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.

Студент гр. 3344 _____ Коршунов П.И.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2024

Цель работы.

Изучить полиморфизм, реализовав интерфейс для способностей игрока и исключения для обработки ошибок.

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

Примечания:

Интерфейс события должен быть унифицирован, чтобы их можно было единообразно использовать через интерфейс

Не должно быть явных проверок на тип данных

Выполнение работы.

AbilityManager: Менеджер способностей управляет доступными способностями, очередью на их активацию, и настройками параметров.
initializeAbilities(): Заполняет вектор availableAbilities экземплярами конкретных способностей (DoubleDamageAbility, ScannerAbility, RandomFireAbility), каждая из которых принимает игровое поле (GameField) в качестве параметра.
fillInitialQueue(): Случайно перемешивает способности и добавляет их в очередь abilityQueue.
setCurrentAbilityParams: Сохраняет текущие параметры способности для их использования при активации.
getCurrentAbilityParams(): Возвращает название и список требуемых параметров для способности из начала очереди. Выбрасывает исключение NoAbilitiesException, если в очереди нет способностей.
getRandomAbility(): Случайно выбирает способность из availableAbilities и добавляет её в очередь.
activateAbility(): Активирует способность из начала очереди, используя установленные параметры, и удаляет её из очереди.

DoubleDamageAbility: Способность, наносящая двойной урон.
activate(): Дважды атакует одну и ту же ячейку на поле, вызывая attackCell на указанных координатах.
getName(): Возвращает имя "Двойной урон".
requiredParams(): Указывает, что для активации нужны параметры x и y.
setParams(): Устанавливает значения x и y из переданных параметров.

IAbility: Интерфейс, задающий общую структуру для всех способностей. Определяет виртуальные методы activate(), getName(), requiredParams(), и setParams().

RandomFireAbility: Способность, выполняющая случайный обстрел на игровом поле.

activate(): Вызывает метод randomFire() на игровом поле, активируя случайную атаку.
getName(): Возвращает имя "Обстрел".
requiredParams(): Возвращает пустой список, так как параметры не требуются.
setParams(): Оставлен пустым, так как эта способность не нуждается в параметрах.

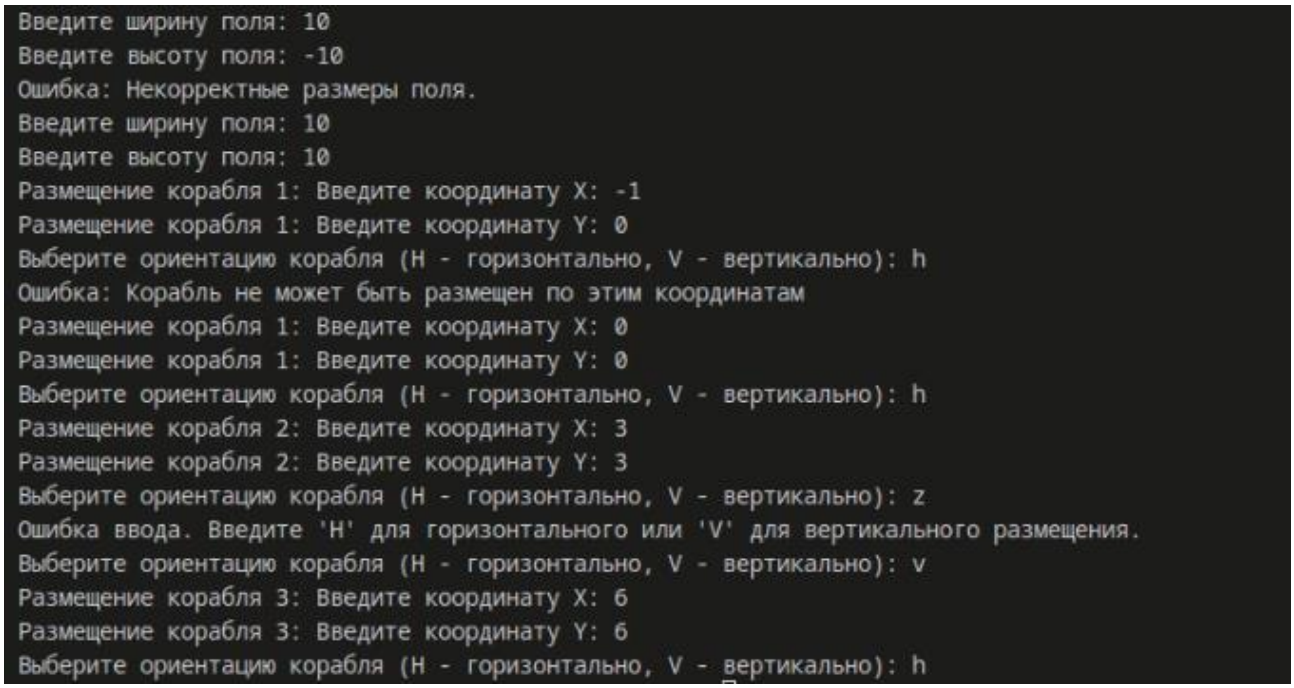
ScannerAbility: Способность, которая сканирует область поля.
activate(): Выводит часть поля в виде окна 3x3, начиная с заданных координат.
getName(): Возвращает имя "Сканер".
requiredParams(): Указывает, что нужны параметры x и y для определения области сканирования.
setParams(): Устанавливает значения x и y из переданных параметров.

