МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Объектно-ориентированное программирование» Тема: Полиморфизм

Студентка гр. 3388 Сурова Е.Г.

Преподаватель Жангиров Т.Р.

Санкт-Петербург
2024

Цель работы.

Создать систему способностей игрока с использованием классов и механизма наследования. Реализовать три способности (Двойной урон, Сканер, Обстрел) с возможностью их хранения, управления и применения через класс-менеджер. Также реализовать функционал получения случайной способности при уничтожении вражеского корабля, и разработать систему исключений для обработки различных ошибок игрового процесса (попытка применения отсутствующей способности, нарушение правил размещения кораблей, атака за границы поля).

Задание.

Создать класс-интерфейс способности, которую игрок может применять. Через наследование создать 3 разные способности:

Двойной урон - следующая атак при попадании по кораблю нанесет сразу 2 урона (уничтожит сегмент).

Сканер - позволяет проверить участок поля 2x2 клетки и узнать, есть ли там сегмент корабля. Клетки не меняют свой статус.

Обстрел - наносит 1 урон случайному сегменту случайного корабля. Клетки не меняют свой статус.

Создать класс менеджер-способностей. Который хранит очередь способностей, изначально игроку доступно по 1 способности в случайном порядке. Реализовать метод применения способности.

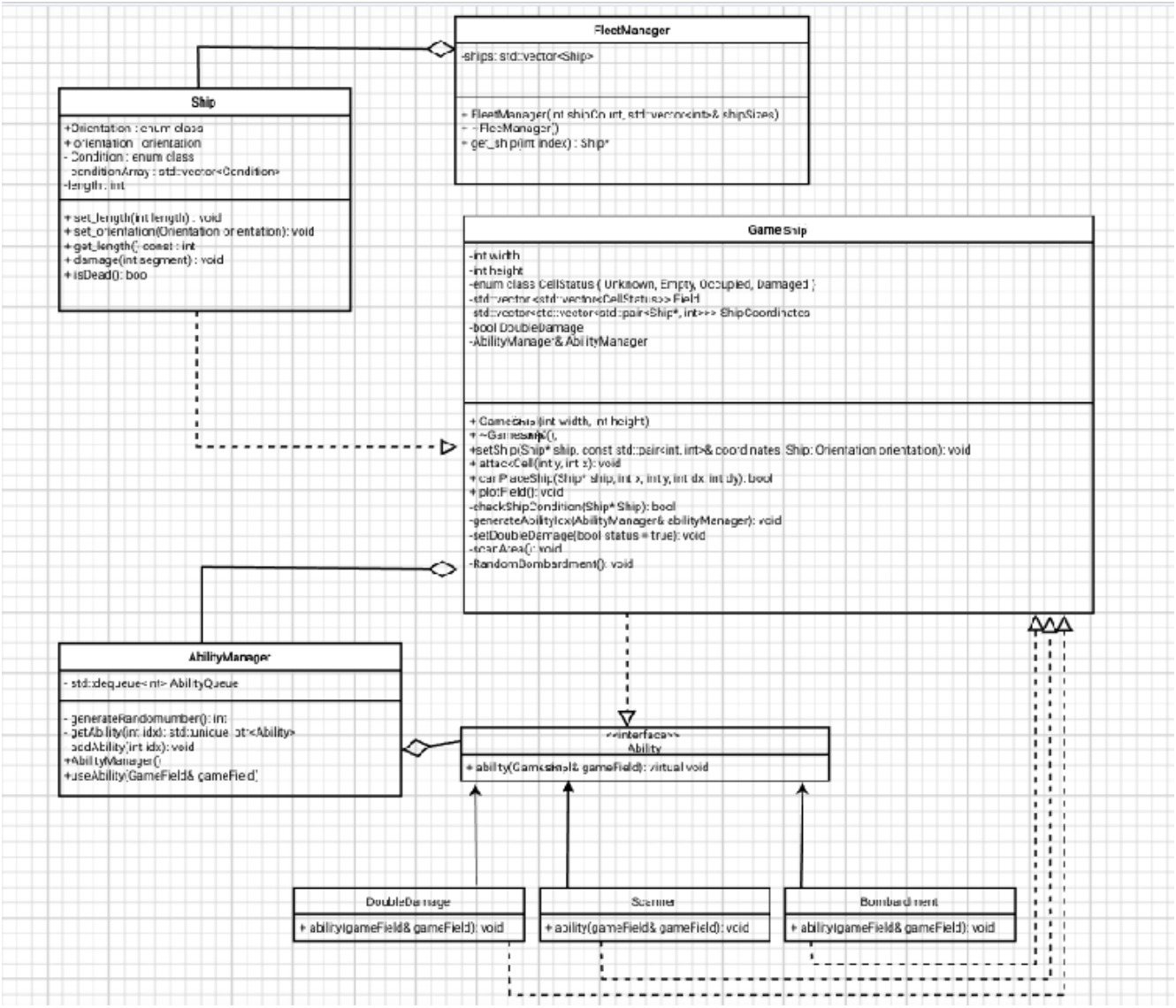
Реализовать функционал получения одной случайной способности при уничтожении вражеского корабля.

