|  |  |
| --- | --- |
|  | МИНОБРНАУКИ РОССИИ  федеральное государственное бюджетное образовательное учреждение высшего профессионального образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова» (БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** |
| БГТУ.СМК-Ф-4.2-К5-02 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Факультет |  | О |  | | Естественнонаучный |
|  |  | шифр |  | | наименование |
| Кафедра |  | О7 |  | | Информационные системы и программная инженерия |
|  |  | шифр |  | | наименование |
| Дисциплина |  | | | Программирование на языке высокого уровня | |

КУРСОВАЯ РАБОТА

на тему

|  |
| --- |
| Объектно-ориентированная разработка программ с графическим пользовательским |
| интерфейсом «сверху-вниз»: предварительное выявление классов, объектов и их |
| отношений. Вариант: Морской бой |

|  |  |
| --- | --- |
| Выполнил студент группы | О713Б |
| Деревцов Е.И. |  |
| Фамилия И.О. |  |
| **РУКОВОДИТЕЛЬ** | |
| Вальштейн К.В. | |
| Фамилия И.О. Подпись | |
| Оценка | |
| « » | 20 г. |

**СОДЕРЖАНИЕ**

Оглавление

[Определения, обозначения, сокращения 3](#_Toc124741532)

[Введение 4](#_Toc124741533)

[Постановка задачи 5](#_Toc124741534)

[Описание предметной области 6](#_Toc124741535)

[Описание иерархии классов 7](#_Toc124741536)

[Функциональная модель приложения 11](#_Toc124741537)

[Демонстрация работы 12](#_Toc124741538)

[Заключение 16](#_Toc124741539)

[Список использованных источников 17](#_Toc124741540)

[Приложение А 18](#_Toc124741541)

[Приложение Б 19](#_Toc124741542)

Определения, обозначения, сокращения

В настоящей пояснительной записке применяются следующие сокращения и обозначения:

Разработка «сверху-вниз» - это методика разработки программ, при котором разработка начинается с целей решения проблемы, которые в дальнейшем дополняются и детализируется, тем самым получается завершенная программа.

Десктопное (настольное) приложение – это программа, которую пользователь скачивает на свой компьютер и работает с ней локально на компьютере или удаленно, например, подключившись к серверу.

ООП – объектно-ориентированное программирование.

Введение

Данная курсовая работа демонстрирует навыки и знания, полученные после изучения дисциплины «Программирование на языке высокого уровня» на языке C#.

Из предложенных тем выполняемым вариантом была выбрана игра «Морской бой». Это объясняется тем, что на данный момент существует не так много десктопных приложений, реализующих данную игру.

Целью данной курсовой работы является создание игры «Морской бой» с режимом игры против компьютера.

Для достижения поставленной цели необходимо выполнить следующие задачи:

* уточнение функциональных требований (при необходимости);
* уточнение функциональной модели (при необходимости);
* проектирование диаграммы классов;
* проектирование интерфейса пользователя;
* программная реализация игры;
* проведение тестирования игры.

В данной работе использовалась разработка программы «сверху вниз».

Постановка задачи

Задачей курсовой работы является разработка десктопного приложения «Морской бой» с использованием Windows Forms.

Назначением разработки является создание продукта, являющегося более качественным и доработанным по сравнению с другими настольными приложениями такого вида.

Целевой аудиторией являются люди, которые хотят провести время в игре «Морской бой», но у них нет возможности сделать это с другим человеком.

# Описание предметной области

Игра морской бой – это игра для двух игроков имеющая множество вариаций и дополнительных правил. В классической вариации морского боя игроки расставляют 1 четырехпалубный, 2 трехпалубных и 3 двухпалубных и 4 однопалубны корабля на поле 10 на 10 клеток таким образом, чтобы они не соприкасались углами и сторонами. Далее игрок делает «выстрел» по полю своего соперника, стараясь попасть во вражеский корабль. Если игрок попал, то он пытается угадать расположение корабля и делает повторный выстрел. В случае промаха ход переходит ко второму игроку.

Игра продолжается до тех пор, пока не будут подбиты корабли одного из игроков.

Существует множество вариантов данной игры, в том числе такие, в которых корабли могут располагаться по диагонали и изгибаться углами. Также можно увеличить количество кораблей и размеры поля, тем самым разнообразив игру.

В данной разработке использовался вариант поля 10 на 10 клеток с 1 четырехпалубным, 2 трехпалубными и 4 двупалубными кораблями.

Что касается выигрышных стратегий в этой игре, то их существует несколько видов. Например, это может быть диагональный алгоритм с выстрелами по диагонали. Также это может быть линейный алгоритм, в этом случае все поле простреливается каждый раз с определенным шагом, после чего шаг уменьшается или увеличивается. Также это может быть случайный алгоритм, при котором каждый раз выбирается случайная ячейка [1].

Однако важно не только иметь стратегию при стрельбе по кораблям противника, также важно расположить большинство своих кораблей по бокам поля, а остальные в центре. За счет этого при «убийстве» корабля противником количество закрашиваемых клеток вокруг корабля будет меньше, что заставит противника сделать большее количество выстрелов по вашему полю [2].