UserInput: Класс для ввода данных от пользователя.
getInt: Запрашивает целое число, выводя сообщение об ошибке при некорректном вводе.
getCoordinates: Получает пару координат от пользователя.
getFieldSize: Получает размер поля от пользователя.
getShipOrientation: Запрашивает ориентацию корабля (H для горизонтального и V для вертикального), проверяя корректность ввода.
clearInput: Очищает поток ввода для предотвращения ошибок.
Исключения
AttackOutOfBoundsException: исключение, которое возникает при попытке атаковать за пределами игрового поля. Это исключение информирует пользователя о недопустимой атаке за границами поля.
FieldSizeException: исключение, связанное с некорректными размерами игрового поля, например, если размеры поля заданы неправильно. Это исключение сообщает пользователю об ошибке в конфигурации поля.
NoAbilitiesException: исключение, возникающее, когда у игрока больше нет доступных способностей для активации. При попытке использовать

способность, если очередь способностей пуста, это исключение информирует о недоступности способностей.

ShipPlacementException: исключение используется для обработки ошибок, связанных с размещением кораблей, если они находятся слишком близко или пересекаются с другими кораблями.

Тестирование



Тестирование создания поля с корректными и некорректными размерами, размещение корабля с корректными и некорректными параметрами.

```
Введите координаты для атаки: Введите координату X: -1
Введите координаты для атаки: Введите координату Y: 2
Ошибка: Атака за границами поля.
Введите координаты для атаки: Введите координату X: 0
Введите координаты для атаки: Введите координату Y: 0
Попадание по координатам (0, 0)
Активируем способность: Двойной урон
Введите значение для параметра 'x': 3
Введите значение для параметра 'y': 3
Попадание по координатам (3, 3)
Попадание по координатам (3, 3)
Активируем способность: Обстрел
Активируем способность: Сканер
Введите значение для параметра 'x': 0
Введите значение для параметра 'y': 0
+ S
. .
Ошибка: Нет доступных способностей.
```

Тестирование атаки клетки с корректными и некорректными параметрами. Проверка всех способностей, проверка на доступность способностей.

```
+ S . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
. . . X . . . . . .
. . . S . . . . . .
. . . + . . . . . .
. . . . . S S S S
. . . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .

+ ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? X ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
? ? ? ? ? ? ? ? ? ?
```

Вывод поля в различных режимах, чтобы убедиться, что атаки сработали корректно, и способность обстрел действительно была вызвана.

Выводы.

Создан интерфейс способностей. Это позволяет легко добавлять и управлять различными типами способностей в игре. Разработан менеджер способностей, который управляет коллекцией доступных способностей, обеспечивая их добавление, случайное распределение и активацию. Очередь способностей позволяет управлять порядком их использования, а исключения — обрабатывать ситуации, когда способности отсутствуют.

UML-диаграмма реализованных классов.