**Национальный исследовательский технологический университет «МИСИС»**

Отчет по лабораторной работе № 4 «Поведенческие паттерны проектирования»

Выполнил:

Миронов Е. А. БИВТ-23-8

Преподаватель:

Гласов А. В.

Москва 2024 г.

GitHub repo: <https://github.com/Valet-V0ult-de-Fur1e/programming_technologies_repo>

**Strategy** - это поведенческий паттерн проектирования, который определяет семейство алгоритмов, инкапсулирует каждый из них и обеспечивает их взаимозаменяемость.

Это позволяет легко подменять и модифицировать алгоритмы во время выполнения программы без изменения класса, который их использует.

Примеры применения:

* Когда есть несколько родственных классов, которые отличаются поведением;
* Когда необходимо обеспечить выбор из нескольких вариантов алгоритмов, Которые можно легко менять в зависимости от условий;
* Когда необходимо менять поведение объектов на стадии выполнения программы;
* Когда класс, применяющий определённую функциональность, ничего не должен знать о её реализации.

## Chain of responsibility - поведенческий паттерн проектирования, который позволяет избежать жесткой привязки отправителя запроса к получателю. Все возможные обработчики запроса образуют цепочку, а сам запрос перемещается по этой цепочке. Каждый объект в этой цепочке при получении запроса выбирает, либо закончить обработку запроса, либо передать запрос на обработку следующему по цепочке объекту.

### Когда применяется цепочка обязанностей?

* Когда имеется более одного объекта, который может обработать определенный запрос
* Когда надо передать запрос на выполнение одному из нескольких объект, точно не определяя, какому именно объекту
* Когда набор объектов задается динамически

**Iterator** - предоставляет абстрактный интерфейс для последовательного доступа ко всем элементам составного объекта без раскрытия его внутренней структуры.