**­Федеральное государственное образовательное бюджетное учреждение  
высшего образования**

**«Финансовый университет при Правительстве Российской Федерации»**

**КОЛЛЕДЖ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ**

**ПМ.02 Разработка и администрирование баз данных**

**Группа: 4ПКС-114**

**УТВЕРЖДАЮ**

**Председатель цикловой комиссии**

**программирования и баз данных**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Пестов А.И./**

**\_\_\_\_.\_\_\_\_. 2017**

**ПРОЕКТ КУРСОВОЙ**

**На тему: «**Компьютерная игра «Сапёр» c графическим интерфейсом на языке С#»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**Руководитель курсового проекта**

**\_\_\_\_\_\_\_\_\_\_\_\_\_ /Морозова М.В./**

**Исполнитель курсового проекта**

**\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/**

**Оценка за проект: \_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_.\_\_\_\_.2017**

**2017**

Содержание

Введение 3

1 Общий раздел 4

1.1 Системные требования 4

1.2 Характеристика среды программирования 4

2 Технологический раздел 6

2.1 Предпроектное исследование предметной области 6

2.2 Анализ требований и определение спецификаций программного обеспечения 6

2.3 Проектирование программного обеспечения 10

2.4 Разработка пользовательских интерфейсов программного обеспечения 12

2.5 Тестирование и отладка программного обеспечения 16

3 Руководство по использованию программы 19

3.1 Руководство системного программиста 19

3.2 Руководство программиста 20

3.3 Руководство пользователя 20

Заключение 22

Список использованных источников 23

Приложение А 25

Приложение Б 42

Введение

В данной курсовой работе требуется реализовать логическую игру “Сапёр”. Задачей пользователя в этой игре является поиск всех мин, разбросанных по игровому полю в случайном порядке.

Определять в какой клетке имеется мина можно с помощью логического мышления, анализируя цифры, которые расположены в некоторых клетках. Эти цифры показывают, сколько мин являются соседями данной клетки (соседей у каждой клетки восемь). Однако иногда даже в середине и в конце игры некоторые ячейки всё же приходится открывать наугад. «Заминированные» ячейки игрок может пометить, чтобы случайно не открыть их. Открыв все «не заминированные» ячейки, игрок выигрывает.

Цель данного курсового проекта – досуг пользователя, а также обучение не опытных пользователей управлению мышью в компьютере.

Данная программа будет реализована для операционной системы Windows с использованием технологии .net и языка C#.

# Общий раздел

## Системные требования

Рекомендованные системные требования:

- Выпуск Windows: Windows 7, Корпорация Майкрософт (Microsoft Corporation), 2008г. Либо более новая версия.

Требования к системе:

- Процессор: Intel (R) Core (TM) i3-2130 CPU @ 3.40GHz 3.40 GHz.

- Установленная память (ОЗУ): 8,00 ГБ.

- Тип системы: 64-разрядная операционная система, процессор х64.

- Клавиатура.

- USB-мышь либо наличие сенсорной панели (touchpad).

* 1. Характеристика среды программирования

Данная программа была написана на объектно-ориентированном языке программирования C# с помощью MS Visual Studio 17 Community с множеством различных встроенных библиотек и функций, удобным интерфейсом, наличием отладчика кода.

Программирование выполнялось на языке C#, тип проекта – графическое окно.

C# — язык программирования, сочетающий объектно-ориентированные и контекстно-ориентированные концепции. Разработан в 1998—2001 годах группой инженеров под руководством Андерсa Хейлсбергa в компании Microsoft как основной язык разработки приложений для платформы Microsoft .NET. Компилятор с C# входит в стандартную установку самой .NET, поэтому программы на нём можно создавать и компилировать даже без инструментальных средств вроде Visual Studio.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет строгую статическую типизацию, поддерживает полиморфизм, перегрузку операторов, указатели на функции-члены классов, атрибуты, события, свойства, исключения, комментарии в формате XML. Переняв многое от своих предшественников — языков C++, Delphi, Modula и Smalltalk — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем: так, C# не поддерживает множественное наследование классов (в отличие от C++) или вывода типов (в отличие от Haskell).

1. Технологический раздел
   1. Предпроектное исследование предметной области

Приложение должно представлять классическую логическую игру Сапер. В данной игре, мины будут располагаться на двумерном поле. Количество мин и размеры поля можно редактировать. Программа должна иметь интуитивно понятный эргономичный интерфейс. Должны присутствовать меню, для удобного управления программой, а также для отображения информации об авторе. Настройки программы должны загружаться из файла, и при изменении пользователем, сохраняться.