# Описание иерархии классов

Класс ячейки Cell.

Класс наследуется от класса Button из библиотеки графических элементов Windows Forms и представляет собой кнопку с дополнительными свойствами и полями.

Входные и выходные данные для класса Cell приведены в таблице 1.

Таблица 1 − Входные и выходные данные для класса Cell

|  |  |  |
| --- | --- | --- |
| Функция | Входные данные | Выходные данные |
| Создание граничной неактивной ячейки. | Координата по Х, координата по У, текст внутри ячейки. | Новая неактивная ячейка.  Возвращаемое значение отсутствует. |
| Смена цвета ячейки. | Выбор цвета, первый исходный цвет, второй исходный цвет. | Изменение цвета ячейки  Возвращаемое функцией значение отсутствует. |
| Проверка на то, является ли ячейка пустой. | Является ли ячейка пустой или нет. | Cмена цвета ячейки в зависимости от того, является ли она пустой или нет.  Возвращаемое функцией значение отсутствует. |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| Функция | Входные данные | Выходные данные |
| Проверка на то, есть ли попадание в корабль или нет. | Является ли ячейка кораблем или пустой клеткой при выстреле. | Cмена цвета ячейки в зависимости от того, есть ли попадание в корабль или нет  Возвращаемое функцией значение отсутствует. |
| Проверка на то, является ли ячейка кораблем. | Является ли ячейка кораблем или нет. | Cмена цвета ячейки в зависимости от того, является ли она кораблем или нет.  Возвращаемое функцией значение отсутствует. |

Класс массива ячеек CellArray.

Класс наследуется от созданного класса Cell и представляет собой массив ячеек, образующий игровое поле.

Входные и выходные данные для класса CellArray приведены в таблице 2.

Таблица 2 − Входные и выходные данные для класса CellArray

|  |  |  |
| --- | --- | --- |
| Функция | Входные данные | Выходные данные |
| Получение элемента из массива ячеек. | Массив ячеек, координаты по осям Х и У | Возвращается элемент массива ячеек. |

Продолжение таблицы 2.

|  |  |  |
| --- | --- | --- |
| Функция | Входные данные | Выходные данные |
| Повторное закрашивание граничных ячеек после уничтожения корабля. | Массив ячеек, сдвиг по координате Х. | Измененный цвет границ поля.  Возвращаемое значение отсутствует. |
| Сокрытие кораблей компьютера. | Массив ячеек, сдвиг по координате Х. | Заполненный массив ячеек для игрового поля. |

Класс игрока Player.

Класс, обозначающий игрока.

Входные и выходные данные для класса Player приведены в таблице 3.

Таблица 3 − Входные и выходные данные для класса Player

|  |  |  |
| --- | --- | --- |
| Функция | Входные данные | Выходные данные |
| Проверка на корректное значение рекорда. | Рекорд игрока. | Проверенное значение, лежащее в пределах от 0 до 100. |

Класс корабля Ship.

Класс, соответствующий кораблю.

Входные и выходные данные для класса Ship приведены в таблице 4.

Таблица 4 − Входные и выходные данные для класса Ship

|  |  |  |
| --- | --- | --- |
| Функция | Входные данные | Выходные данные |
| Проверка потерянного корабля | Массив ячеек. | Если корабль считается убитым, то область вокруг него закрашивается.  Возвращается булево значение. |

Продолжение таблицы 4

|  |  |  |
| --- | --- | --- |
| Добавление корабля | Массив ячеек. | Добавленный корабль, отображающийся среди массива ячеек.  Возвращается булево значение. |
| Удаление корабля | Массив ячеек. | Удаленный корабль среди массива ячеек. Возвращаемое значение отсутствует |

Для построение этой диаграммы использовалась программа Visual Paradigm, имеющая большие возможности для моделирования бизнес-процессов за счёт хорошо написанной документации и наглядных примеров. Диаграмма классов и use-case построены с использованием этого инструмента [3], [4].

Диаграмма классов представлена на рисунке 2 в приложении Б.

# Функциональная модель приложения

IDEF0 – нотация графического моделирования системы на функциональном уровне, отображая структуру, функцию системы и информационные потоки. Правила и примеры построения данной модели представлены в источнике [5].

Функциональная модель приложения в нотации IDEF0 представлена на рисунке 3 и 4 в приложении Б.

Диаграмма вариантов использования(Use-case) представлена на рисунке 1 в приложении Б.

# Демонстрация работы

На рисунке 5.1 представлен вход в игру. На данном этапе пользователь вводит имя игрока, после чего открывается главное меню.

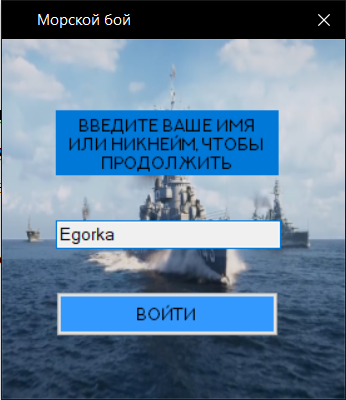


Рисунок 5.1 − Вход в игру

На рисунке 5.2 представлено главное меню, состоящее из следующих пунктов: «Начать игру», «Рекорды», «Выход».



