Практическая работа №23
Реализация паттернов проектирования

1Цель работы

1.1 Научиться применять паттерны проектирования в разработке программ.

2 Литература

2.1 https://www.tutorialspoint.com/design_pattern/

2.2 https://metanit.com/sharp/patterns/

2.3 https://www.freecodecamp.org/news/the-basic-design-patterns-all-developers-need-to-know/

3 Задание

Требуется перевести код программ, реализующих паттерны проектирования, с языка Java на язык C# с учетом требований к наименованиям элементов, типов данных, объектов.

На схемах, приведенных на рисунках 1-5, аналог класса Program – это класс, в котором указан метод main(). Класс Program переименовывать не требуется.

Отличия кода на C# от кода на Java:

- для вывода данных вместо System.out.println() используется Console.WriteLine()
- строковый тип данных принято писать с маленькой буквы: string
- названия методов нужно писать с заглавной буквы: ИмяМетода
- названия интерфейсов должны начинаться с буквы I: IИнтерфейс
- не нужно писать public в интерфейсах
- при реализации интерфейсов вместо implements пишется двоеточие
- @Override при реализации интерфейса указывать не нужно
- при наследовании класса вместо implements пишется двоеточие
- при наследовании класса для обращения к родительскому классу вместо super пишется base

3.1 Реализовать поведенческий паттерн «Стратегия» согласно UML-диаграмме на рисунке 1.

https://www.tutorialspoint.com/design_pattern/strategy_pattern.htm

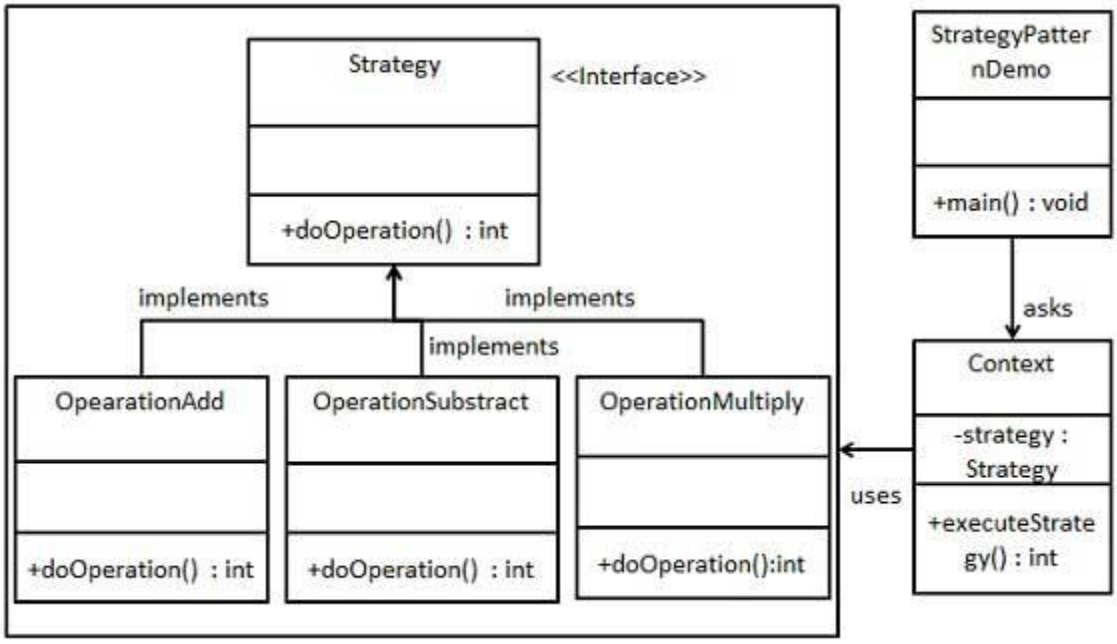

Рисунок 1

3.2 Реализовать порождающий паттерн «Фабричный метод» согласно UML-диаграмме на рисунке 2.

https://www.tutorialspoint.com/design_pattern/factory_pattern.htm

Для сравнения строк без учета регистра используется:

строка1.Equals(строка2, StringComparison.InvariantCultureIgnoreCase)