Функциональные требования:

- Реализация логической игры «Сапер»;

- Возможность начать новую игру.

Управление параметрами, включающими в себя возможность корректировки следующих параметров:

- Высота игрового поля;

- Ширина игрового поля;

- Количество мин.

Нефункциональные требования:

- Работа программы под операционной системой Microsoft Windows;

- Графический интерфейс пользователя, простота в освоении для пользования.

- Отказоустойчивость при вводе неверных данных;

- Игровое поле в виде матрицы размерами MxN.

- Таймер отсчета времени

2.2 Анализ требований и определение спецификаций программного обеспечения

2.2.1 Диаграмма вариантов использования

Диаграмма вариантов использования представлена на рисунке 1, и состоит из действий, совершаемых пользователем в программе.

Пользователь может получить справку об игре, обозначить размер поля или начать игру.



Рисунок 1 – Диаграмма вариантов использования

2.2.2 Функциональная диаграмма

Функциональная диаграмма - диаграмма, отражающая взаимосвязи функций разрабатываемого программного обеспечения

Контекстная функциональная диаграмма представлена на рисунке 2.

На ней можно увидеть, что на вход программе подаются:

- Высота игрового поля;

- Ширина игрового поля;

- Количество мин.

Программой управляет программой пользователь.

На выходе у программы:

- Реализация игры Сапёр.

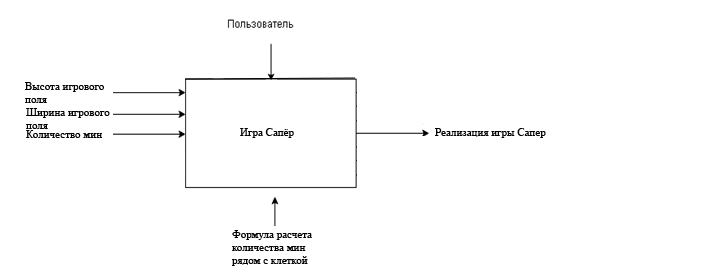


Рисунок 2 – Контекстная функциональная диаграмма

Детализированная функциональная диаграмма представлена на рисунке 3.

Данная диаграмма содержит основные функции программы:

- Вывод меню;

- Формирование поля и мин в зависимости от размера;

- Вывод поля на экран.

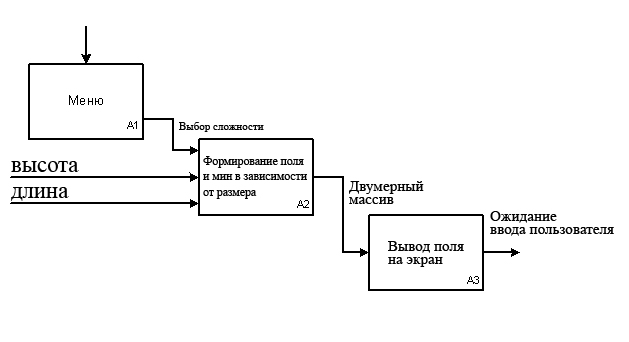


Рисунок 3­­­ – Детализированная функциональная диаграмма

2.2.3 Диаграмма потоков данных

Контекстная диаграмма потоков данных представлена на рисунке 4, и представляет собой взаимодействие пользователя и приложения, связанных потоками данных.

На диаграмме можно увидеть, что от пользователя к программе поступают такие действия, как:

- Выбор пункта меню;

- Выбор клетки;

- Выбор размера поля.

Программа, в свою очередь, выводит пользователю игру “Сапер”, само меню и вывод поля.

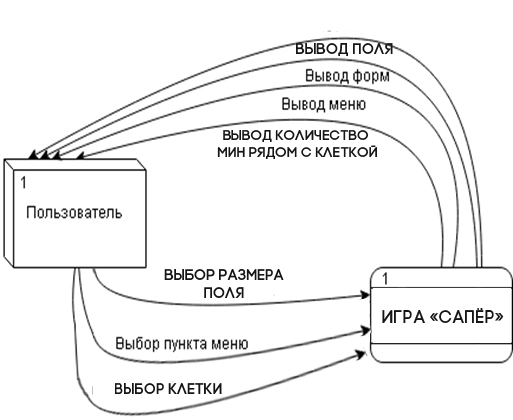


Рисунок 4 – Контекстная диаграмма потоков данных

Детализированная диаграмма потоков данных является более подробной и представлена на рисунке 5.

На ней можно заметить, как при выводе меню в программу пользователь вводит размер поля и количество мин. Далее передаются введенные данные для вычислений, и применяется формула по построению поля для игры. Как итог, выводятся реализация игры “Сапёр”.