Рисунок 5.2 − Главное меню

На рисунке 5.3 представлен интерфейс начала игры. Поле игрока находится слева, расстановка кораблей выполняется с помощью соответствующих элементов интерфейса.

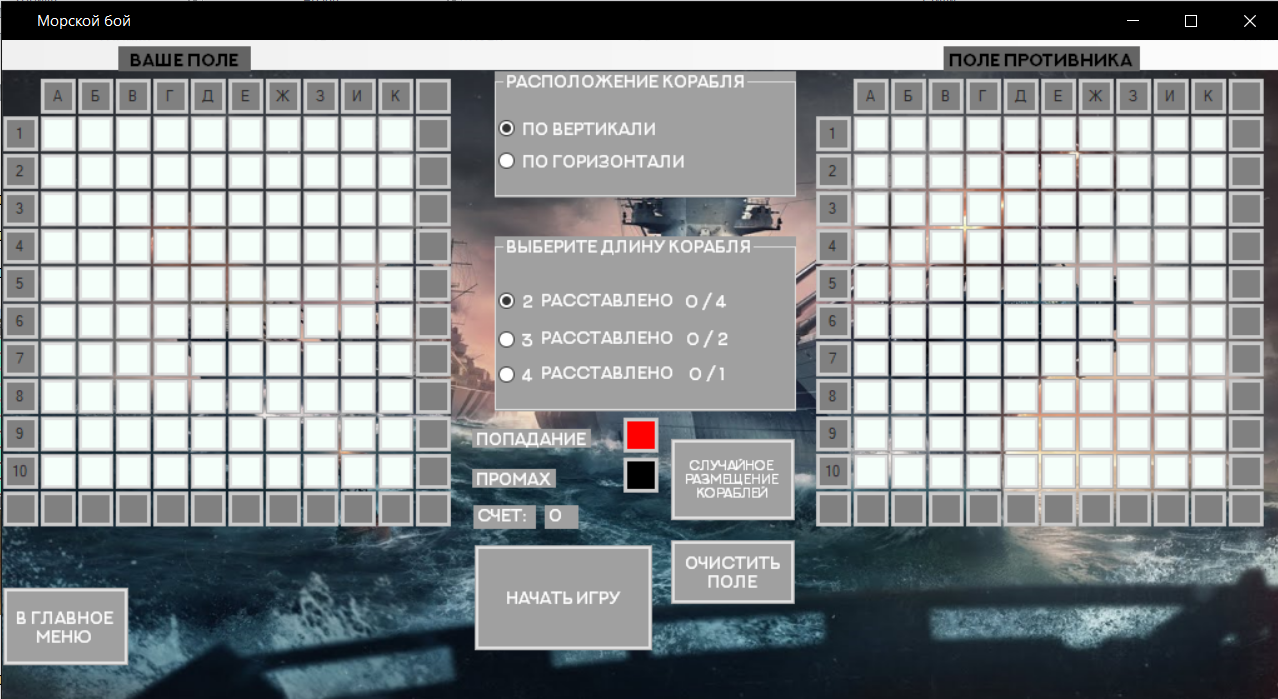


Рисунок 5.3 − Начало игры

На рисунке 5.4 представлено сообщение, выводящееся при попытке установить корабли вплотную.

