Агентство по образованию Российской Федерации

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**Логическая игра “Крестики-нолики” на неограниченном поле**

Пояснительная записка к курсовой работе по дисциплине

"Объектно-ориентированное программирование"

Выполнил:

студент гр.587-1

Бевз О.Д.

\_\_.12.2009 г.

Руководитель проекта

доцент каф КСУП

Коцубинский В.П.

\_\_.12.2009 г.

2009

РЕФЕРАТ

Курсовой проект 17 с., 2 табл., 4 рис.

Пояснительная записка содержит проектную документацию программной системы “Крестики-нолики” на неограниченном поле. Программная система спроектирована в программе Microsoft Visio 2008. Программа реализована в интегрированной среде разработки «Microsoft Visual Studio 2008» на языке Visual C#. Пояснительная записка составлена в текстовом редакторе Microsoft Word 2007.

Федеральное агентство по образованию РФ

ТОМСКИЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ

(ТУСУР)

Кафедра систем управления и проектирования (КСУП)

Утверждаю Разработчик

Заведующий кафедрой КСУП Студент группы 587-1

\_\_\_\_\_\_\_\_\_Ю.А.Шурыгин \_\_\_\_\_\_\_\_\_О.Д.Бевз

“\_\_\_\_”\_\_\_\_\_\_\_2009г. “\_\_\_\_”\_\_\_\_\_\_\_2009г.

ЗАДАНИЕ

на курсовую работу по предмету

"Объектно-ориентированное программирование"

Логическая игра “Крестики-нолики” на неограниченном поле

Согласовано

Аспирант кафедры КСУП

\_\_\_\_\_\_\_\_\_ А.С. Барышников

“\_\_\_\_”\_\_\_\_\_\_\_2009г.

Томск 2009

1.Название системы: Логическая игра “Крестики-нолики” на неограниченном поле .

2.Описание: Приложение является реализацией известной логической игры “Крестики-нолики”. В данной реализации предусматривается игра двух игроков на неограниченном поле.

Цель игры : построить непрерывную линию из пяти или более фишек (крестиков или ноликов) по горизонтали, вертикали или диагонали.

Возможны два режима игры, когда выигрывает тот, кто первым построит линию, или кто больше наберет очков за определенное количество времени. Во втором случае количество построенных линий не ограничено, каждая фишка в линии приносит игроку одно очко. Приложение должно обеспечивать начало новой игры на чистом поле, а также проверку соответствия действий игроков правилам игры и условия окончания игры. Роль одного из игроков (по выбору пользователя) может выполнять компьютер.

Для исключения возможности образования изолированных игр на одном поле и неоправданного увеличения размера поля следует установить максимально допустимое расстояние (не более пяти) от уже существующих фишек до новой, размещаемой игроком.

3.Разработчик:

студент гр.587-1 ФВС ТУСУР Бевз Олег Денисович.

4.Заказчик:

Заведующий кафедрой КСУП Шурыгин Юрий Алексеевич.

5.Система создана на основе предложенного задания документов:

Разработка программы производится на основе полученных знаний для курсовой работы.

6. Содержание пояснительной записки:

‑ анализ задания;

‑ система тестов;

‑ реализация;

‑ тестирование;

‑ заключение;

‑ список литературы.

7. Дата выдачи задания: 24.09.2009

Руководитель доцент каф КСУП

Коцубинский В.П.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Задание принял к исполнению

Бевз О.Д.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Содержание**

|  |  |
| --- | --- |
| 1 Введение |  |
| 2 Анализ задания |  |
| 2.1 Входные данные системы |  |
| 2.2 Выходные данные системы |  |
|  |  |
|  |  |
|  |  |
| 3 Реализация |  |
| 3.1 Проектирование интерфейса пользователя |  |
| 3.2 Проектирование модулей |  |
| 3.4 Проектирование классов |  |
|  |  |
|  |  |
| 4 Тестирование |  |
| 4.1 Тестирование пользовательского интерфейса |  |
| 4.2 Тестирование разборщика командной строки |  |
| 4.3 Тестирование виртуальной машины |  |
| 4.4 Тестирование стека вещественных чисел |  |
| 5 Заключение |  |
| 6 Список литературы |  |

**1.Введение**

Разработанная графическая игра «Крестики - Нолики» обладает следующими достоинствами:

1.Приятный и простой пользовательский интерфейс.