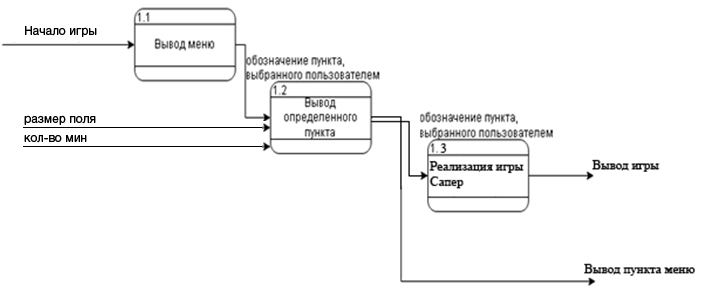


Рисунок 5 – Детализированная диаграмма потоков данных

2.3. Проектирование программного обеспечения.

На рисунке 6 представлена структурная схема программы, а на рисунке 7 представлена функциональная схема. На структурной схеме программы можно заметить основные функциональные части программы. Схема отображает принципы действия программы в самом обобщенном виде. На функциональной схеме представлено вывод и ввод данных отдельных частей ее функций. Например в функцию Выбор сложности вводятся ширина, высота поля и количество мин, так же выводит размер соответствующий с уровнем.

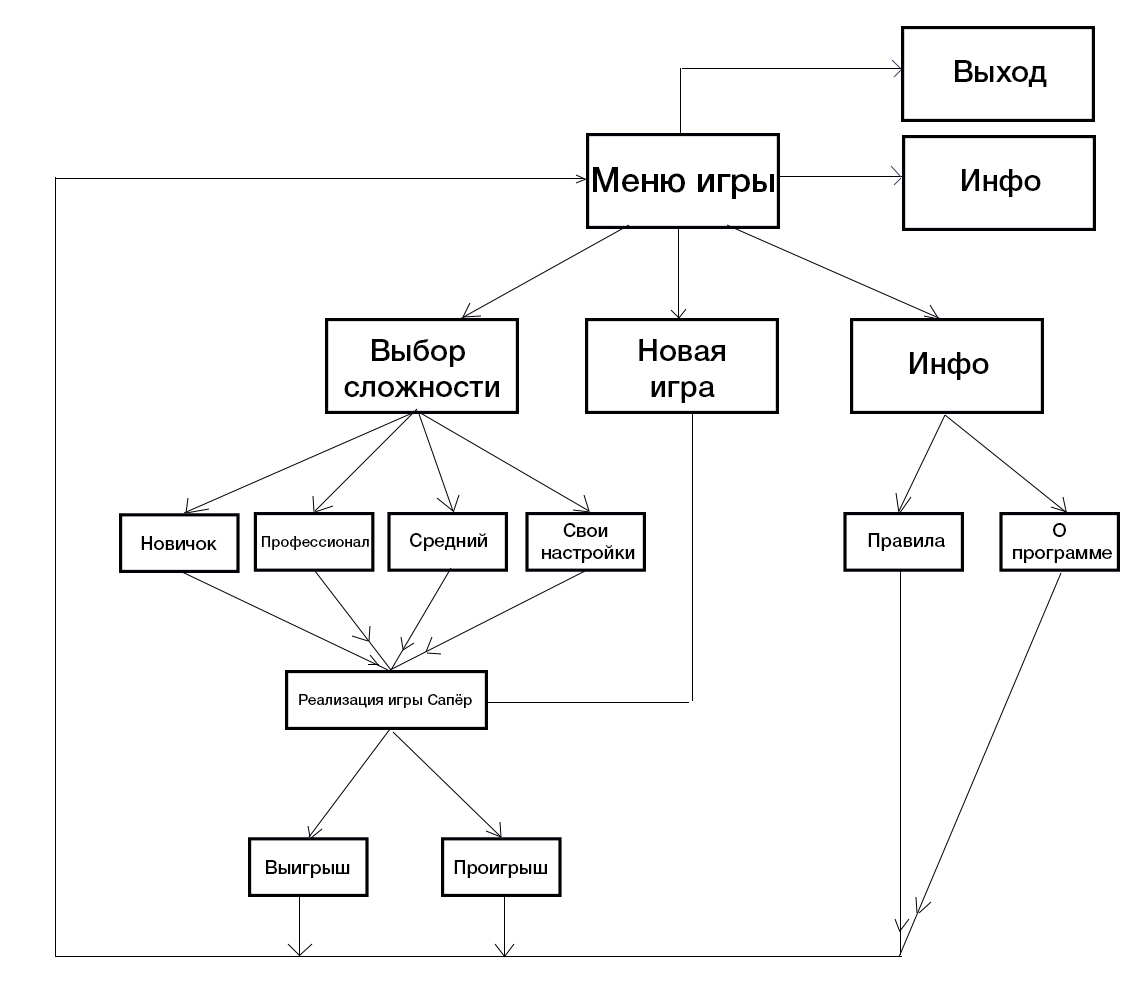


Рисунок 6 – Структурная схема­

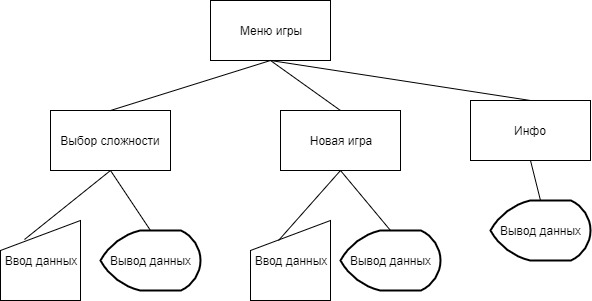


Рисунок 7 – Функциональная схема

2.4. Разработка пользовательских интерфейсов программного обеспечения.

2.4.1 Интерфейс главного меню

Перед пользователем снизу появляется меню с несколькими пунктами, при наведении на которые выводится список пунктов (рисунок 8).

Список имеющихся пунктов меню:

- Новая игра;

- Новичок;

- Средний;

- Профессионал;

- Свои настройки;

- Выход.