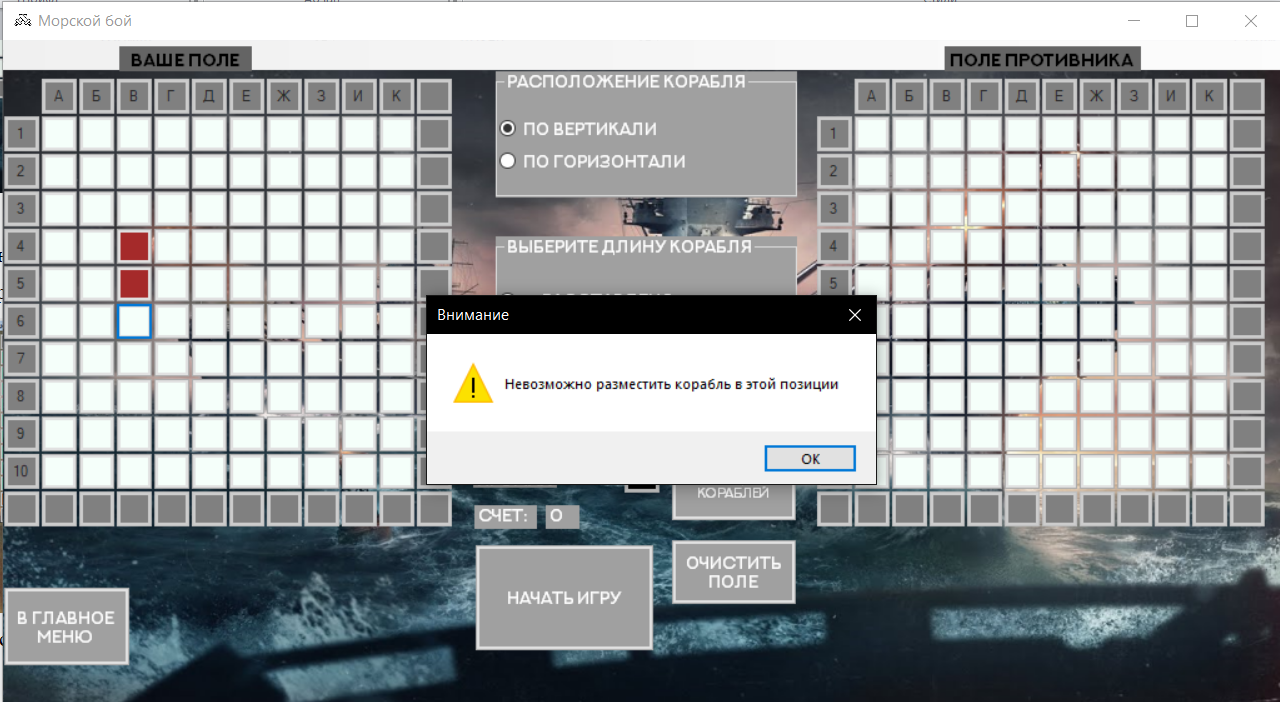


Рисунок 5.4 − Проверка на соприкосновения кораблей

На рисунке 5.5 представлен процесс выстрелов игрока и компьютера.

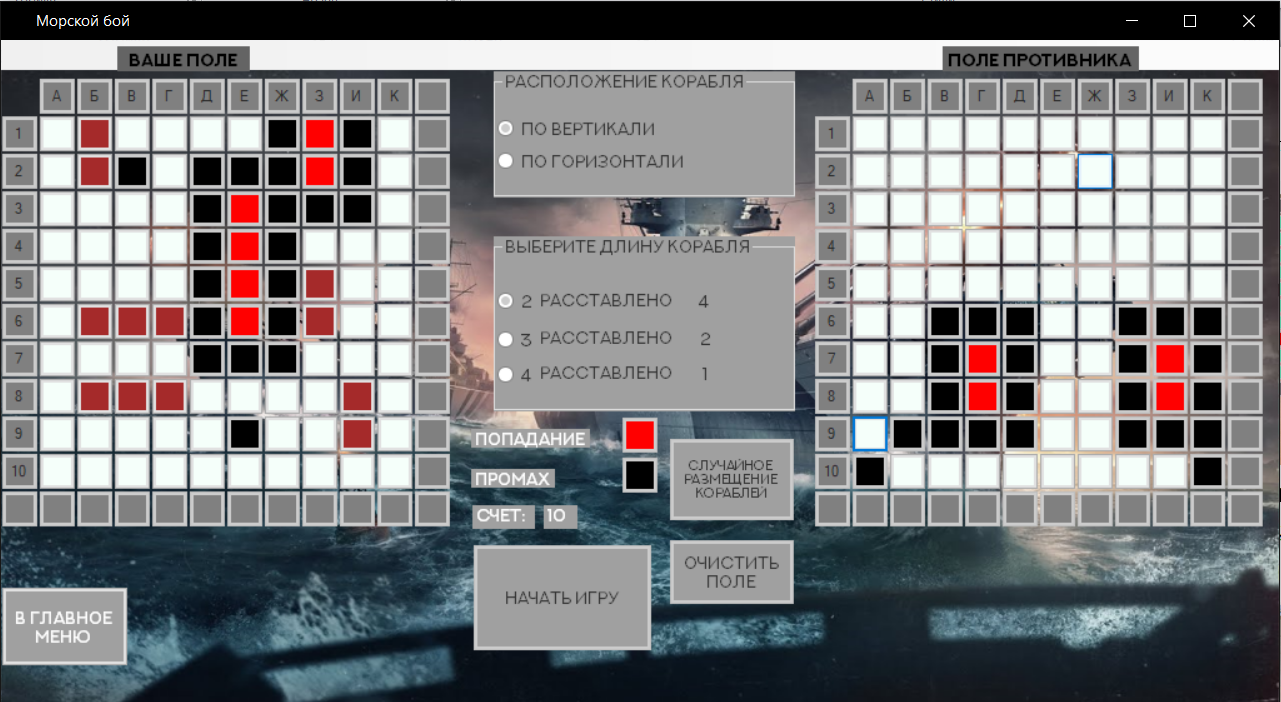


Рисунок 5.5 − Демонстрация игрового процесса

На рисунке 5.6 представлено сообщение, выводящаяся при победе игрока над компьютером.

