Санкт-Петербургский государственный политехнический университет

Факультет технической кибернетики

Курсовая работа по курсу «Программирование»

Тема работы: «Разработка приложения с графическим интерфейсом на языке C++ с использованием библиотеки QT»

Работу выполнил:

студент группы 1081/3

Арсёнов Юрий

Работу принял:

Пышкин Е. В.

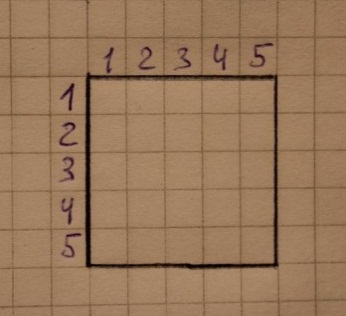
Санкт - Петербург 2012

## Образ и границы проекта

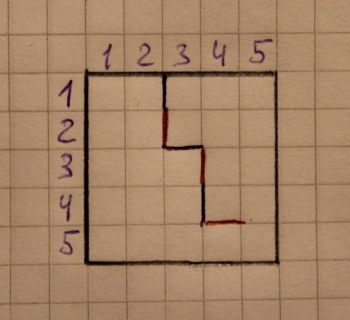
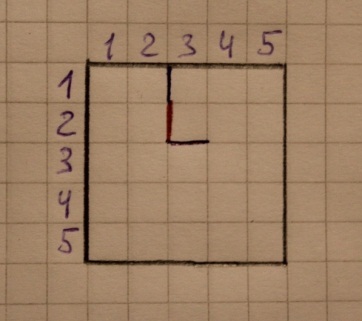
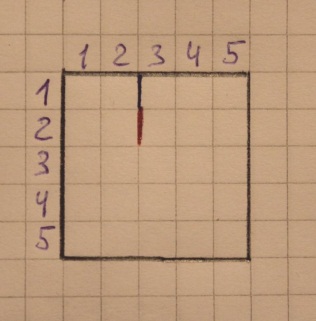
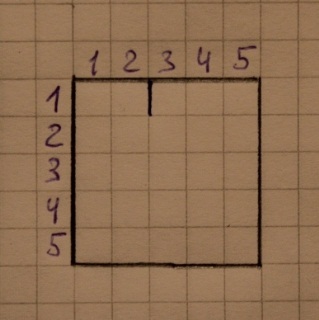
Для реализации в этом семестре, с учетом вашего замечания по поводу статичности приложения, я предлагаю следующую игру( её названия я не знаю).

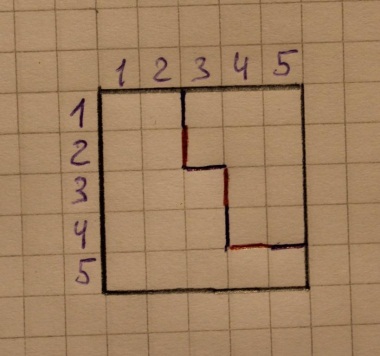
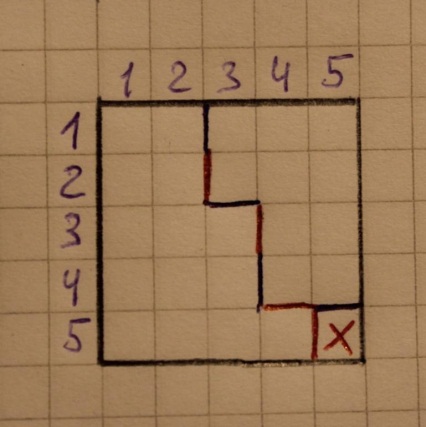
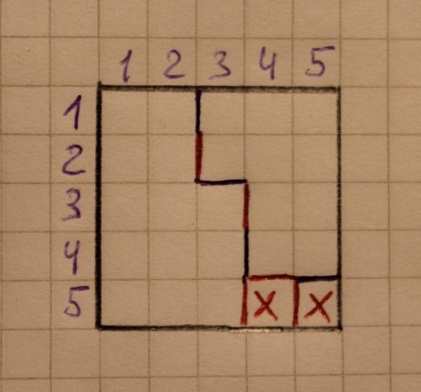
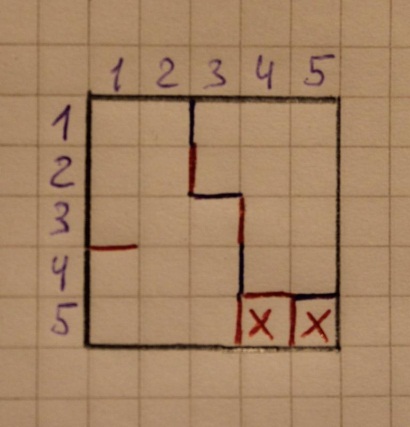
А) На поле чертится квадрат размером 10\*10 или любого другого размера. Так же для игры подойдет практически любая геометрическая фигура, но в данной реализации возможно обойтись квадратом.

