**Архитектура проекта**

Имеется класс Game, за счет которого происходит весь процесс игры. В его полях находятся 2 игрока. Основной метод - play(), в нем происходит процесс игры. Игроки по очереди делают золы, пока не будет найден winner с помощью метода getWinner()

**Сущности**

**Ship**

В полях корабля хранятся его размер, позиция его начала, сет из его "живых" клеток, направление и состояние. Начальное состояние - ALIVE.

При попадании в клетку корабля, эта клетка удаляется из сета, состояние сменяется на DAMAGED. Когда сет становится пуст, состояние меняется на DESTROYED.

**Field**

Поле хранит в себе двумерный массив field, массив кораблей, счетчик живых кораблей. Ячейки двумерного массива кодируются следующим образом: field[I][j] = -1, если клетка поля пустая, field[I][j] = -2, если в пустую ячейку поля стреляли, field[I][j] > -1, если эта клетка - клетка корабля, в ячейке лежит номер этого корабля в массиве кораблей.

**OwnField**

Это - некоторая "обертка" над Field самого игрока, чтобы он мог пользоваться своим полем так, как предполагается в игре. В OwnField хранится ссылка на Field данного игрока.

**OpponentsField**

Это - некоторая "обертка" над Field противника, чтобы игрок мог проводить действия с полем противка. В OpponentsField хранится ссылка на Field противника.

**Player**

Игрок хранит в себе свое имя, OwnField и OpponentsField. Имеется метод - initSelf(), в нем игрок вводит свое имя и расставляет корабли на своем поле. Основной метод - makeMove(), с помощью него игрок делает ход. В этом методе игрок выбирает, хочет ли он просмотреть текущее состояние своего поля, поля противника, или желает сразу делать выстрел.

**Обоснование архитектуры**

Хотелось бы, чтобы совершению выстрела игроком, в поле его противника автоматически происходили изменения. Иными словами, хотелось бы не передавать координаты выстрела из одного экземпляра Player в другой. Именно для этого мы "оборачиваем" новым классом OwnField и OpponentsField, которые хранят ссылки на тот участок памяти, что соответствует обычным полям игроков. Таким образом, при завершении выстрела вызывается opponentsField.shootInCell(...) { field->shootInCell(...); }, противник в своем поле автоматически увидет выстрел, его не придется передавать для изменения поля после выстрела.

Так же благодаря данным "оберткам" игроки могут делать только те действия со своим полем и полем противника, которые предполагаются в игре. А именно: игрок может только стрелять в поле противника, узнавать количество живых кораблей только в своем поле. При просмотре своего поля игрок видит свои корабли и выстрелы, которые сделал противник. При просмотре поля противника игрок видит только свои выстрелы.

**Расположение проекта**

Проект находится в директории SeaBattle. В SeaBattle/include находятся заголовочные файлы ship.h, field.h, player.h, game.h, которые хранят объявления соответствующих сущностей. В SeaBattle/src хранятся соответствующие cpp файлы.

В SeaBattle/info хранятся UML диаграммы, сценарии использования и описание архитектуры(данное описание).

**Запуск**

Запускается проект с помощью следующих команд: