МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 5

з дисципліни “Програмування”
тема “Шаблони проєктування”

Виконав студент
ІІ курсу групи КП-01
Тітов Єгор Павлович
Варіант 16

Перевірив
“____” “_____” 20___р
викладач
Заболотня Тетяна Миколаївна

Київ 2021

Постановка задач

1. В залежності від кількості часу, який домогосподарка може витратити на прибирання квартири, існує 3 варіанти прибирання: легке прибирання (скласти розкидані речі, підмести підлогу), звичайне прибирання (витерти пил, використати пилосос) та генеральне прибирання (вологе прибирання підлоги, а також миття вікон). За допомогою шаблону проєктування реалізувати дані види прибирання.
2. Для того, щоб отримати товари для продажу, супермаркет не звертається напряму до виробників, а направляє відповідний запит постачальнику товарів, який має координати виробників товарів та допомагає супермаркету замовити товари, які характеризуються потрібною якістю та ціною. За допомогою шаблону проєктування змоделювати процес надходження товарів на прилавки супермаркету.

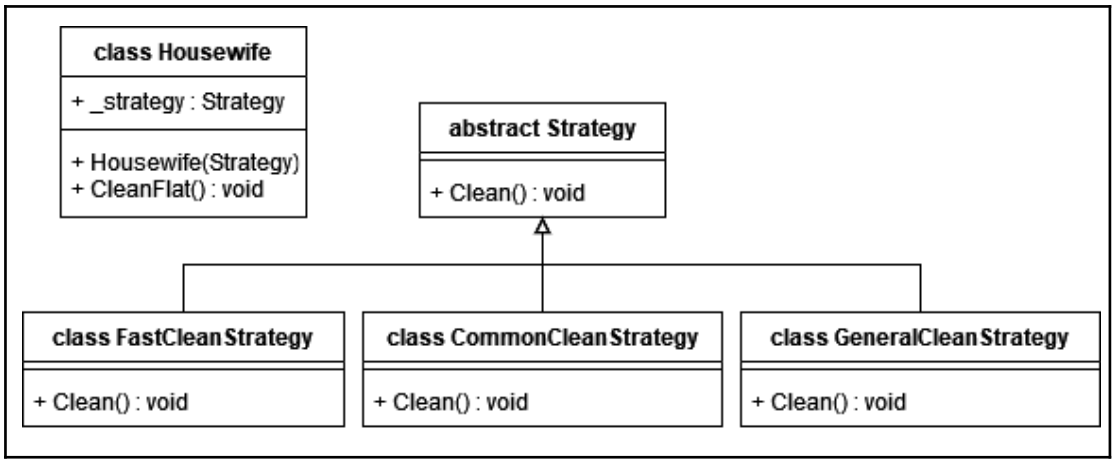
Обгрунтування вибору шаблонів

1. Маємо різні варіації дій при різних умовах, що дозволяє втілити шаблон “Strategy”.