Пользователю предоставляется выбор уровня сложности, как представлено на рисунке 8, по умолчанию – Новичок (рисунок 9). Для того чтобы выбрать уровень игры – требуется зайти в подпункт “Игра”, затем выбрать уровень. Уровень “Средний” представлен на рисунке 14, “Профессионал” на рисунке 15. Пользователь может выставить свои параметры игры, размер поля и количество мин, выбрав пункт Свои настройки, как показано на рисунке 8. Данная функция продемонстрирована на рисунке 13.

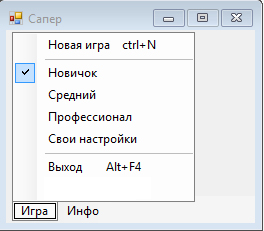


Рисунок 8 – Меню

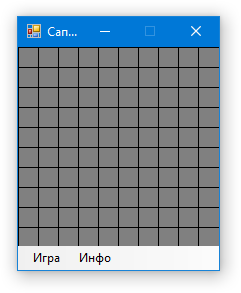


Рисунок 9 – Игровое поле уровня “Новичок”

После выбора предполагаемой клетки, в которой нету мины, игроку открывается клетка или поле, представлено на рисунке 10. При ошибочном выборе клетки – появляется окно “Вы проиграли”, показано на рисунке 11. В случае выигрыша - появляется окно “Вы победили”, показано на рисунке 12.

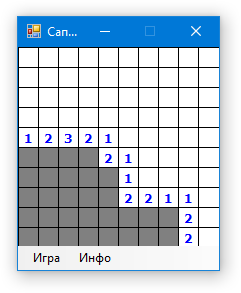


Рисунок 10 – Реализация игры “Сапёр”

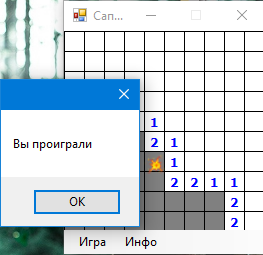


Рисунок 11 – Проигрыш

Для того чтобы ознакомится с правилами, пользователь должен нажать подпункт меню “Инфо”, и затем выбрать “Правила”. Из формы “Правила игры “Сапёр””, как показано на рисунке 16, можно выйти с помощью кнопки “Принимаю”.

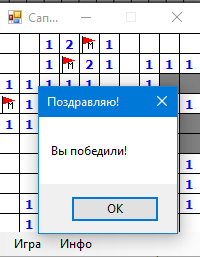


Рисунок 12 – Победа

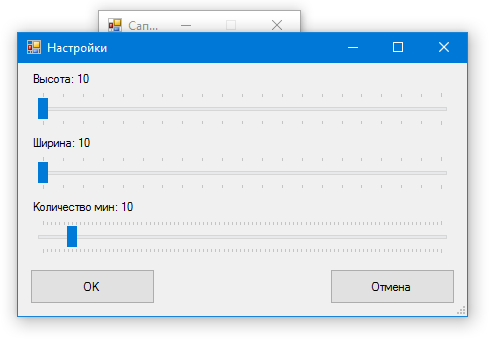


Рисунок 13 – Свои настройки

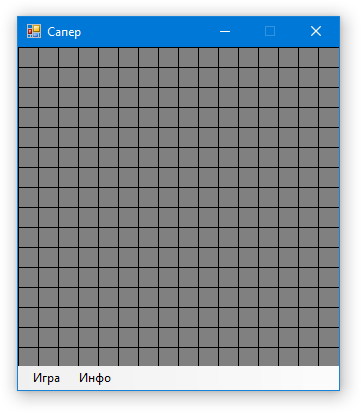


Рисунок 14 – Игровое поле уровня “Средний”

