра программного обеспечения информационных систем и технологий

**О Т Ч Ё Т**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

**«Системы контроля версий. Git»**

**по курсу:**

***"Абстракция"***

|  |  |
| --- | --- |
| Выполнили студенты гр. 10701221: | Шишкунова А. Д.  Прокофьева А. В. |
|  |  |
| Проверил: | Станкевич С.Н. |

Минск, 2023

**Цель работы:** Научиться грамотно анализировать предметную область и с помощью абстракции выделять существенные детали, на базе которых в дальнейшем проектируются классы и объекты будущей программной системы согласно методологии ООП, а также практически закрепить данные навыки при решении соответствующих задач (бизнес проблем).

## Задание: Необходимо решить задачу с использованием методологии ООП. Для чего необходимо подобрать самостоятельно соответствующую проблемную (предметную/доменную) область, которая базируется на объектах и событиях реального мира (примеры соответствующих предметных областей приведены ниже). Спроектировать классы (собственные пользовательские типы данных) для программного представления данных объектов и основной логики будущей программной системы.

Система должна решать, как минимум, два полезных действия и иметь дополнительно следующие опции:

− не менее 3 разнообразных классов предметной области;

− не менее 5 атрибутов (состояния) и методов (поведения) в классе-сущности;

− не менее 3 методов, которые реализуют бизнес-логику программы, в соответствующих функциональных классах;

На базе спроектированной программной системы реализовать программу и продемонстрировать её работоспособность.

Предметная область: Цветочный магазин.

**UML диаграмма классов**

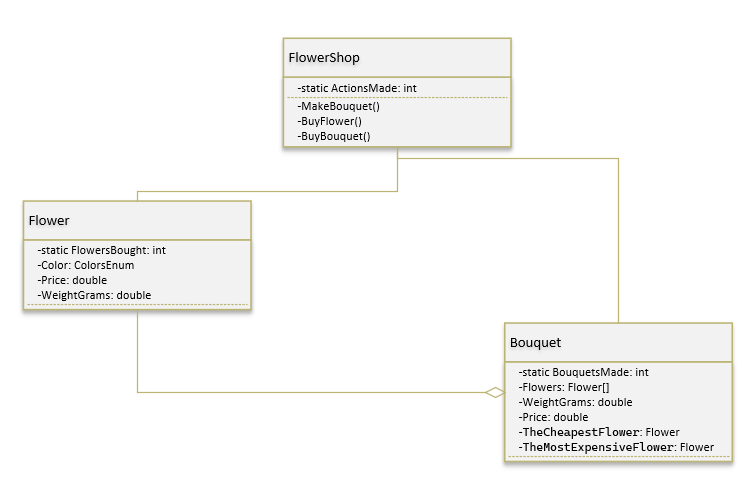


Рис. 1. UML диаграмма классов.

**Результаты отлаженной программы**

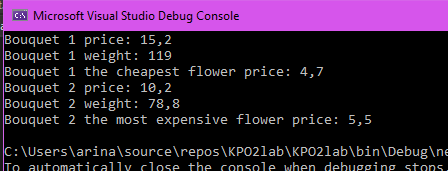


Рис. 2. Результат работы программы.

**Листинг кода**

**Program.cs:**

namespace KPOLab2

{

public class Program

{

public static void Main()

{

var flowerShop = new FlowerShop();

var flowers = new List<Flower>();

flowers.Add(flowerShop.BuyFlower(ColorsEnum.Red));

flowers.Add(flowerShop.BuyFlower(ColorsEnum.Green));

flowers.Add(flowerShop.BuyFlower(ColorsEnum.Blue));

var bouquet1 = flowerShop.MakeBouquet(flowers.ToArray());

Console.WriteLine("Bouquet 1 price: {0}", bouquet1.Price);

Console.WriteLine("Bouquet 1 weight: {0}", bouquet1.WeightGrams);

Console.WriteLine("Bouquet 1 the cheapest flower price: {0}", bouquet1.TheCheapestFlower.Price);

var bouquet2 = flowerShop.BuyBouquet(new ColorsEnum[] { ColorsEnum.Green, ColorsEnum.Blue });

Console.WriteLine("Bouquet 2 price: {0}", bouquet2.Price);

Console.WriteLine("Bouquet 2 weight: {0}", bouquet2.WeightGrams);

Console.WriteLine("Bouquet 2 the most expensive flower price: {0}", bouquet2.TheMostExpensiveFlower.Price);

}

}

}

**FlowerShop.cs:**

namespace KPOLab2

{

internal class FlowerShop

{

public Bouquet MakeBouquet(Flower[] flowers)

{

if (flowers == null) throw new ArgumentException("Flowers array is null!");

return new Bouquet(flowers);

}

public Flower BuyFlower(ColorsEnum color)

{

return new Flower(color);

}

public Bouquet BuyBouquet(ColorsEnum[] colors)

{

if (colors == null) throw new ArgumentException("Colors array is null!");

var flowers = new List<Flower>();

foreach (var color in colors)

{

flowers.Add(new Flower(color));

}

return new Bouquet(flowers.ToArray());

}

}

}

**Bouquet.cs:**

namespace KPOLab2

{

internal class Bouquet

{

public static int BouquetsMade { get; private set; } //number of bouquets made

public Flower[] Flowers { get; private set; }

public double WeightGrams

{

get

{

var weight = 0d;

foreach (var flower in Flowers)

{

weight += flower.WeightGrams;

}

return weight;

}

}

public double Price

{

get

{

var price = 0d;

foreach (var flower in Flowers)

{

price += flower.Price;

}

return price;

}

}

public Flower TheCheapestFlower

{

get

{

var theCheapestFlower = Flowers[0];

foreach (var flower in Flowers)

{

if (flower.Price < theCheapestFlower.Price)

theCheapestFlower = flower;

}

return theCheapestFlower;

}

}

public Flower TheMostExpensiveFlower

{

get

{

var theMostExpensiveFlower = Flowers[0];

foreach (var flower in Flowers)

{

if (flower.Price > theMostExpensiveFlower.Price)

theMostExpensiveFlower = flower;

}

return theMostExpensiveFlower;

}

}

public Bouquet(Flower[] flowers)

{

if (flowers == null) throw new ArgumentException("Flowers array is null!");

Flowers = flowers;

BouquetsMade++;

}

}

}

**Flower.cs:**

namespace KPOLab2

{

public enum ColorsEnum

{

Red,

Green,

Blue

}

internal class Flower

{

public static int FlowersBought { get; private set; } //number of flowers bought

public readonly ColorsEnum Color;

public readonly double Price;

public readonly double WeightGrams;

public Flower(ColorsEnum color)

{

Color = color;

if (color == ColorsEnum.Red)

{

Price = 5;

WeightGrams = 40.2;

}

else if (color == ColorsEnum.Green)

{

Price = 5.5;

WeightGrams = 37.8;

}

else

{

Price = 4.7;

WeightGrams = 41;

}

FlowersBought++;

}

}

}

## Вывод

Научился грамотно анализировать предметную область и с помощью абстракции выделять существенные детали, на базе которых в дальнейшем проектируются классы и объекты будущей программной системы согласно методологии ООП, а также практически закрепил данные навыки при решении соответствующих задач (бизнес проблем).