2.Неограниченность игрового поля. Пользователю не придется каждый раз перед началом игры задавать размеры поля.

3.Возможность игры на очки.

4.Возможность игры с компьютером

5.Возможность игры двух компьютеров

6.Программа занимает мало места на жестком диске и обладает низкими системными требованиями.

Также программа обладает следующими недостатками:

1.Возможно наличие мелких ошибок.

2.Исскуственный интеллект не является непобедимым, что ,с другой точки зрения, делает игру привлекательнее.

**2.Анализ задания**

**2.1.Входные данные системы**

Разрабатываемой игре должны поступать следующие данные от пользователя:

* Введется ли игра с компьютером или со вторым игроком;
* Вариант победы в игре:

Вариант№1. Выигрывает игрок, первый составивший в ряд пять крестиков (или ноликов).

Вариант№2. Выигрывает игрок, набравший большее количество очков за определенное время. За составленные в ряд 2 крестика (нолика) игрок получает 2 очка, за 3 крестика (нолика) - 3 очка и так далее.

- Координаты на поле, где игрок хочет поставить свою фишку.

**2.2.Выходные данные системы**

Разрабатываемая система должна выдавать пользователю следующие данные:

- Начало игры с чистым полем.

- При Варианте победы №2 количество набранных очков, каждым игроком.

- Сообщение о победе «крестиков» или «ноликов».

**3.Реализация.**

**3.1.Проектирование интерфейса пользователя.**

Был спроектирован пользовательский интерфейс, представленный на рисунках 1 и 2,3.

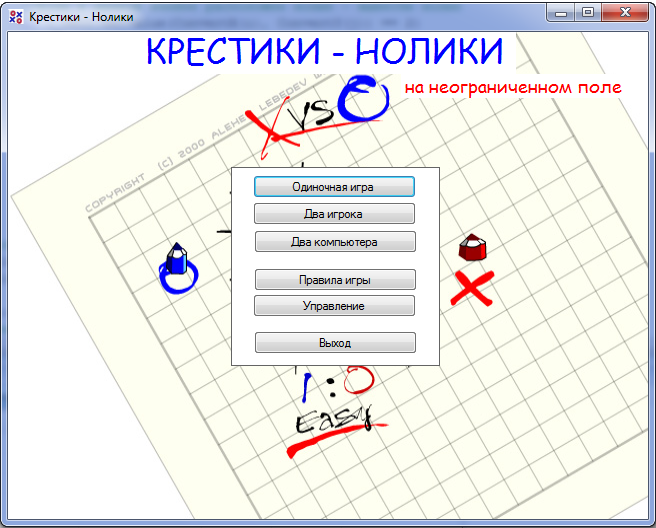
****

Рисунок 1 – Спроектированный пользовательский интерфейс главного меню.

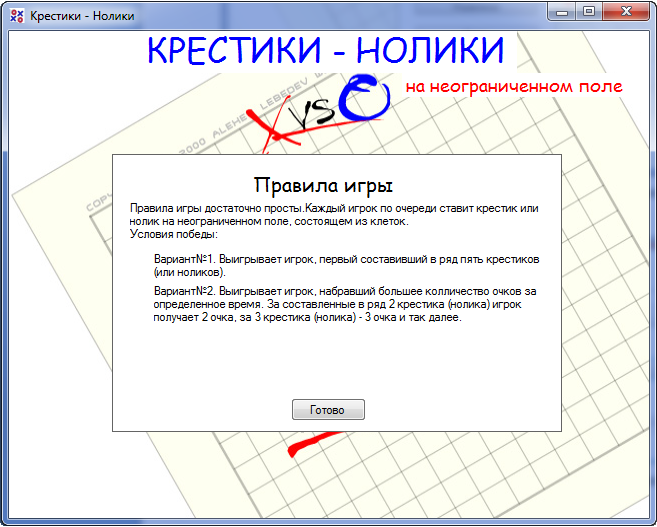


Рисунок 2 – Спроектированный пользовательский интерфейс правил игры.