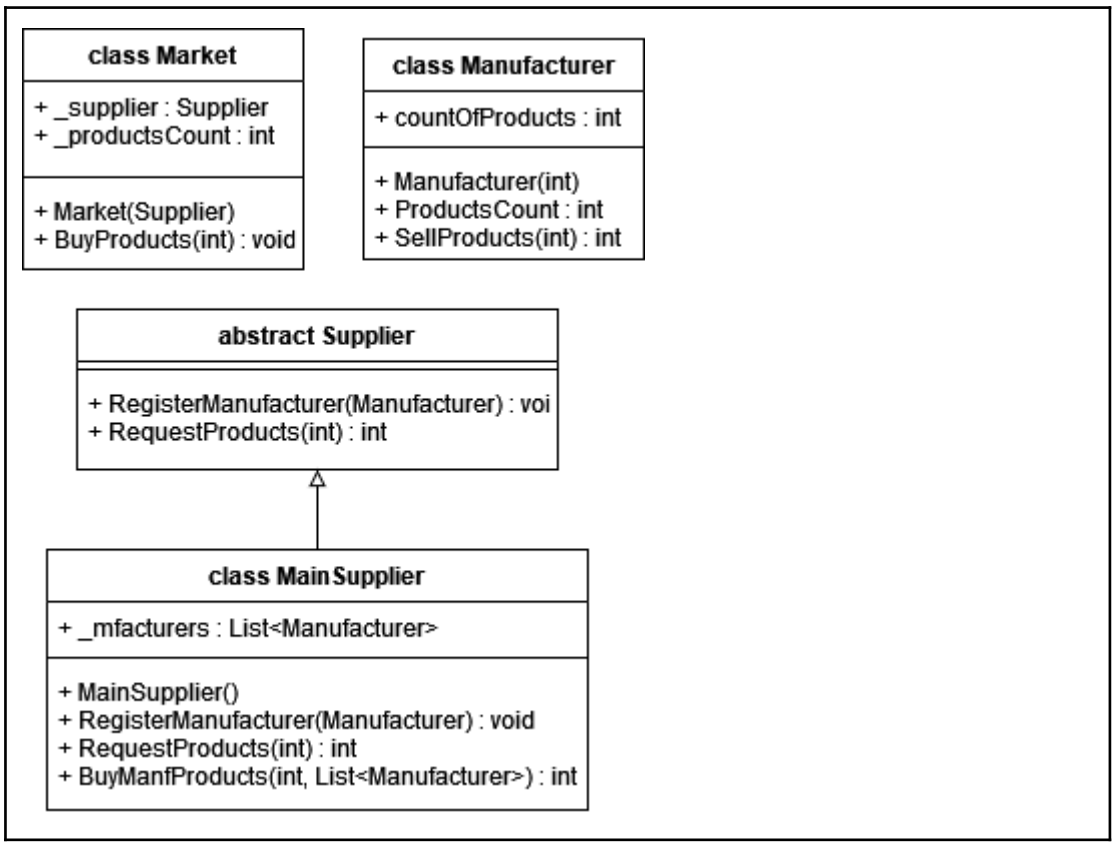
2. Маємо супермаркет, що через посередника звертається до виробника. Ключовим словом є “посередник”, концепцію якого можна втілити використовуючи шаблон “Mediator”.

UML діаграма класів

1. Завдання 1



2. Завдання 2



Текст програми

Завдання 1

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

namespace No_1
{
    class Program
    {
        static void Main(string[] args)
        {
            Dictionary<string, Strategy> possibleBeverageBuild = new
Dictionary<string, Strategy>();
            possibleBeverageBuild.Add("fast", new FastCleanStrategy());
            possibleBeverageBuild.Add("common", new CommonCleanStrategy());
            possibleBeverageBuild.Add("general", new
GeneralCleanStrategy());

            Console.Write("Choose cleaning type(fast/common/general): ");
            string choice = Console.ReadLine();

            Strategy strategy;

            if (possibleBeverageBuild.TryGetValue(choice, out strategy))
            {
                Console.WriteLine($"Cleaning '{choice}' in progress...");

                Housewife hw = new Housewife(strategy);

                hw.CleanFlat();
            }
            else
            {
                Console.WriteLine($"Clean strategy '{choice}' does not
exist");
            }
        }
    }
}
```

```
namespace No_1
{
    class Housewife
    {
        private Strategy _strategy;

        public Housewife(Strategy strategy)
        {
            _strategy = strategy;
        }

        public void CleanFlat()
        {
            _strategy.Clean();
        }
    }
}
```

```
namespace No_1
{
    abstract class Strategy
    {
        public abstract void Clean();
    }
}
```

```
using System;

namespace No_1
{
    class FastCleanStrategy : Strategy
    {
        public override void Clean()
        {
            Console.WriteLine("Scattered things are stacked; The floor was
swept");
        }
    }
}
```

```
using System;
```

```
namespace No_1
{
    class CommonCleanStrategy : Strategy
    {
        public override void Clean()
        {
            Console.WriteLine("The dust has been wiped away; Vacuum cleaner
used");
        }
    }
}
```

```
using System;

namespace No_1
{
    class GeneralCleanStrategy : Strategy
    {
        public override void Clean()
        {
            Console.WriteLine("Wet cleaning of the floor carried out; The
windows were washed");
        }
    }
}
```

2. Завдання 2