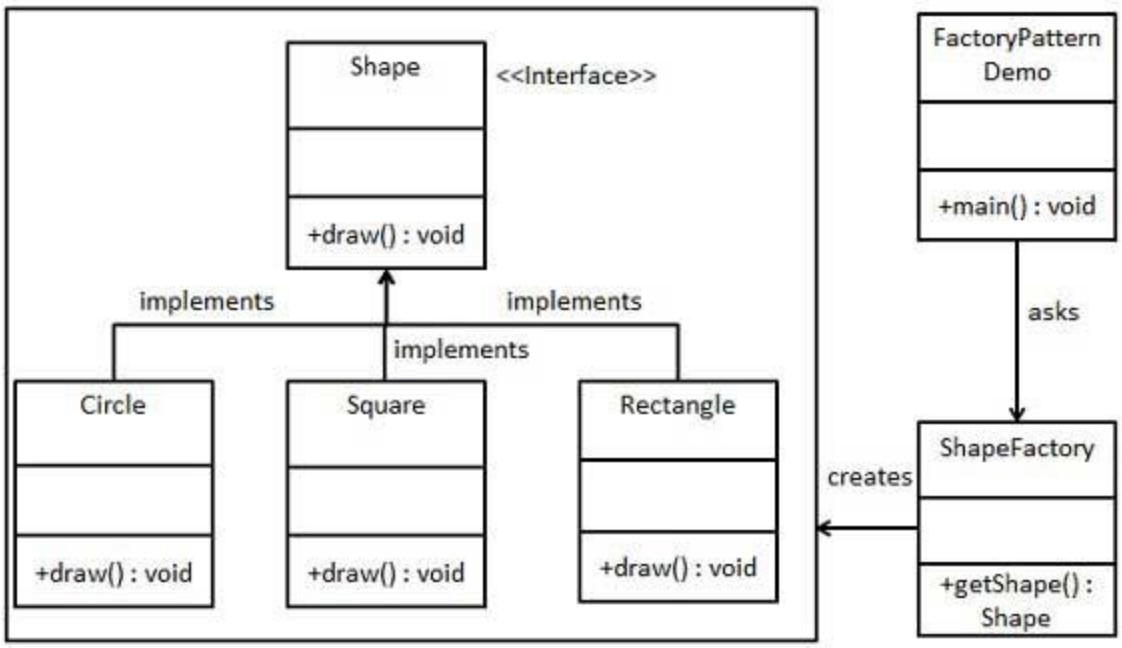

Рисунок 2

3.3 Реализовать структурный паттерн «Адаптер» согласно UML-диаграмме на рисунке 3.

