Практическая работа № 1.  
шаблон проектирования стратегия

**Цель практической работы**

Закрепление теоретических знаний по основам применения шаблонов проектирования при разработке приложений.

**Постановка задачи**

Паттерны проектирования — это проверенные решения распространенных проблем, с которыми разработчики сталкиваются при проектировании программного обеспечения. Они помогают создать гибкую и поддерживаемую архитектуру, улучшая читаемость и повторное использование кода.

Паттерн "Стратегия" (Strategy) относится к поведенческим паттернам проектирования. Он позволяет определять семейство алгоритмов, инкапсулировать каждый из них и делать их взаимозаменяемыми. Паттерн позволяет изменять алгоритм независимо от клиентов, которые его используют.

Основная идея паттерна заключается в том, чтобы выделить алгоритмы в отдельные классы, которые реализуют общий интерфейс. Это позволяет динамически менять алгоритмы во время выполнения программы, не изменяя код, который их использует.

**Структура паттерна**

Паттерн "Стратегия" состоит из следующих компонентов:

1. **Контекст (Context)**: класс, который использует стратегию. Он содержит ссылку на объект стратегии и делегирует выполнение алгоритма этому объекту.
2. **Стратегия (Strategy)**: интерфейс, который определяет общий контракт для всех конкретных стратегий.
3. **Конкретные стратегии (ConcreteStrategy)**: классы, которые реализуют интерфейс стратегии и предоставляют конкретные алгоритмы.

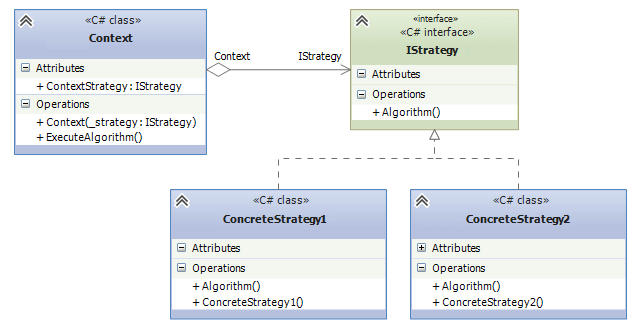


Рисунок 1 – Структура паттерна «Стратегия»

**Преимущества паттерна "Стратегия"**

1. **Гибкость**: Позволяет легко добавлять новые алгоритмы без изменения существующего кода.
2. **Повторное использование кода**: Общий интерфейс позволяет использовать разные алгоритмы в одном и том же контексте.
3. **Упрощение кода**: Уменьшает количество условных операторов (if/else) в коде, так как выбор алгоритма осуществляется через стратегию.

**Задание на практическую работу**

1. В соответствии с паттерном «**Стратегия**» спроектировать структуру программы для сортировки произвольным алгоритмом.
2. Предусмотреть:

а) выбор алгоритма сортировки (по вариантам),

б) выбор направления сортировки (возрастание/убывание),

в) ввод значений,

г) генерация случайного набора данных,

д) отображение времени выполнения сортировки.

1. Определить структуру интерфейса стратегии, конкретные стратегии и контекст стратегии.
2. **Для оценки отлично**:

Разработанная структура должна предусматривать:

а) произвольный тип данных элементов,

б) произвольный тип коллекции.

**Пример разработки приложения**

Рассмотрим пример, в котором есть несколько алгоритмов сортировки, и мы хотим использовать паттерн "Стратегия" для их реализации.

**Шаг 1: Определение интерфейса стратегии**

public interface ISortStrategy

{

    void Sort(int[] array);

}

**Шаг 2: Реализация конкретных стратегий**

public class BubbleSort : ISortStrategy

{

    public void Sort(int[] array)

    {

        for (int i = 0; i < array.Length - 1; i++)

        {

            for (int j = 0; j < array.Length - i - 1; j++)

            {

                if (array[j] > array[j + 1])

                {

                    // Обмен элементов

                    int temp = array[j];

                    array[j] = array[j + 1];

                    array[j + 1] = temp;

                }

            }

        }

    }

}

public class QuickSort : ISortStrategy

{

    public void Sort(int[] array)

    {

        QuickSortAlgorithm(array, 0, array.Length - 1);

    }

    private void QuickSortAlgorithm(int[] array, int low, int high)

    {

        if (low < high)

        {

            int pi = Partition(array, low, high);

            QuickSortAlgorithm(array, low, pi - 1);

            QuickSortAlgorithm(array, pi + 1, high);

        }

    }

    private int Partition(int[] array, int minIndex, int maxIndex)

    {

        var pivot = minIndex - 1;

        for (var i = minIndex; i < maxIndex; i++)

        {

            if (array[i] < array[maxIndex])

            {

                pivot++;

                Swap(ref array[pivot], ref array[i]);

