Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

По дисциплине «СПП»

за 5-й семестр

Выполнил:
студент 2 курса
группы ПО-3 (1)
Афанасьев В.В.

Проверил:
Крощенко А.А.

Брест, 2020

Цель работы: приобрести базовые навыки в области объектно-ориентированного проектирования на языке программирования C#.

Вариант: 2

Задание 1:

Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

2) Создать класс Payment (покупка) с внутренним классом, с помощью объектов которого можно сформировать покупку из нескольких товаров.

Задание 2:

Реализовать агрегирование. При создании класса агрегируемый класс объявляется как атрибут (локальная переменная, параметр метода). Включить в каждый класс 2-3 метода на выбор. Продемонстрировать использование разработанных классов.

2) Создать класс Абзац, используя класс Строка.

Задание 3:

Построить модель программной системы с применением отношений (обобщения, агрегации, ассоциации, реализации) между классами. Задать атрибуты и методы классов. Реализовать (если необходимо) дополнительные классы. Продемонстрировать работу разработанной системы.

2) Система ***Платежи***. ***Клиент*** имеет ***Счет*** в банке и ***Кредитную Карту (КК)***. ***Клиент*** может оплатить ***Заказ***, сделать платеж на другой ***Счет***, заблокировать ***КК*** и аннулировать ***Счет***. ***Администратор*** может заблокировать ***КК*** за превышение кредита.

Код программы:

1)

```
using System;
using System.Collections.Generic;
using System.Reflection;

namespace task1
{
    class Program
    {
        static void Main(string[] args)
        {
            Payment database = new Payment();
            database.AddToList("Car", 2500);
            database.AddToList("Pen", 17);
            Console.WriteLine(database.FullPrice());
            database.ShowInfo();
            Console.WriteLine();
            database.DeleteFromList(1);
            database.ShowInfo();
        }
    }

    class Payment
    {
        List<Good> Goods = new List<Good>();

        public void AddToList(string _Name, int _Price)
        {
            Good obj = new Good(_Name, _Price);
            Goods.Add(obj);
        }
```

```
        public void DeleteFromList(int index)
        {
            Goods.RemoveAt(index);
        }

        public int FullPrice()
        {
            int sum = 0;
            foreach (var item in Goods)
            {
                sum += item.Price;
            }
            return sum;
        }

        public void ShowInfo()
        {
            foreach (var item in Goods)
            {
                Console.WriteLine(item.Name + " " + item.Price);
            }
        }

        public class Good
        {
            public Good(string _Name, int _Price)
            {
                Name = _Name;
                Price = _Price;
            }
            public string Name { get; set; }
            public int Price { get; set; }
        }
    }

}
```

2)

```
using System;

namespace task2
{
    class Program
    {
        static void Main(string[] args)
        {
            MString str1 = new MString("First example. ");
            MString str2 = new MString("Second example. ");

            Paragraph par = new Paragraph();
            par.Add(str1);
            par.Show();
            par.Add(str2);
            par.Show();
        }
    }

    class Paragraph
    {
        public string Value { get; set; }

        public void Add(MString str)
        {
            Value += str.ToString();
        }

        public void Show()
        {
            Console.WriteLine(Value);
        }

        public override string ToString()
        {
            return Value;
        }
    }
```

```
    class MString
    {
        public string Value { get; set; }

        public MString(string _Value)
        {
            Value = _Value;
        }

        public override string ToString()
        {
            return Value;
        }

        public void Show()
        {
            Console.WriteLine(Value);
        }
    }
}
```

3)

```
using System;
using System.Collections.Generic;
using System.Numerics;

namespace task3
{
    class Program
    {
        static void Main(string[] args)
        {
            Payments.Client client1 = new Payments.Client();
            Payments.Client client2 = new Payments.Client();

            Good good1 = new Good
            {
                Sum = 200,
            };

            Payments.Administrator admin = new Payments.Administrator();

            Console.WriteLine("Count client1: "+client1.GetCount());
            client1.Pay(good1);

            Console.WriteLine("Count client1: " + client1.GetCount());
            Console.WriteLine("Count client2: " + client2.GetCount());

            client1.PayTo(client2.GetAccount(), 10000);
            Console.WriteLine("Count client1: " + client1.GetCount());
            Console.WriteLine("Count client2: " + client2.GetCount());

            Console.WriteLine("Close Account client2");
            client2.CloseAccount();
            Console.WriteLine("Close Card client2");
            client2.CloseCard();

            Console.WriteLine("Admin close Card client1");
            admin.BlockClientCard(client1);
        }
    }
    public class Good
    {
        public int Sum { get; set; }
    }

    public class Payments
    {
        static public List<Client> Clients = new List<Client>();

        public class User
        {
            // some functional
        }

        public class Client : User
        {
```