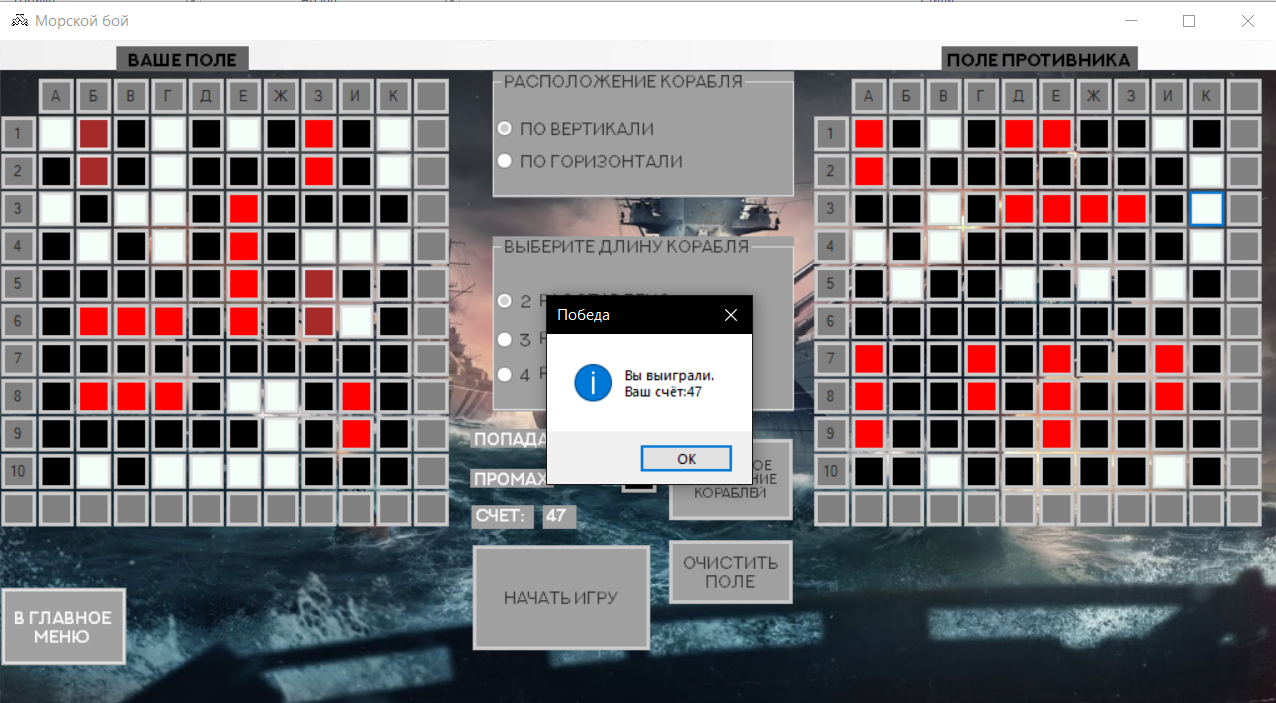


Рисунок 5.6 − Сообщение при победе игрока

На рисунке 5.7 представлена таблица рекордов. Таблица загружается из файла и содержит имя и счет игроков, которые выиграли против компьютера.

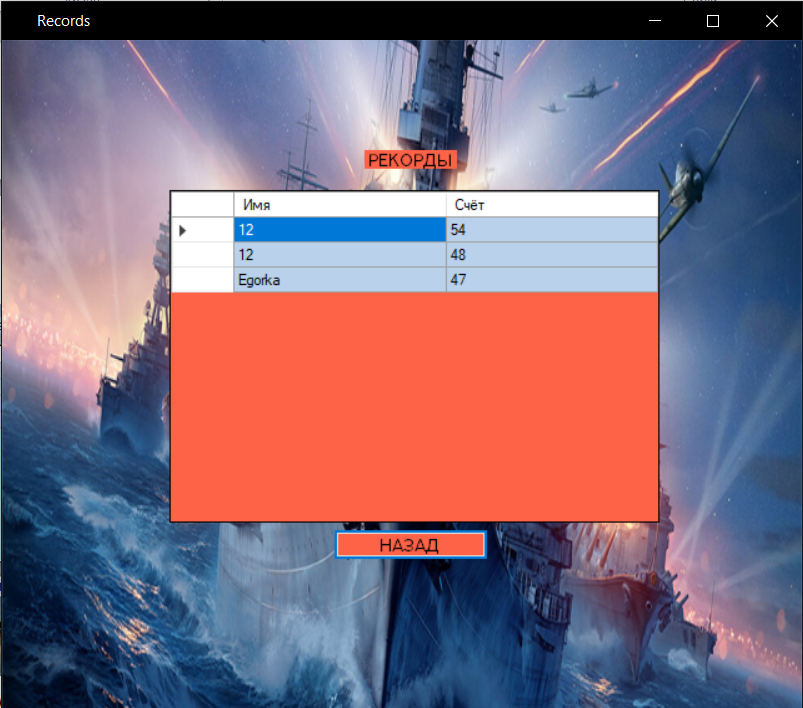


Рисунок 5.7 − Таблица рекордов

# Заключение

В результате выполнения данной работы была создана игра «Морской бой» согласно функциональным моделям с использованием принципов ООП.

Были решены следующие задачи:

* изучение предметной области;
* формулирование требований к разработке;
* проектирование и реализация модели игры на разных уровнях представления;
* реализация программной части;
* реализация пользовательского интерфейса;
* тестирование программы;
* подготовка отчётной документации.