Реализуйте набор классов-исключений и их обработку для следующих ситуаций (можно добавить собственные):

Попытка применить способность, когда их нет

Размещение корабля вплотную или на пересечении с другим кораблем Атака за границы поля

Выполнение работы.



UML-диаграмма

Class Ability:

Класс-интерфейс для способностей, которые игрок может использовать. Он обеспечивает единый интерфейс для работы с различными способностями, позволяя использовать их без знания конкретной реализации. Метод virtual void ability является виртуальным, что обязывает все производные классы реализовать его.

Взаимодействие с другими классами:

- **GameShip**: Передается в метод ability, чтобы модифицировать состояние игрового поля согласно логике способности.

Производные классы:

- DoubleDamage
- Scanner
- RandomBombardment

  Эти классы наследуют Ability и реализуют метод ability, определяя поведение соответствующей способности.

Class DoubleDamage:

Способность, которая позволяет следующей атаке нанести двойной урон. При вызове метода apply, активируется режим двойного урона через вызов метода gameField.setDoubleDamage().

Механизм действия:

- Способность не наносит урон напрямую, а изменяет состояние игрового поля.
- Поле doubleDamage (по умолчанию false) добавлено в GameShip.
- Метод setDoubleDamage изменяет это поле на true.
- В методе attackCell проверяется значение doubleDamage, и при значении true наносится дополнительный урон.

Class Scaner:

Способность, позволяющая сканировать участок игрового поля размером 2x2 клетки.

Механизм действия:

- Метод ability запрашивает координаты верхнего левого угла области для сканирования.
- Если координаты выходят за пределы поля, программа автоматически корректирует их в пользу игрока.
- Вызывает метод scanArea у GameShip, передавая координаты.

Изменения в GameShip:

- Добавлен метод scanArea, который проверяет область 2x2.

- Возвращает true, если найдены сегменты корабля, иначе false.

Class Bombardment:

Способность, наносящая один урон случайному сегменту случайного корабля.

Механизм действия:

- Метод ability вызывает метод randomBombardment у GameShip.
- В randomBombardment используется std::rand для выбора случайного корабля и сегмента, которым наносится урон.

GenerateRandomAbilityIdx()

Метод для генерации случайного индекса способности.

- **Механизм действия:**
  - Генерирует индекс способности (0, 1 или 2), соответствующий DoubleDamage, Scanner или RandomBombardment.
  - Преобразует индекс в объект способности через getAbility и возвращает его в виде уникального указателя (std::unique_ptr<Ability>)

Class AbilityManager:

Класс для управления способностями, доступными игроку.

Основные функции:

- **Конструктор:** Создает вектор способностей, перемешивает их и помещает в очередь abilityQueue.
- **addAbilityIdx:** Добавляет новую способность в очередь по её индексу, что экономит память.
- **useAbility:** Применяет первую способность из очереди.
  - Извлекает индекс способности.
  - Преобразует его в указатель на объект через getAbility.
  - Вызывает метод ability и удаляет использованную способность из очереди.

Class AbilityException

Класс для пользовательских исключений, связанных с игровым процессом. Наследуется от std::runtime_error и используется для обработки ошибок, возникающих при управлении способностями, размещении кораблей или атаке.

Выводы.

Был разработан менеджер способностей и набор классов-исключений для обработки различных ошибок в игре. Реализованы три способности: "Двойной урон", "Сканер" и "Обстрел", каждая из которых интегрирована в систему через интерфейс *Ability*. Способности добавляются в случайном порядке и управляются с помощью класса *AbilityManager*. Для обработки ошибок, таких как неправильное размещение кораблей, атака за пределы поля и отсутствие доступных способностей, созданы специализированные классы-исключения. Исключения перехватываются и обрабатываются, позволяя программе продолжать работу без завершения при возникновении ошибок, с выводом соответствующих сообщений для пользователя.