```
namespace No_2
{
    class Program
    {
        static void Main(string[] args)
        {
            Manufacturer man1 = new Manufacturer(100);
            Manufacturer man2 = new Manufacturer(50);
            Manufacturer man3 = new Manufacturer(70);

            Supplier sup = new MainSupplier();
```

```
            sup.RegisterManufacturer(man1);
            sup.RegisterManufacturer(man2);
            sup.RegisterManufacturer(man3);

            Market market = new Market(sup);
            market.BuyProducts(100);
            market.BuyProducts(100);
            market.BuyProducts(100);
        }
    }
}
```

```
using System;

namespace No_2
{
    class Market
    {
        private Supplier _supplier;
        private int _productsCount = 0;

        public Market(Supplier sup)
        {
            _supplier = sup;
        }

        public void BuyProducts(int count)
        {
            if (_supplier != null)
            {
                Console.WriteLine($"Requesting '{count}' products in
supplier");

                int receivedCount = _supplier.RequestProducts(count);

                _productsCount += receivedCount;

                Console.WriteLine($"Supplier has provided '{receivedCount}'
products");
            }
        }
```

```
    }
}
```

```
namespace No_2
{
    class Manufacturer
    {
        protected int countOfProducts;

        public Manufacturer(int countOfProducts)
        {
            this.countOfProducts = countOfProducts;
        }

        public int ProductsCount
        {
            get
            {
                return countOfProducts;
            }
            set
            {
                if (countOfProducts >= 0)
                {
                    this.countOfProducts = value;
                }
            }
        }

        public int SellProducts(int requestedCount)
        {
            int sellCount = 0;

            if (countOfProducts < requestedCount)
            {
                sellCount = countOfProducts;

                countOfProducts = 0;
            }
            else
            {
```

```
                sellCount = requestedCount;

                countOfProducts -= requestedCount;
            }

            return sellCount;
        }
    }
}
```

```
namespace No_2
{
    abstract class Supplier
    {
        public abstract void RegisterManufacturer(Manufacturer
manufacturer);
        public abstract int RequestProducts(int requestedCount);
    }
}
```

```
using System.Collections.Generic;

namespace No_2
{
    class MainSupplier : Supplier
    {
        private List<Manufacturer> _mfacturers;

        public MainSupplier()
        {
            this._mfacturers = new List<Manufacturer>();
        }

        public override void RegisterManufacturer(Manufacturer manufacturer)
        {
            if (!_mfacturers.Contains(manufacturer))
            {
                _mfacturers.Add(manufacturer);
            }
        }
```

```
        public override int RequestProducts(int requestedCount)
        {
            List<Manufacturer> currentMarketManufacturers = new
List<Manufacturer>();
            int guarantedCount = 0;

            foreach (Manufacturer manf in _mfacturers)
            {
                guarantedCount += manf.ProductsCount;
                currentMarketManufacturers.Add(manf);

                if (guarantedCount >= requestedCount)
                {
                    guarantedCount = requestedCount;

                    break;
                }
            }

            int receivedCount = BuyManfProducts(guarantedCount,
currentMarketManufacturers);

            return receivedCount;
        }

        private int BuyManfProducts(int count, List<Manufacturer> list)
        {
            int receivedCount = 0;

            foreach (Manufacturer manf in list)
            {
                int soldProds = manf.SellProducts(count);

                receivedCount += soldProds;

                count -= soldProds;
            }
            return receivedCount;
        }
    }
}
```

Приклади результатів

Завдання 1:

```
Choose cleaning type(fast/common/general): common
Cleaning 'common' in progress...
The dust has been wiped away; Vacuum cleaner used
```

Завдання 2:

```
Requesting '100' products in supplier
Supplier has provided '100' products
Requesting '100' products in supplier
Supplier has provided '100' products
Requesting '100' products in supplier
Supplier has provided '20' products
```

Висновки

В результаті виконання лабораторної роботи я ознайомився і використав для вирішення поставлених завдань такі шаблони як “Strategy” і “Mediator”. Внаслідок цього я зрозумів переваги використання поведінкових шаблонів проєктування і як важливо правильно їх застосовувати.