            }

        }

        pivot++;

        Swap(ref array[pivot], ref array[maxIndex]);

        return pivot;

    }

    private void Swap(ref int x, ref int y)

    {

        var t = x;

        x = y;

        y = t;

    }

}

**Шаг 3: Реализация контекста**

public class SortContext

{

    private ISortStrategy \_sortStrategy;

    public SortContext(ISortStrategy sortStrategy)

    {

        \_sortStrategy = sortStrategy;

    }

    public void SetSortStrategy(ISortStrategy sortStrategy)

    {

        \_sortStrategy = sortStrategy;

    }

    public void Sort(int[] array)

    {

        \_sortStrategy.Sort(array);

    }

}

**Шаг 4: Использование паттерна "Стратегия"**

class Program

{

    static void Main(string[] args)

    {

        int[] array = { 5, 3, 8, 6, 2 };

        SortContext context = new SortContext(new BubbleSort());

        Console.WriteLine("Сортировка пузырьком:");

        context.Sort(array);

        Console.WriteLine(string.Join(", ", array));

        // Изменение стратегии на быструю сортировку

        array = new int[] { 5, 3, 8, 6, 2 };

        context.SetSortStrategy(new QuickSort());

        Console.WriteLine("Быстрая сортировка:");

        context.Sort(array);

        Console.WriteLine(string.Join(", ", array));

    }

}

# Содержание пояснительной записки

1. Постановка задачи. Приводится теоретический материал, использованный при написании приложения.

2. Формулировка задания и вариант. Приводится задание на лабораторную работу и вариант этого задания.

3. Описание выполняемых действий. Необходимо привести описание последовательности разработки программы, реализации используемых методов, алгоритмов, блок-схем.

4. Анализ результатов. Привести анализ входных и выходных данных. Показать результаты выполнения программного кода. Предоставить скриншоты обработки тестовых примеров. Сделать выводы.

5. Листинг программы. Привести листинг разработанного программного кода, содержание файлов входных и выходных данных.

# Используемое программное обеспечение

1. Среда программирования MS Visual Studio Community 2022 (Свободно распространяемое программное обеспечение (в учебных целях));
2. Microsoft Office Standard 2007 (Open License: 42267924);
3. Open Office (Свободно распространяемое программное обеспечение).
4. Браузер (Свободно распространяемое программное обеспечение).

# Список литературы

* + - 1. Мейер Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс]/ Мейер Б. – Электрон. текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 285 c.
      2. Биллиг, В. A. Основы объектного программирования на С# (C# 3.0, Visual Studio 2008) [Электронный ресурс] : учебное пособие / В. A. Биллиг. — Электрон. текстовые данные. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 583 c. — 978-5-4487-0145-0. — Режим доступа: http://www.iprbookshop.ru/72339.html
      3. Павловская, Т. А. Программирование на языке высокого уровня C# [Электронный ресурс] / Т. А. Павловская. — 2-е изд. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 245 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/73713.html
      4. Агапов, В. П. Основы программирования на языке С# [Электронный ресурс] : учебное пособие / В. П. Агапов. — Электрон. текстовые данные. — М. : Московский государственный строительный университет, ЭБС АСВ, 2012. — 128 c. — 978-5-7264-0576-6. — Режим доступа: http://www.iprbookshop.ru/16366.html
      5. Медведев, М. А. Программирование на СИ# [Электронный ресурс] : учебное пособие / М. А. Медведев, А. Н. Медведев ; под ред. А. В. Присяжный. — Электрон. текстовые данные. — Екатеринбург : Уральский федеральный университет, ЭБС АСВ, 2015. — 64 c. — 978-5-7996-1561-1. — Режим доступа: http://www.iprbookshop.ru/69667.html
      6. Казанский А.А. Объектно-ориентированное программирование на языке Microsoft Visual С# в среде разработки Microsoft Visual Studio 2008 и .NET Framework. 4.3 [Электронный ресурс]: учебное пособие и практикум/ Казанский А.А.— Электрон. текстовые данные.— М.: Московский государственный строительный университет, ЭБС АСВ, 2011.— 180 c
      7. Уйманова Н.А. Основы объектно-ориентированного программирования [Электронный ресурс]: практикум/ Уйманова Н.А., Таспаева М.Г.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2017.— 156 c.— Режим доступа: http://www.iprbookshop.ru/78808.html.— ЭБС «IPRbooks»
      8. Новиков П.В. Объектно-ориентированное программирование [Электронный ресурс]: учебно-методическое пособие к лабораторным работам/ Новиков П.В.— Электрон. текстовые данные.— Саратов: Вузовское образование, 2017.— 124 c.— Режим доступа: http://www.iprbookshop.ru/64650.html.— ЭБС «IPRbooks»