https://www.tutorialspoint.com/design_pattern/adapter_pattern.htm

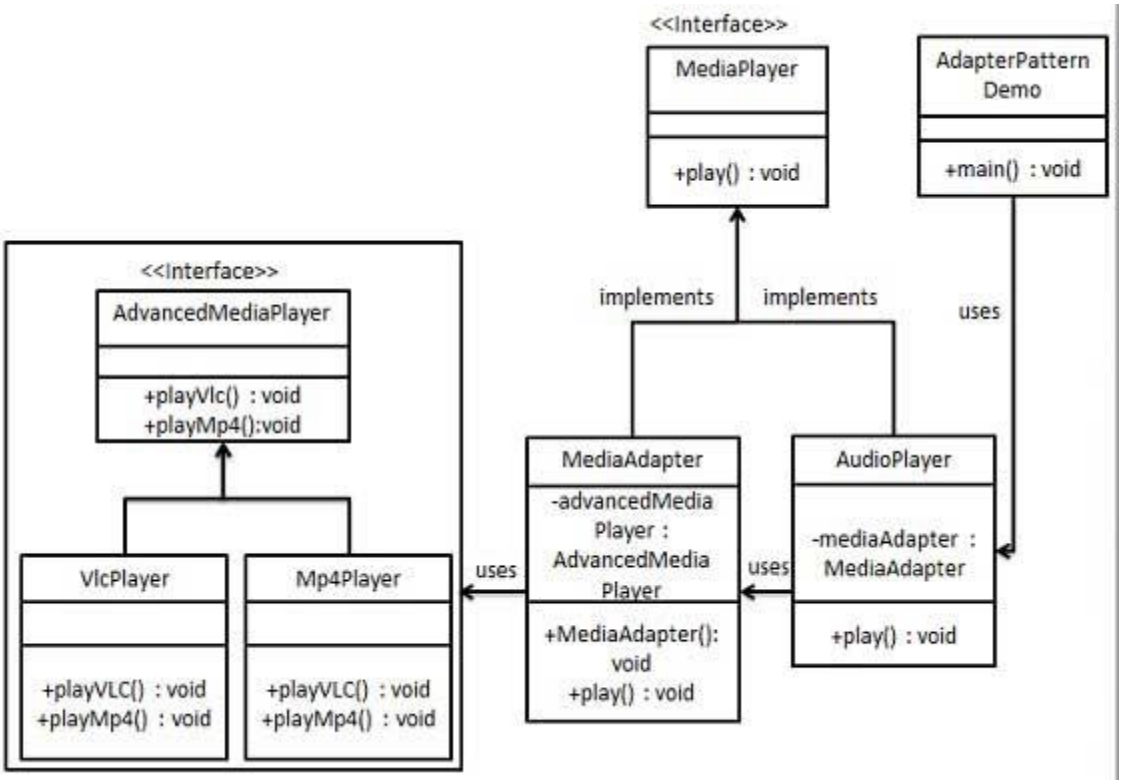

Рисунок 3

3.4 Реализовать поведенческий паттерн «Наблюдатель» согласно UML-диаграмме на рисунке 4.

https://www.tutorialspoint.com/design_pattern/observer_pattern.htm

Для перевода в систему счисления используется:

Convert.ToString(число, основание)

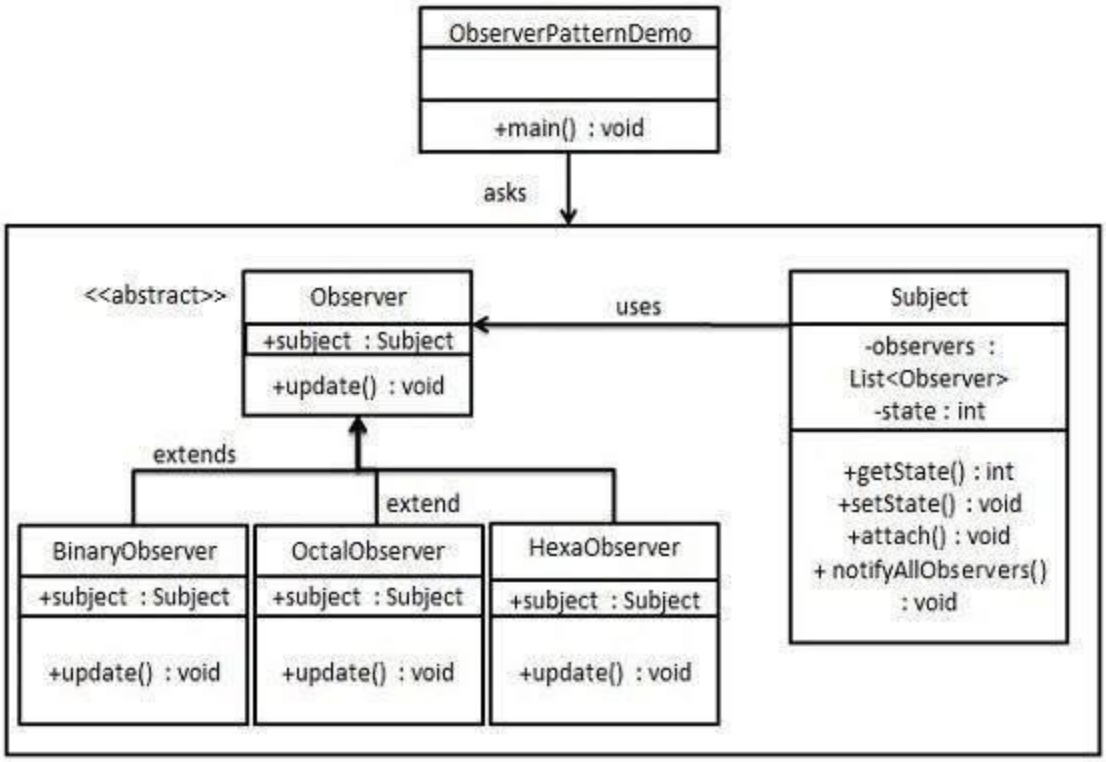

Рисунок 4

3.5 Реализовать паттерн «MVC» согласно UML-диаграмме на рисунке 5.