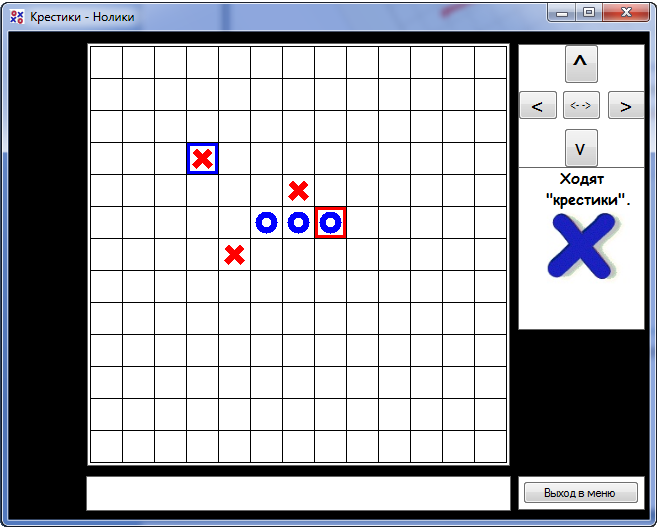


Рисунок 3 – Спроектированный пользовательский интерфейс игры.

При запуске игры пользователь попадает в главное меню. При нажатии на кнопку «Правила игры и Управление» на экране появляется панель, при помощи которой пользователь может ознакомиться с правилами игры, а также выбрать вариант победы. Нажав на кнопку «Готово» пользователь возвращается в главное меню. Кнопка «Выход» отвечает за выход из программы. Кнопки «Одиночная игра» и «Два игрока» запускают игру с компьютером или игроком.

В центре формы находится игровое поле, разбитое на десять клеток. При желании пользователь может перемещаться по полю в любом направлении при помощи стрелок в верхнем правом углу окна. В верхнем левом углу окна показывается картинка «крестика» или «нолика», указывающая на то, кто сейчас ходит. Под игровым поле расположена панель, на которой отображаются различные сообщения (сообщение о неправильном выборе места фишки и т.д.).

**3.2.Проектирование классов.**

Диаграмма взаимодействия классов, представлена на рисунке 1.5.