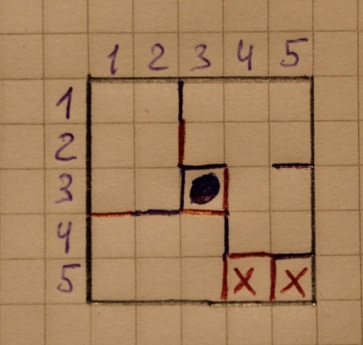
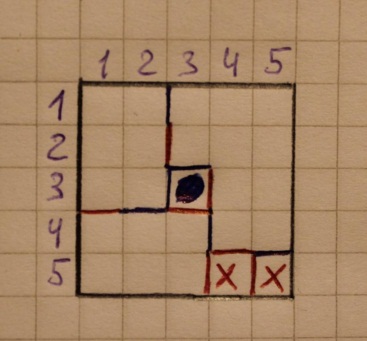
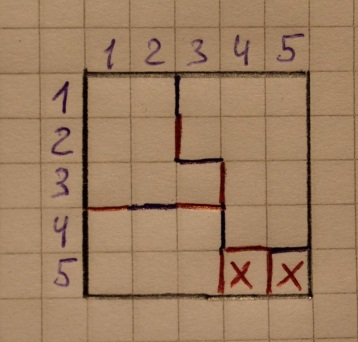
Б) Поле расчерчивается клетками, при данной реализации клеток будет 100, так как размер поля 10\*10. (В примере 5\*5)



В) В игре 2 игрока, каждый из них по очереди ставит на поле черточки по стороне одной клетки, н/р вот так: (первый игрок – синими, второй игрок – красными)



Г) Черточки можно ставить в любом порядке, но когда кто-то из игроков поставит сою черточку так:, что ход последующего игрока может захватить целую клетку, тоэтот игрок ходит и ставаит в захваченную клетку крестик. После этого следующий ход принадлежит этому же игроку, у него есть шанс ещё раз захватить клетку, уже расположенную слева от него, что он и делает. Больше игроку нечего захватывать, но ход принадлежит ему, следовательно он ставит свою черточку в любое свободное место:.

Далее повторяестя то же самое, но в этом случае повезло синему игроку, он и захватывает свободное поле:

Д)Игра продолжается до тех пор, пока не будут заняты все свободные клетки, после этого ведется подсчет количества захваченных клеток. Победил тот, у кого клеток больше.

## Техническое задание

Разработать реализацию игры, представленной в пункте 1, с использованием библиотеки Qt на языке C++.

Игра будет разработана в 2 этапа:

1. Консольное приложение без графического интерфейса.
2. Интеграция кода программы для консоли с графической библиотекой Qt, и собственно создание этого графического интерфейса.

### Основные требования

Главное меню приложения:

1) Начать новую игру.

2) Продолжить игру.

3) Посмотреть статистику игроков

4) Сменить тему оформления.

5) Выйти из игры.

### Основные сценарии работы приложения

Команды меню связаны с реализацией следующих сценариев работы:

### 1)Начало игры

А)Предложение из списка игроков выбрать 2 игрока или создать новых, если не найдется подходящего игрока или захочется создать других.

Б)Когда будут выбраны 2 игрока начнется игра, где каждый из пользователей по очереди будет ходить согласно установленным правилам в игре.