https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm

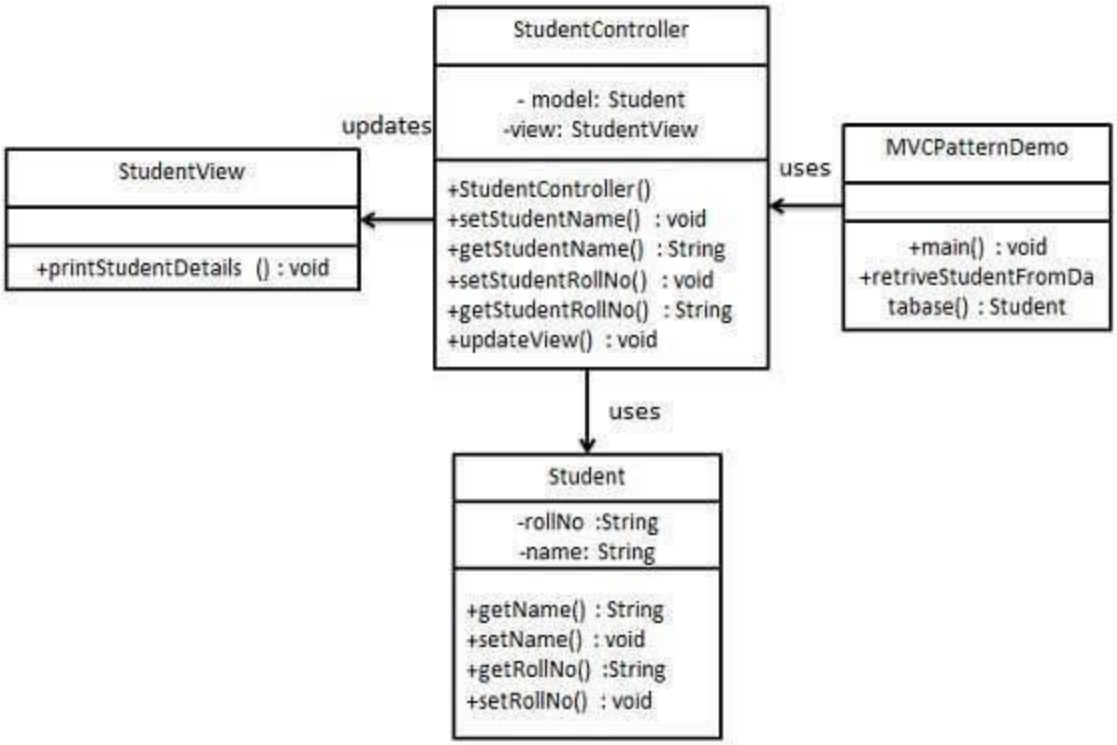

Рисунок 5

4 Порядок выполнения работы

4.1 Выполнить все задания из п.3 в одном решении PractWork23, каждое – в своем консольном проекте. Возможные ошибки требуется обрабатывать. Выполнить форматирование и рефакторинг кода.

4.2 Ответить на контрольные вопросы.

5 Содержание отчета

5.1 Титульный лист
5.2 Цель работы
5.3 Ответы на контрольные вопросы
5.4 Вывод

6 Контрольные вопросы

6.1 Для чего используются порождающие паттерны?
6.2 Какие паттерны относятся к порождающим?
6.3 Для чего используются структурные паттерны?
6.4 Какие паттерны относятся к структурным?
6.5 Для чего используются поведенческие паттерны?
6.6 Какие паттерны относятся к поведенческим?