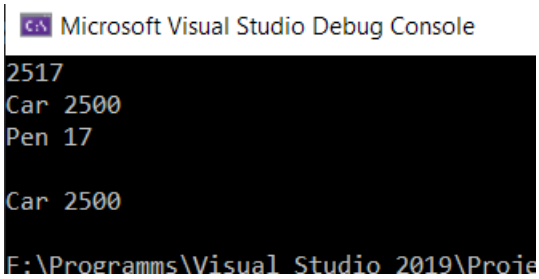
```
    Account account;
    CCard card;

    public Client()
    {
        account = new Account(5000);
        card = new CCard(account);
        Clients.Add(this);
    }

    public void Pay(Good good)                  // using Card
    {
        card.Pay(good);
    }

    public void PayTo(Account other, int sum)   // using Card
    {
        card.PayTo(other, sum);
    }

    public void CloseCard()                     // using Card
    {
        card.Close();
    }

    public void CloseAccount()                  // using Account
    {
        account.CloseAccount();
    }

    public int GetCount()
    {
        return card.Count();
    }

    public Account GetAccount()
    {
        return account;
    }
}

public class Administrator : User
{
    public void BlockClientCard(Client obj)
    {
        if (obj.GetCount() < 0)
        {
            obj.CloseCard();
        }
        else Console.WriteLine("Card is not blocked. The count is correct.");
    }
}

public class CCard
{
    public Account Account;

    public bool Closed;

    public CCard(Account _account)                  // any Card has Account
    {
        Closed = false;
        Account = _account;
    }

    public void Close()
    {
        Closed = true;
        Console.WriteLine("The card was closed.");
    }

    public int Count()                              // return Count from Account
    {
        if (Closed)
        {
            Console.WriteLine("Card is locked");
            return 0;
        }
        else return Account.Count;
```

```
        }

        public void Pay(Good obj)                       // taking Good and change our Count
        {
            if (Closed)
            {
                Console.WriteLine("Card is locked");
                return;
            }
            else
            {
                Account.TakeSum(obj.Sum);
                Console.WriteLine("The good was paid.");
            }
        }

        public void PayTo(Account other, int sum)
        {
            if (Closed)
            {
                Console.WriteLine("Card is locked");
                return;
            }
            else
            {
                Account.TakeSum(sum);
                other.AddSum(sum);
                Console.WriteLine("The sum was sent to the other client.");
            }
        }
    }

    public class Account
    {
        public int Number { get; private set; }         // the private number of the Account

        public int Count { get; set; }                  // the Count

        public bool Validation { get; private set; }    // private Validation

        public void CloseAccount()
        {
            Validation = false;
            Console.WriteLine("The account was closed.");
        }

        public Account(int _count)
        {
            Random random = new Random();
            Number = random.Next(1000, 9999);           // the number is random value

            Count = _count;                             // open on our private Sum
            Validation = true;                          // default - Account is valid
        }

        public void AddSum(int sum)                     // add some sum to Count
        {
            if(!Validation)
            {
                Console.WriteLine("Account is not valid");
                return;
            }
            else Count += sum;
        }

        public void TakeSum(int sum)                     // take some sum from Count
        {
            if (!Validation)
            {
                Console.WriteLine("Account is not valid");
                return;
            }
            else
            {
                Count -= sum;
            }
        }
    }
}
```

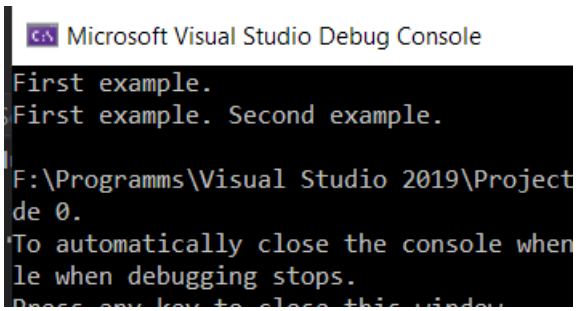
```
}
```

Результаты работы:

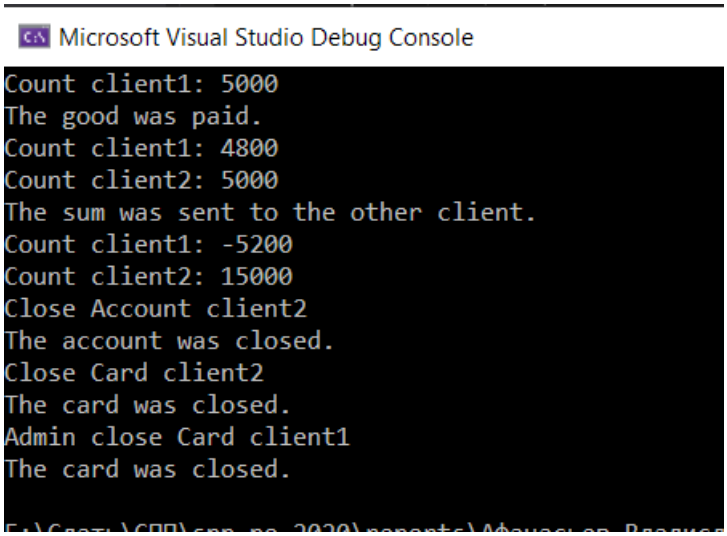
1)



2)



3)



Выводы: в ходе выполнения лабораторной работы были получены базовые навыки в области объектно-ориентированного проектирования на языке программирования C#.