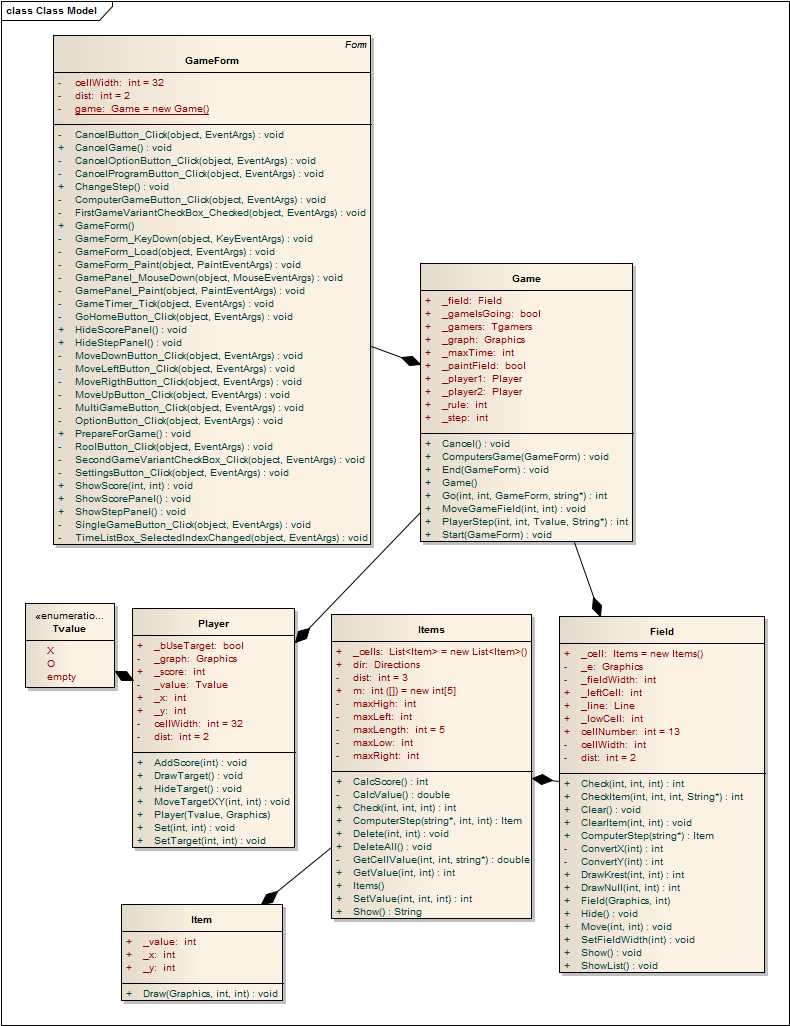


Рисунок 4 – Диаграмма взаимодействия классов

На данной таблице 1 отображены нижеследующие классы:

Таблица 1.1 Описание классов

|  |  |  |  |
| --- | --- | --- | --- |
| Название класса | Описание класса | Поля и методы класса | Описание полей и методов класса |
| GameForm | Форма программы. Содержит в себе все элементы интерфейса. | Game game  void ShowScorePanel()  void ShowScore(int score1, int score2)  void HideScorePanel()  void ShowStepPanel()  void ChangeStep()  void HideStepPanel()  void PrepareForGame()  void CancelGame() | Класс игры  Отобразить панель очков игроков  Отобразить очки игроков  Спрятать панель очков игроков  Отобразить картинку ходящего игрока  Сменить картинку ходящего игрока  Спрятать картинку ходящего игрока  Подготовить форму к игре (спрятать меню, отобразить игровое поле и пр.)  Выйти из игры (спрятать игровое поле, отобразить меню) |
| Game | Основной класс программы, логически объединяющий все остальные классы в один блок. Также содержит такие параметры игры как вариант победы, лимит времени, и переменную, указывающую, кто сейчас ходит. | int \_step  int \_rule  int \_maxTime  Field \_field  Player \_player1  Player \_player2  bool \_gameIsGoing;  bool \_paintField;  Game()  void Start(Form1 form1)  voi**d MoveGameField(int dx,**int dy)  void СomputersGame(GameForm form1)  int PlayerStep(int X, int Y, Tvalue value,ref String Message)  void Go(int X,int Y,Form1 form1)  void End(Form1 form1)  void Cancel() | Номер ходящего игрока  Вариант правил игры  Максимальное время игры  Игровое поле  Игрок№1  Игрок№2  Идет игра или нет  Отображать поле или нет  Конструктор  Начало игры    Переместить игровое поле  Игра двух компьютеров  Ход одного из игроков  (в графическом плане)  Игровой ход (в плане игры, использует int PlayerStep)  Конец игры (вывод сообщения, блокировка поля)  Выход из игры (блокировка и скрытие поля) |
| Player | класс, представляющий игрока, прорисовку указателя игрока. Также содержит в себе количество набранных очков (вариант победы №2). | int \_x  int \_y  Tvalue \_value;  int \_score  bool \_bUseTarget;  Player(Tvalue Value, Graphics e)  void SetTargetXY(int x, int y)  void MoveTargetXY(int dx, int dy  void DrawTarget()  void SetTarget(int X,int Y)  void AddScore(int Score)  void HideTarget() | Координаты последней поставленной фишки  Фишка игрока  Очки игрока  Отрисовывать указатель или нет  Конструктор  Установить координаты  Передвинуть координаты  Нарисовать указатель  Установить координаты  Прибавить очки  Спрятать указатель |
| Field | класс, отвечающий за все процедуры связанные с игровым полем (кроме прорисовки прямоугольника выбора игрока).Данный класс чертит само поле, рисует на нем крестики и нолики, перемещает поле а также хранит в себе матрицу, в которой хранятся все крестики и нолики на поле. | int \_leftCell  int \_lowCell  Items \_cell  Field()  int ConvertX(int X)  int ConvertY(int Y)  void Show()  void Clear()  int DrawKrest(int x, int y)  int DrawNull(int x, int y)  void DrawLine(int x1, int y1,int x2,int y2)  void ClearItem(int x, int y)  int Check(int x, int y)  int CheckItem(int x, int y,int value, ref String Message)  void Hide()  void Move(int dX, int dY)  void ComputerStep() | Координаты левого нижнего угла, отображаемого поля.  Матрица фишек  Конструктор  Конвертирование координат отображаемого поля в координаты общего поля  Показать поле  Очистить поле  Нарисовать крестик  Нарисовать нолик  Нарисовать линию  Очистить ячейку  Проверка на условие победы  Проверить не занята ли ячейка  Спрятать поле  Передвинуть общее поле  Ход компьютера |
| Items | матрица, в которой хранятся все крестики и нолики на поле. Матрица представляет собой список из структур классов Item, содержащих в себе координаты определенной клетки на поле и его значения (0,1,2). | List<Item> \_cells  directions dir;  int \_index;  int maxLeft, maxRight, maxHigh, maxLow;  int maxLength = 5  Cells()  int SetValue(int x, int y, int value) int CalcScore()  int GetValue(int x, int y)  void Delete(int x, int y)  void DeleteAll()  int Check(int x, int y)  Item ComputerStep(int lowCell)  double CalcValue()  double GetCellValue(int x,int y) | Список фишек  Количество одинаковых фишек в 8 направлениях  Количество фишек  Дозволенные границы для расстановки фишек  Максимальное количество составленных в ряд фишек  Конструктор  Установить фишку  Посчитать очки  Найти величину заданной ячейки  Удалить ячейку  Удалить все ячейки  Проверка на условие победы  Расчет хода компьютера  Расчет цены ячейки для одной стороны  Расчет цены ячейки |
| Item | структура, содержащая данные о каждой поставленной фишке | int \_x;  int \_y;  int \_value; | Координаты ячейки на поле  Значение ячейки |