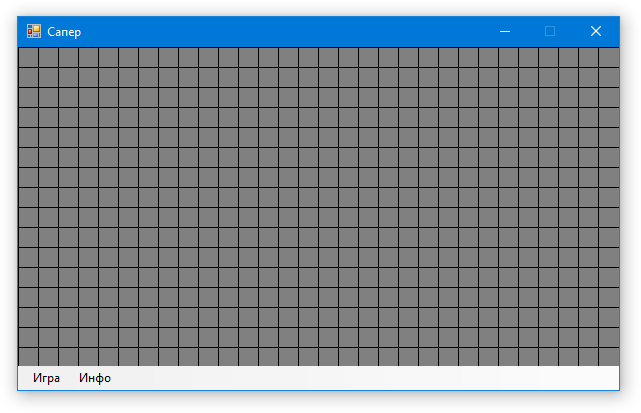


Рисунок 15 – Игровое поле уровня “Профессионал”

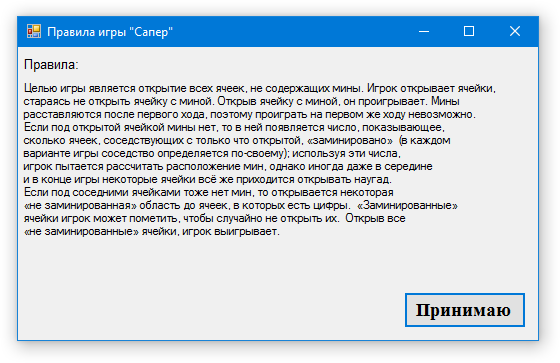


Рисунок 16 – Правила игры “Сапёр”

2.5. Тестирование и отладка программного обеспечения.

Для того чтобы пользователю было удобно, игровое окно невозможно уменьшить и увеличить вручную. Игровое поле меняется исключительно само благодаря заданному размеру поля, и не может как превышать размеры самого поля, так и быть меньше его, представлено на рисунках 9,14,15.

А также количество мин не может быть больше, чем клеток на поле, но может быть равно ему, что исключает выигрыш пользователя. Представлено на рисунке 17.

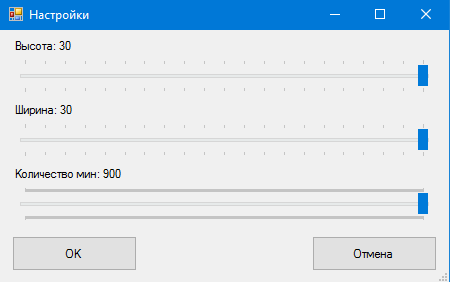


Рисунок 17 – Количество ограничение количество мин

Во избежание экстренного завершения приложения, при нажатии на выход высвечивается предложение завершить игру. Пользователь в праве нажать нет, и продолжить игру. Показано на рисунке 18.

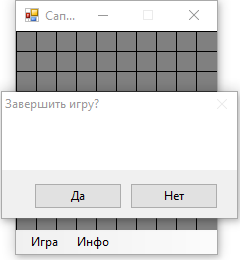


Рисунок 18 – Завершить игру

По окончанию игры у пользователя нету привилегий открывать пустые и заполненные клетки, игровой процесс полностью остановлен, представлено на рисунке 19. Также при игровом процессе пользователь не может повторно открыть клетку. Показано на рисунке 20.

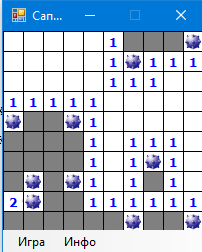


Рисунок 19 – Конец игры

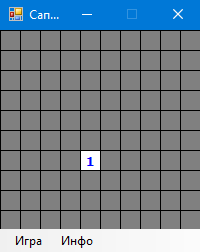


Рисунок 20 – Игровой процесс, клетка

# 3. Руководство по использованию программы

3.1. Руководство системного программиста.

3.1.1 Общие сведения о программе

Программа «Сапёр» является игровым приложением, с помощью которого пользователь имеет возможность испытать или обрести навыки игры в «Сапёр».