В)В конце, когда поле заполнится, то начнется подсчет очков, затем очки каждого из игроков занесутся в XML файл( структура XML файла определена в дополнении к документу в конце). Для всех игроков планируется создать один XML файл. Так же в режиме игры будет выполнена функция сохранения текущей игры, для выхода и последующего восстановления игры: Для этого будет создан отдельный XML файл, куда будут сохраняться текущие положения элементов на поле и имена игроков.

### 2)Продолжение игры.

В случае, если она была раньше сохранена.

A) Продолжение игры с теми же игроками, что и были с сохраненными данными на поле. Сохраненные данные загружаются из соответствующего XML файла.

Б) Конец игры такой же, как и в подпункте В пункта 1.

### 3)Статистика каждого игрока.

Определяется в процентном соотношении относительно количества побед и сыгранных игр, на основе тех данных которые будут сохраняться в XML файл. После нажатия на этот пункт меню на экран выведится сначала список всех игроков, а затем предложится выбрать интерисующего игрока.

### 4)Смена темы оформления.

Смена фона приложения и стиля кнопок в меню.

### 5)Выход их игры.

### Требования к консольному интерфейсу

1. Отображение пунктов стартового меню с его подпунктами.
2. Полная прорисовка поля во время игры + функции сохранения.
3. Возможность добаления палочек и черточек по координатам.
4. Отображение подсчета очков в игре.

### Требования к графическому интерфейсу

1)Начальный экран с пунктами меню в виде кнопок.

2)Экран игры с разлинованным игровым полем с возможностью с помощью мышки ставить черточки и крестики с ноликами по очереди каждому игроку.

3)Остальные пункты меню кроме игрового аналогичны главному пункту меню.

### 3. Системная архитектура

### Классы, консольного приложения

1)Игровое поле: матрица игровых клеток, с присутвующей в ней методами: создать поле(конструктор), сделать ход черточкой, сделать ход крестиком или ноликом, проверка победителя, очистка поля и т.д.

2)Игровая клетка с параметрами: Пустая, крестик, Нолик, Верхняя граница, Правая граница и т.д.

3)Каждый для меню программы.

4)Класс игра будет управлять поочередно ходами игроков и вести подсчет очков

5)Класс игрок, для каждого отдельно взятого игрока, позволяющий делать ходы по полю.

### Классы графического интерфейса

В связи с отсутствием достаточных знаний в этой области программирования, этот проект будет доработан после представления готового рабочего консольного приложения.

## Дополнение(Структура XML файла):

### 1)XML файл, хранящий информацию о игроках и их статистике.

**<player id="1">** <!—-Контейнер с данными об определенном игроке, на номер игрока указывает атрибут тэга id-->  
**<games>34</games>** <!—-Количество сыгранных игр --> **<wins>10</wins>** <!—-Количество выигранных игр --> **</player>**

**<player id="2">  
<games>34</games>  
<wins>10</wins>**

**</player>**

### 2)XML файл, хранящий информацию о сохраненной игре.

**<set>**

**<palyer1>Ivan</player1> <!-—Имя первого игрока, он всегда ходит ”X”-->**

**<player2>Petr</player2> <!-—Имя второго игрока, он всегда ходит “O”-->**

**<turn>player1</turn> <!—-Определение очереди игрока-->**

**<cel\_1\_1> <!-—Координата ячейки на поле-->**

**<up>yes</up> <!—Присутствие верхней грани-->**

**<down>no</down> <!-— Присутствие нижней грани -->**

**<left>no</left> <!-— Присутствие левой грани -->**

**<right>yes</right> <!-— Присутствие правой грани -->**

**<figure>cross</figure> <!—заполнение ячейки: О, Х, Пусто-->**

**</cel\_1\_1>**

**<cel\_1\_2>**

**<up>yes</up>**

**<down>no</down>**

**<left>no</left>**

**<right>yes</right>**

**<figure>empty</figure>**

**</cel\_1\_2>**

**<cel\_1\_3>**

**<up>yes</up>**

**<down>no</down>**

**<left>no</left>**

**<right>yes</right>**

**<figure>cross</figure>**

**</cel\_1\_4>**

**\***

**\***

**\***

**\***

**\***

**<cel\_10\_10>**

**<up>yes</up>**

**<down>no</down>**

**<left>no</left>**

**<right>yes</right>**

**<figure>cross</figure>**

**</cel\_10\_10>**

**</set>**