# Список использованных источников

1. Статический анализ стратегий игры морской бой [Электронный ресурс]−URL: <https://cyberleninka.ru/article/n/statisticheskiy-analiz-strategiy-igry-morskoy-boy/viewer>
2. Как правильно играть в морской бой [Электронный ресурс]−URL: <https://cyberleninka.ru/article/n/kak-pravilno-igrat-v-morskoy-boy/viewer>.
3. What is Use Case Diagram? [Электронный ресурс]−URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>
4. UML Class Diagram Tutorial [Электронный ресурс]−URL: [<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/)
5. Нотация IDEF0 [Электронный ресурс]−URL: [<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/)

# Приложение А

Исходные тексты программы располагаются в архиве загруженном по ссылке:

# Приложение Б

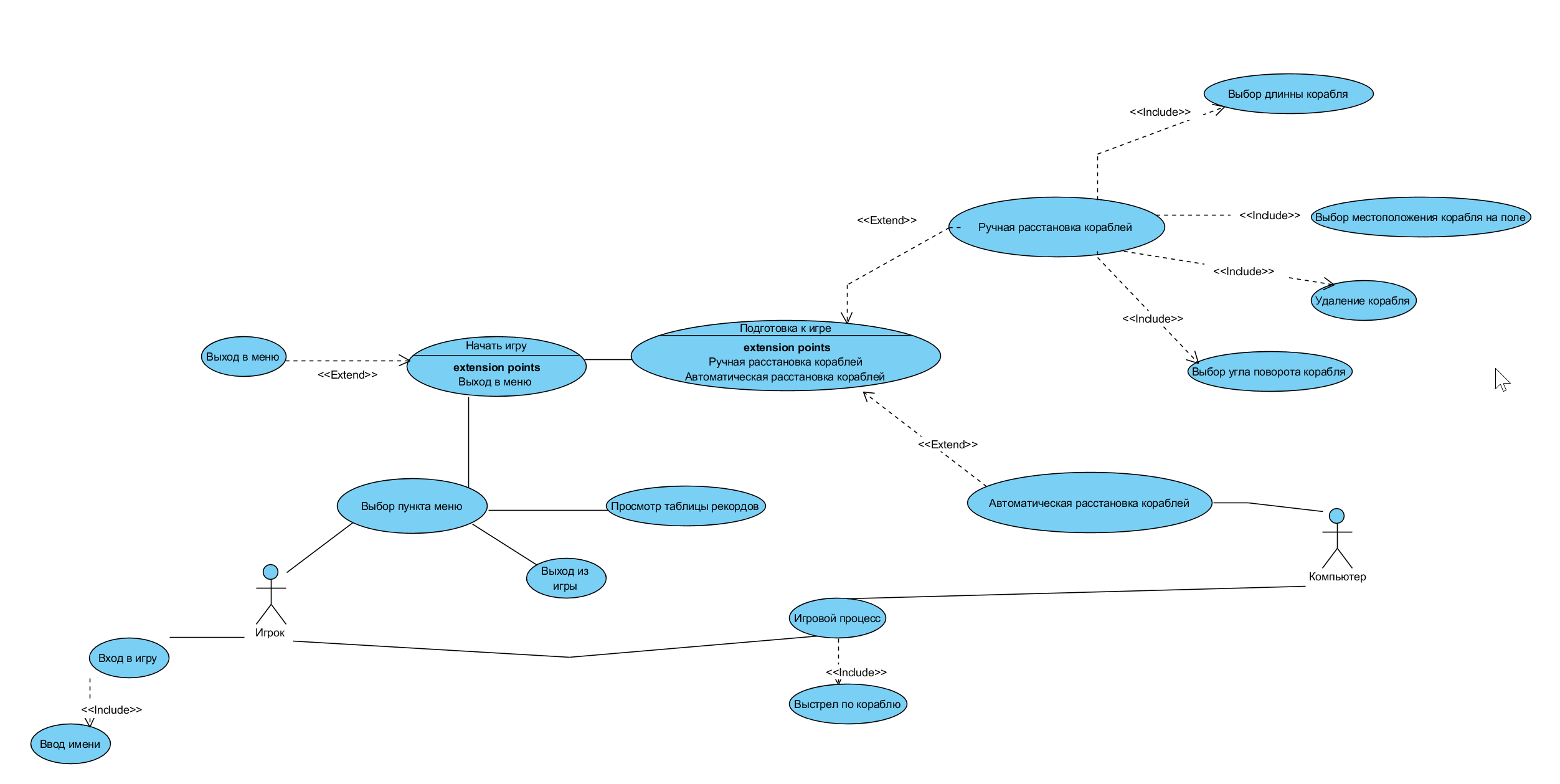


Рисунок 1 – Диаграмма вариантов использования

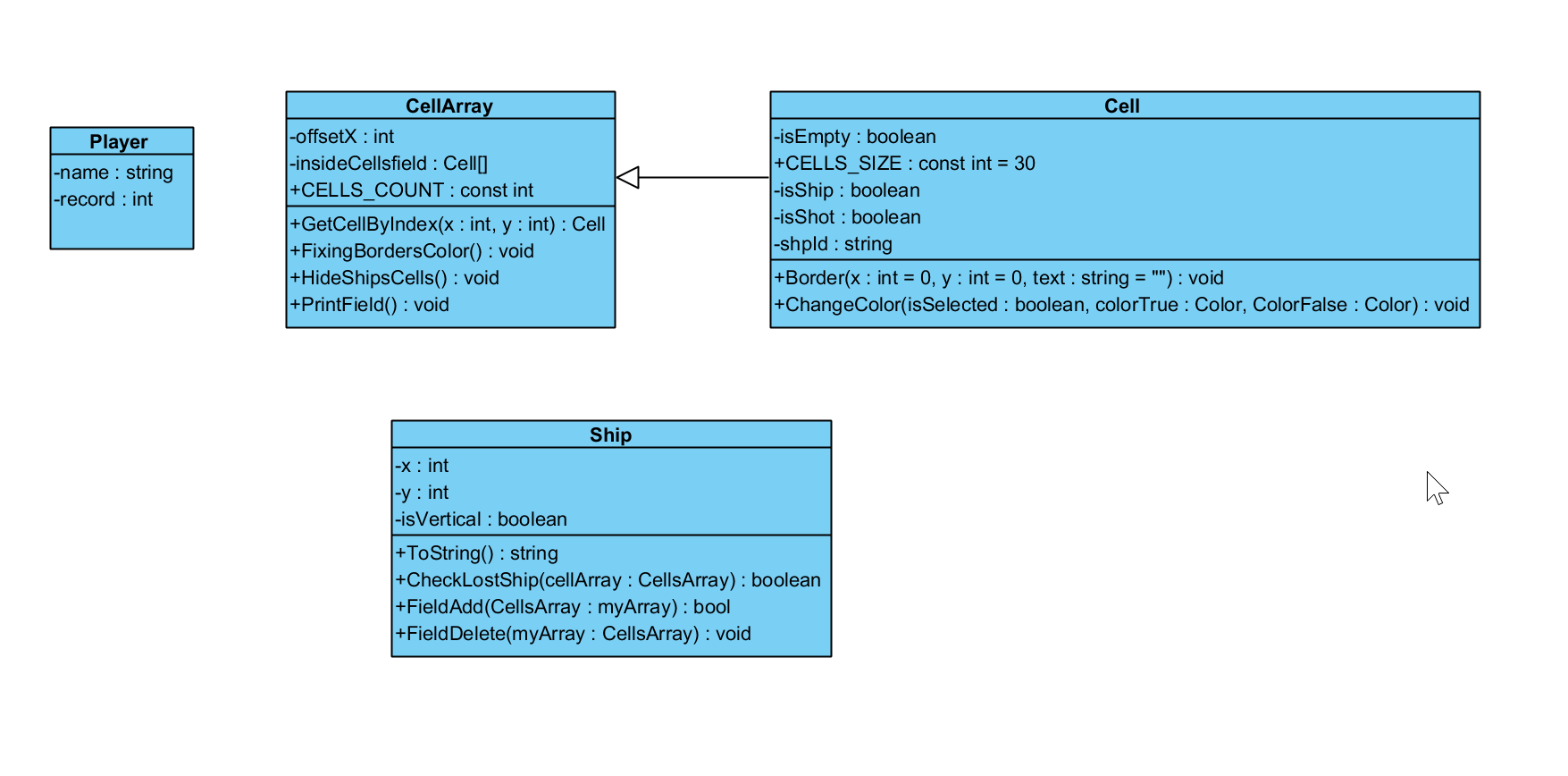


Рисунок 2 – Диаграмма классов

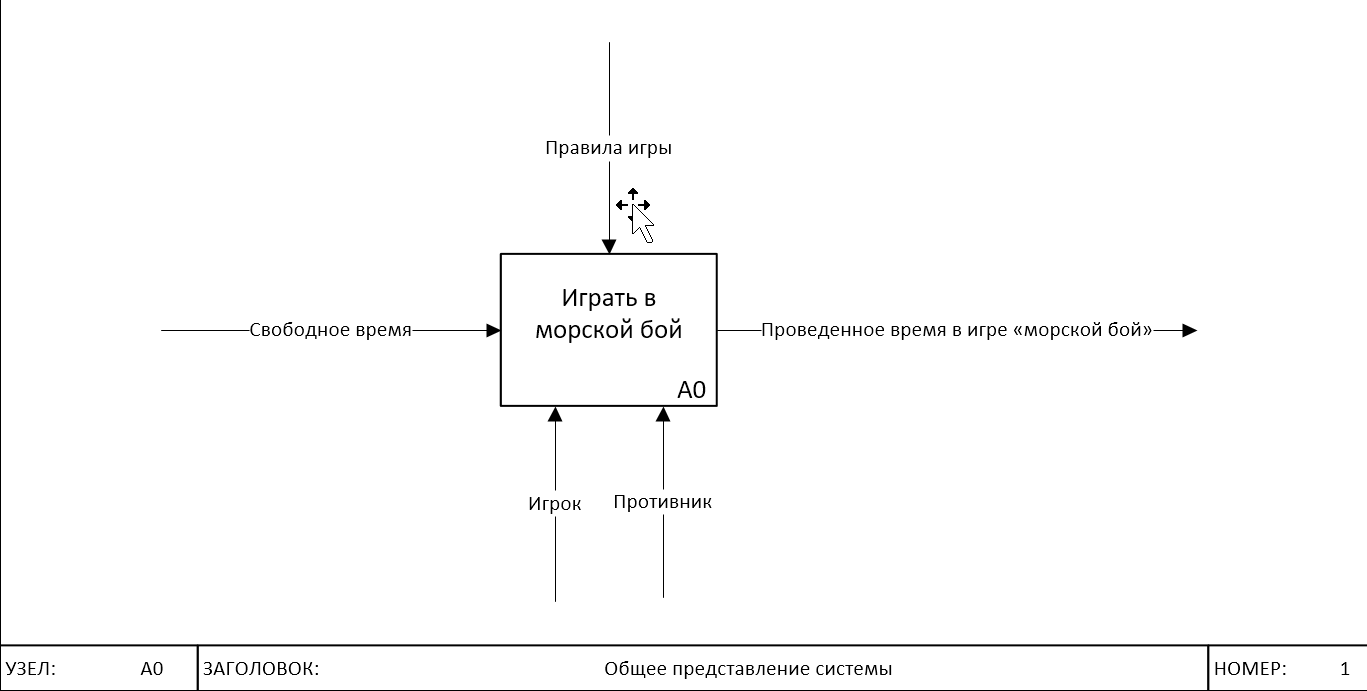


Рисунок 3 – Диаграмма IDEF0 на уровне представления А0

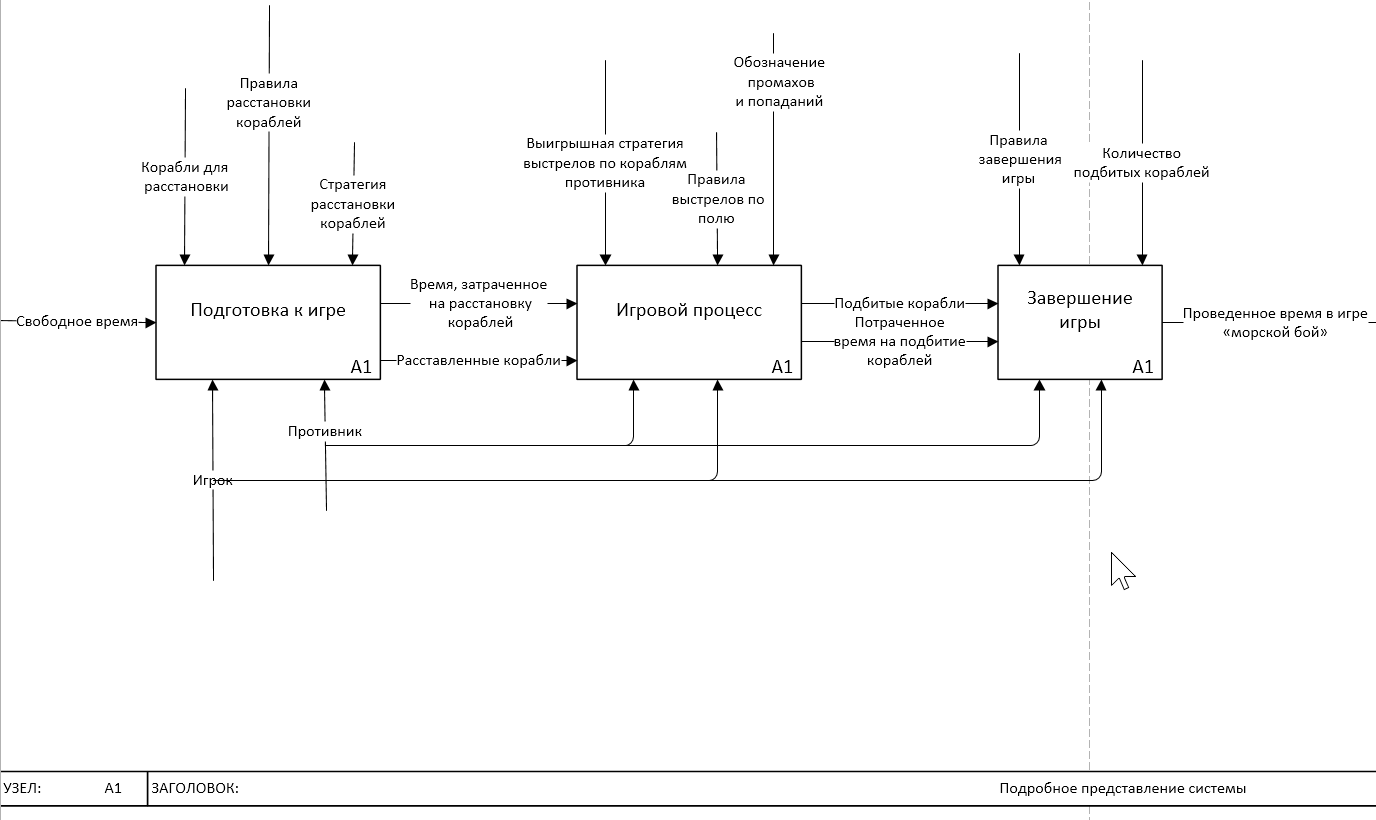


Рисунок 4 – Диаграмма IDEF0 на уровне представления А1