Функции, выполняемые программой:

- Реализация игры;

- Выбор сложности;

- Выбор количество мин и размер поля;

- Сохранение изображения в различных форматах.

3.1.2 Системные требования

- Системный блок;

- Монитор;

- Мышь либо наличие touchpad;

- Клавиатура;

- Процессор: Intel core i5;

- OЗУ: 6,00 ГБ;

- Операционная система Windows 8/10;

- Тип системы: 64-разрядная операционная система.

3.1.3 Файлы программы

Приложение состоит из следующих файлов:

- Файл «Saper.exe», являющийся главным файлом приложения.

3.1.4 Проверка программы

Для проверки программы следует запустить исполняемый файл «Saper.exe» двойным щелчком левой клавишей мыши.

Далее на экране появится поле для игры в Сапер, снизу меню для выбора сложности или правил.

## 3.2 Руководство программиста

3.2.1 Назначение и условия применения программы

Программа “Сапёр” является игровым приложением и предназначена для реализации игры “Сапёр”.

Минимальные системные требования для работы приложения:

- Операционная система Windows Vista;

- Оперативная память 1ГБ;

- Цветной монитор с адаптером VGA, и разрешением не ниже 1400х800.

3.2.2 Характеристики программы

Программа представляет собой приложение, состоящее из исполняемого файла «saper.exe», расположенном в одном каталоге и картинок, расположенных в каталоге images.

3.2.3 Обращение к программе

Для запуска программы перейти в каталог с программой и запустить на выполнение файл «saper.exe». В случае размещения программы в каталогах с ограниченными правами пользователя необходимо запускать программу с правами администратора (выбрать в контекстном меню программы).

## 3.3 Руководство пользователя

3.3.1 Назначение и условия применения программы

Программа предназначена для игры “Сапер”. Игра поможет развить логику, интуицию, внимание.

Минимальные системные требования для работы приложения:

- Операционная система Windows Vista;

- Оперативная память 1ГБ;

- Цветной монитор с адаптером VGA, и разрешением не ниже 1400х800.

3.3.2 Выполнение программы

Для запуска программы перейти в каталог с программой и запустить на выполнение файл «saper.exe». В случае размещения программы в каталогах с ограниченными правами пользователя необходимо запускать программу с правами администратора (выбрать в контекстном меню программы).

3.3.3 Сообщения программы

Для запуска программы следует дважды кликнуть левой кнопкой мыши по файлу «Saper.exe».

Далее на экране появится поле для игры в Сапер, снизу меню для выбора сложности или правил, как показано на рисунке 21.

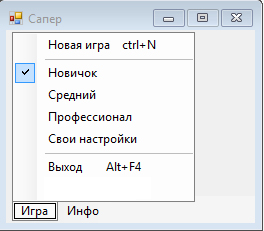


Рисунок 21 – Главное меню

Пользователю предоставляется выбор уровня сложности, по умолчанию – Новичок. Для того чтобы выбрать уровень игры – требуется зайти в подпункт “Игра”, затем выбрать уровень. Уровень “Новичок”, уровень “Средний”, уровень “Профессионал. Пользователь может выставить свои параметры игры, размер поля и количество мин, выбрав пункт Свои настройки.

После выбора предполагаемой клетки, в которой нету мины, игроку открывается клетка или поле. При ошибочном выборе клетки – появляется окно “Вы проиграли”. В случае выигрыша - появляется окно “Вы победили”.

Для того чтобы ознакомится с правилами, пользователь должен нажать подпункт меню “Инфо”, и затем выбрать “Правила”. Из формы “Правила игры “Сапёр”” можно выйти с помощью кнопки “Принимаю”.

Заключение

В ходе курсового проектирования поставленные цели были достигнуты. Было разработано программное обеспечение, решающее требуемые задачи и отвечающее предъявленным требованиям.

В результате выполнения курсового проекта были выполнены следующие задачи:

- Создана игровое приложение “Сапёр”, обладающее настройками поля;

- Проведено тестирование разработанного приложения;

- Произведен функциональный выбор сложности игры.

- В результате выполнения проекта удалось проанализировать функцию случайного выпадение чисел, т.к. для игры «Сапер» главной задачей компьютера является расстановка мин в произвольном порядке.

К достоинствам программного средства можно отнести простоту освоения, использования и низкие системные требования.

