**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**

Студентка гр. 3341 _____ Кузнецова С.Е.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2024

**Цель работы**

Целью работы является изучение основ объектно-ориентированного программирования, изучение концепции шаблонных классов в языке программирования C++, и их применение, а также создание и использование шаблонных классов для решения различных задач.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Разработать архитектуру классов, отвечающих за управление игрой;
2. Создать классы, отвечающие за отрисовку игры и ввод пользователя из терминала;
3. Связать реализованные классы.

**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 считывании 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**

1) Шаблонный класс GameController – отвечает за управление игрой. Принимает параметры шаблона – классы Input и Displayer – классы, отвечающие за ввод пользователя и за вывод игровой информации на экран.

Хранит ссылку на игровой цикл, map с командами и их ключами и единственные экземпляры классов ввода и вывода (для обеспечения единственной точки доступа для обработчиков).

Конструктор GameController(GameCycle& game) – ставит каждой команде в соответствие функцию из класса GameCycle.

Метод run() – запускает игру и считывает команду, введенную игроком. Если нашлась введённая команда, то вызывается соответствующая функция. После хода игрока идет ход бота, далее вызывается метод для вывода текущего состояния игры. Также производятся проверки условий чьей-либо победы.

2) Класс TerminalInput – отвечает за ввод разных типов информации через терминал. Хранит свой единственный экземпляр, пути к файлам, содержащим команды и ключи к ним, хэш-таблицу с необходимыми для игры ключами. Содержит методы для ввода необходимой информации – публичные методы getString(), getAction(), getFieldSize(), getShipSize() и т.д. и использующиеся в них методы getX(), getXY(), getXYR() для ввода разного количества чисел.

3) Класс TerminalOutput – отвечает за вывод игровых объектов и вспомогательных сообщений на экран. Аналогично, хранит свой единственный экземпляр. Содержащиеся методы для вывода сообщений – abilityMsg(), scannerMsg(), logMsg(), методы для отрисовки игровых объектов – displayPlayerField(), displayBotField(), displayFields(), displayShipManager().

4) Шаблонный класс GameDisplayer – отвечает за логику отрисовки игры. В качестве параметра шаблона передается класс Output, отвечающий за способ вывода информации. Содержит методы display(), displayMsg() для вывода сообщений и отрисовки объектов в классе GameController.

Диаграмма классов, разработанных в ходе выполнения лабораторной работы, представлена на рис.1

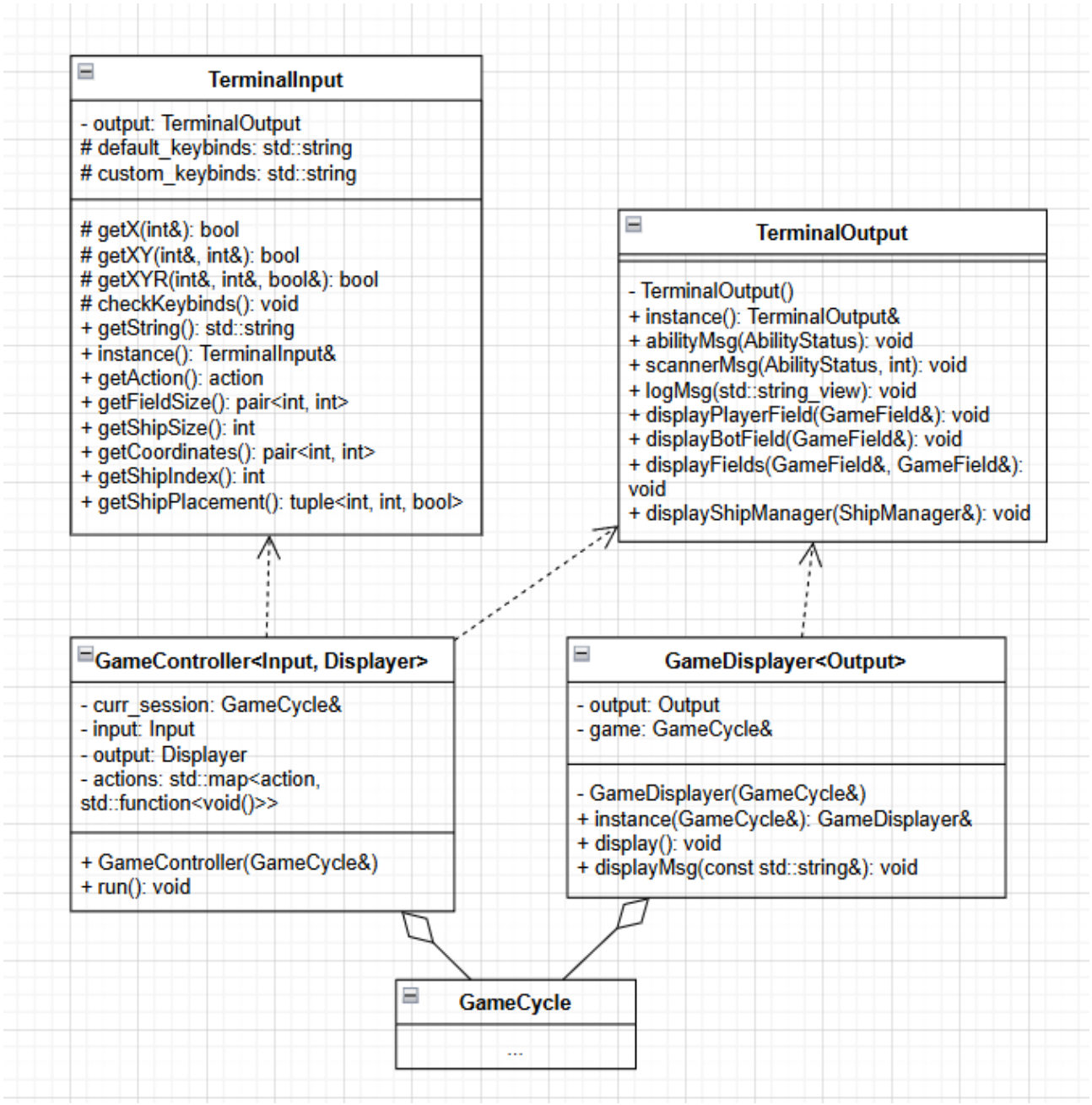


Рисунок 1: UML-диаграмма классов

**Выводы**

В результате выполнения лабораторной работы были изучены основы объектно-ориентированного программирования, изучены концепции шаблонных классов в языке программирования C++ и их применение, а также созданы и использованы шаблонные классы для решения различных задач.

Были выполнены поставленные задачи:

1. Разработана архитектура классов, отвечающих за управление игрой;
2. Созданы классы, отвечающие за отрисовку игры и ввод пользователя из терминала;
3. Связаны реализованные классы.