**3.3.Проектирование модулей.**

В процессе проектирования модулей была построена таблица 1.2 модулей и содержащихся в них классов. Связи между модулями остаются теми же, что и между соответствующими классами.

Таблица 2.Проектирование модулей.

|  |  |
| --- | --- |
| Название модуля | Содержащиеся внутри классы |
| Game.cs | Game |
| Player.cs | Player |
| Items.cs | Items |
| Item.cs | Item |
| Fields.cs | Field |

**4.Тестирование**

Программа прошла через серию следующих тестов:

**4.1.Проверка граничных условий**

Программа адекватно отреагировала на следующие ситуации:

- координаты новой фишки выходили за диапазон целочисленного типа.

- значения границ для расстановки фишек выходили за диапазон целочисленного типа.

**4.2.Тестирование вычислений**

Математической частью программы является функцияItems::ComputerStep.

Функция была проверена с помощью игры с компьютером.

**4.3.Проверка инициализации данных**

Было проверено, обнуляются ли все значения при повторном запуске игры (счет игроков, массив фишек, номер ходящего игрока т.д.)

**4.4.Проверка обработки ошибок**

Программа адекватно отреагировала на следующие ситуации:

- игрок ставил одну фишку поверх другой

- игрок ставил фишку за доступными пределами

**4.5.Проверка обработки или интерпретации данных**

Были проверены данные передаваемые одной функции другой. В частности конвертирование координат отображаемого поля (13\*13) в координаты общего поля.

**4.6.Тестирование пользовательского интерфейса**

Было выяснено, что пользовательский интерфейс удовлетворяет следующим условиям:

- достаточная и неизбыточнаяфункциональность;

- быстрота изучения программы новым пользователем;

- легкость запоминания необходимых приемов работы;

- высокая производительность;

- низкая вероятность возникновения ошибок пользователя;

**4.7.Тестирование при повышенной нагрузке**

Была проверена работа программы в условиях высокой загруженности процессора

**5.Заключение**

В процессе проделанной работы была разработана игра «Крестики - Нолики», поддерживающая как игру с компьютером, так и игру двух компьютеров. Были улучшены навыки программирования, в частности работа с Windows Forms Designer, составление UML диаграмм и анализ задачи.

Дальнейшее развитие данной программы может продвигаться по следующим направлениям:

- добавление новой функциональности

- улучшение структуры классов

- исправление мелких ошибок

- улучшение искусственного интеллекта

**6 Список использованных источников**

1. Троелсен Эндрю. Язык программирования С# 2005 и платформа .NET 2.0, 3-у издание : Пер. с англ. – М. : ООО «И.Д.Вильямс», 2007. – 1168с.
2. Карле Ватсон. С#. Перевод с англ. : Издательство «Лори» , 2005г. - 862с.