# Список использованных источников

1. Введение в язык C# и .NET Framework. – [Электронный ресурс]. – Режим доступа: <https://msdn.microsoft.com/ru-ru/library/z1zx9t92.aspx>
2. ГОСТ 7.1. – 2003. Библиографическая запись. Библиографическое описание. Общие требования и правила составления. – М.: ИПК Издательство стандартов, 2004. – 169 с.
3. ГОСТ 7.32 – 2001. Система стандартов по информацию, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления. – М.: ИПК Издательство стандартов, 2001. – 21 с.
4. ГОСТ 7.82 – 2001. Библиографическая запись. Библиографическое описание электронных ресурсов. Общие требования и правила составления. – М.: ИПК Издательство стандартов, 2001. – 21 с.
5. Гуриков С.Р. Введение в программирование на языке Visual C#: учебное пособие. – М.: ФОРУМ: ИНФРА-М, 2013.
6. Макконелл С. Совершенный код – Microsoft Press, Русская Редакция, 2013.
7. Описание разработки сложных алгоритмов игры Сапёр. – [Электронный ресурс]. – Режим доступа https://habrahabr.ru/post/218057, свободный. – (Дата обращения: 03.11.2017).
8. Павловская Т.А. С#. Программирование на языке высокого уровня. Учебник для вузов. – СПб.: Питер, 2014.
9. Полное руководство по языку программирования C# 6.0 и платформе .NET 4.6. – [Электронный ресурс]. – Режим доступа: <http://metanit.com/sharp/tutorial/>
10. Рудаков А.В. Технология разработки программных продуктов: учебник для студ. учреждений сред. проф. образования. – М.: Академия, 2013.
11. Рудаков А.В., Федорова Г.Н. Технология разработки программных продуктов. Практикум: учеб. пособие для студ. учреждений сред. проф. образования. – М.: Академия, 2013.
12. Руководство по программированию в Windows Forms. – [Электронный ресурс]. – Режим доступа: <http://metanit.com/sharp/windowsforms/>

Приложение А

Программный модуль Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace Saper

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

CreatePole();

this.NewGame();

this.ClientSize = new Size(20\*MC + 1, 20\*MR + 23);

}

public void CreatePole()

{

for (int row = 0; row <= MR + 1; row++)

{

pole[row, 0] = -3;

pole[row, MC + 1] = -3;

}

for (int col = 0; col <= MC + 1; col++)

{

pole[0, col] = -3;

pole[MR + 1, col] += -3;

}

}

public static int MR = 10, MC = 10,MinNum = 10;

private int [,] pole = new int[MR+2,MC+2];

private int mins, flags; bool alreadySown = false;

private int status;

// 0 - начало игры

// 1 - игра

// 2 - результат

private void NewGame()

{

int row, col;

int n = 0;

int k;

for (row = 1; row <= MR; row++)

for (col = 1; col <= MC; col++)

pole[row, col] = 0;

Random rnd = new Random();

while (n != MinNum)

{

row = rnd.Next(1, MR + 1);

col = rnd.Next(1, MC + 1);

if (pole[row, col] != 9)

{

pole[row, col] = 9;

n++;

}

}

for (row = 1; row <= MR; row++)

for (col = 1 ;col <= MC; col++)

if (pole[row, col] != 9)

{

k = 0;

if (pole[row-1, col-1] == 9) k++;

if (pole[row-1, col] == 9) k++;

if (pole[row-1, col+1] == 9) k++;

if (pole[row, col-1] == 9) k++;

if (pole[row, col+1] == 9) k++;

if (pole[row+1, col-1] == 9) k++;

if (pole[row+1, col] == 9) k++;

if (pole[row+1, col+1] == 9) k++;

pole[row, col] = k;

}

status = 0;

mins = 0;

flags = 0;

}

private void showPole(Graphics g,int status)

{

for (int row = 1; row <= MR; row++)

for (int col = 1; col <= MC; col++)

this.kletka(g, row, col, status);

}

private int GetTopX(int col)

{

return (col - 1) \* 20 + 1;

}

private int GetTopY(int row)

{

return (row - 1) \* 20 + 1;

}

private void kletka(Graphics g, int row, int col, int status)

{

int x = GetTopX(col),

y = GetTopY(row);

if (pole[row, col] < 100)

{

g.FillRectangle(Brushes.Gray, x, y, x+20, y+20);

}

if (pole[row, col] >= 100)

{

if (pole[row, col] != 109)

{

g.FillRectangle(Brushes.White, x, y, x + 20, y + 20);

}

else

{

g.FillRectangle(Brushes.Red, x, y, x + 20, y + 2);

Bitmap img3 = new Bitmap("icon3.png");

g.DrawImage(img3, x-5, y +2);

}

if ((pole[row, col] >= 101) && (pole[row, col] <= 108))

