**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: Шаблонные классы

|  |  |  |
| --- | --- | --- |
| Студент гр. 3341 |  | Шаповаленко Е.В. |
| Преподаватель |  | Жангиров Т.Р. |

Санкт-Петербург

2024

## Цель работы

Изучить и понять концепцию шаблонных классов в языке программирования C++, их применение, преимущества и недостатки, а также научиться создавать и использовать шаблонные классы для решения различных задач. Применить полученные знания на практике, создав шаблонный класс управления игрой.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Изучение основ шаблонов: ознакомиться с синтаксисом и структурой шаблонных классов.
2. Разработать шаблонный класс управления игрой

## Задание

А) Создать шаблонный класс управления игрой. Данный класс должен содержать ссылку на игру. В качестве параметра шаблона должен указываться класс, который определяет способ ввода команда, и переводящий введенную информацию в команду. Класс управления игрой, должен получать команду для выполнения, и вызывать соответствующий метод класса игры.

Б) Создать шаблонный класс отображения игры. Данный класс реагирует на изменения в игре, и производит отрисовку игры. То, как происходит отрисовка игры определяется классом переданном в качестве параметра шаблона.

В) Реализовать класс считывающий ввод пользователя из терминала и преобразующий ввод в команду. Соответствие команды введенному символу должно задаваться из файла. Если невозможно считать из файла, то управление задается по умолчанию.

Г) Реализовать класс, отвечающий за отрисовку поля.

**Примечание**:

* Класс отслеживания и класс отрисовки рекомендуется делать отдельными сущностями. Таким образом, класс отслеживания инициализирует отрисовку, и при необходимости можно заменить отрисовку (например, на GUI) без изменения самого отслеживания
* После считывания клавиши, считанный символ должен сразу обрабатываться, и далее работа должна проводить с сущностью, которая представляет команду.
* Для представления команды можно разработать системы классов или использовать перечисление enum.
* Хорошей практикой является создание “прослойки” между считыванием/обработкой команды и классом игры, которая сопоставляет команду и вызываемым методом игры. Существуют альтернативные решения без явной “прослойки”
* При считывания управления необходимо делать проверку, что на все команды назначена клавиша, что на одну клавишу не назначено две команды, что на одну команду не назначено две клавиши.

## Выполнение работы

Шаблонный класс GameController представляет собой контролер игры. В качестве параметров шаблона он принимает классы ввода/вывода информации. Таким образом можно определить способ взаимодействия с игрой, не создавая каждый раз новый класс управления.

GameController хранит ссылку на управляемую игру, класс-отрисовщик, класс для ввода команд, а также словарь с лямбда-функциями – командами.

Метод start инициализирует контроллер.

Метод startGame начинает новую игру.

Метод restartGame перезапускает игру при победе игрока.

Метод initializeCommands инициализирует команды.

Метод gameCycle реализует логику игрового цикла.

Метод roundCycle реализует логику игрового раунда.

Метод runRound реализует логику хода в раунде.

Класс TerminalInput реализует логику ввода информации через терминал (консоль/командную строку).

В конструкторе вызывается метод loadCommands. Этот метод пытается сопоставить командам клавиши из файла. Если это не удается, загружается назначение клавиш по умолчанию.

У TerminalInput есть набор различных методов для ввода необходимой информации:

1) Координат

2) Числа

3) Ориентации

4) Ответа да/нет

5) Команды

Шаблонный класс GameRenderer реализует логику отрисовки игры. В качестве параметра шаблона передается класс, отвечающий за то, как будет выводиться информация.

Метод render позволяет отобразить поля всех участников, конкретного участника, помощи с доступными командами и какое-либо сообщение в зависимости от полученных параметров.

Класс TerminalOutput реализует логику вывода информации в терминал (консоль/командную строку).

Набор методов позволяет отобразить всю необходимую информацию об игре в терминал.

Ниже представлена UML-диаграмма реализованных в данной работе классов:

|  |
| --- |
|  |
| Рисунок 1: UML-диаграмма классов |

Разработанный программный код см. в приложении А.

Демонстрацию работы программы см. в приложении Б.

## Выводы

В ходе изучения концепции шаблонных классов в языке программирования C++ была достигнута основная цель — понимание их структуры, применения, а также преимуществ и недостатков. Шаблонные классы являются мощным инструментом, позволяющим создавать обобщенные решения, которые могут работать с различными типами данных без необходимости дублирования кода. Это значительно упрощает разработку и поддержание программного обеспечения.

# Приложение А Исходный код программы

https://github.com/lastikp0/Battleship

.

# Приложение Б Демонстрация работы программы