{

g.DrawString((pole[row,col]-100).ToString(),new Font("Tahoma",10,System.Drawing.FontStyle.Bold),Brushes.Blue,x+3,y+2);

}

}

if (pole[row, col] >= 200)

{

this.flag(g,x,y);

}

g.DrawRectangle(Pens.Black, x - 1, y - 1, x + 20, y + 20);

if ((status == 2) && ((pole[row, col] % 10) == 9))

{

this.mina(g,x,y);

}

}

private void open(int row, int col)

{

int x = GetTopX(col),

y = GetTopY(row);

if (pole[row, col] == 0)

{

pole[row, col] = 100;

this.Invalidate(new Rectangle(x,y,20,20));

this.open(row,col-1);

this.open(row-1,col);

this.open(row,col+1);

this.open(row+1,col);

this.open(row-1,col-1);

this.open(row-1,col+1);

this.open(row+1,col-1);

this.open(row+1,col+1);

}

else

if ((pole[row, col] < 100) && (pole[row, col] != -3))

{

pole[row, col] += 100;

this.Invalidate(new Rectangle(x, y, 20, 20));

}

}

private void mina(Graphics g, int x, int y)

{

g.DrawImage(new Bitmap("bomb.bmp"), x, y);

}

private void flag(Graphics g, int x, int y)

{

Point[] p = new Point[3];

Point[] m = new Point[5];

p[0].X = x + 2; p[0].Y = y + 2;

p[1].X = x + 15; p[1].Y = y + 6;

p[2].X = x + 2; p[2].Y = y + 10;

g.FillPolygon(Brushes.Red,p);

g.DrawLine(Pens.Black,x+2,y+2,x+2,y+18);

m[0].X = x + 8; m[0].Y = y + 14;

m[1].X = x + 8; m[1].Y = y + 8;

m[2].X = x + 10; m[2].Y = y + 10;

m[3].X = x + 12; m[3].Y = y + 8;

m[4].X = x + 12; m[4].Y = y + 14;

g.DrawLines(Pens.Black, m);

}

private void Form1\_MouseDown(object sender, MouseEventArgs e)

{

if (status == 2)

{

return;

}

if (status == 0)

{

status = 1;

}

int row = (int)(e.Y / 20) + 1,

col = (int)(e.X/20)+1;

int x = GetTopX(col),

y = GetTopY(row);

if (e.Button == MouseButtons.Left)

{

if (pole[row, col] == 9)

{

pole[row, col] += 100;

status = 2;

BOOM(x, y);

this.Invalidate();

}

else if (pole[row, col] == 0)

open(row, col);

else if (pole[row, col] < 9)

{

pole[row, col] += 100;

this.Invalidate(new Rectangle(x, y, 20, 20));

}

}

if (e.Button == MouseButtons.Right)

{

if (pole[row, col] <= 9)

{

flags += 1;

if (pole[row, col] == 9)

{

mins += 1;

}

pole[row, col] += 200;

if ((mins == MinNum) && (flags == MinNum))

{

status = 2;

YouWon();

this.Invalidate();

}

else

{

this.Invalidate(new Rectangle(x, y, 20, 20));

}

}

else

{

if (pole[row, col] >= 200)

{

flags -= 1;

pole[row, col] -= 200;

this.Invalidate(new Rectangle(x, y, 20, 20));

}

}

}

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

this.showPole(e.Graphics,status);

}

private void YouWon()

{

alreadySown = true;

MessageBox.Show("Вы победили!", "Поздравляю!");

}

private void Form1\_MouseMove(object sender, MouseEventArgs e)

{

if ((mins == MinNum) && (flags == MinNum) && !alreadySown)

{

status = 2;

YouWon();

}

}

private void BOOM(int x, int y)

{

Graphics g = this.CreateGraphics();

Bitmap img1 = new Bitmap("icon1.png");

Bitmap img2 = new Bitmap("icon2.png");

Bitmap img3 = new Bitmap("icon3.png");

g.DrawImage(img1, x-5, y+2);

DateTime dt = DateTime.Now;

MessageBox.Show("Вы проиграли");

g.DrawImage(img2, x - 5, y + 2);

FormHelp f = new FormHelp(); f.Show(); f.Visible = false; f.Close();

g.DrawImage(img3, x - 5, y + 2);

}

private void НоваяИграToolStripMenuItem\_Click(object sender, EventArgs e)

{

alreadySown = false;

this.NewGame();

this.Invalidate();

}

private void Form1\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar.ToString() == '\_'.ToString())

{

alreadySown = false;

this.NewGame();

this.Invalidate();

}

}

private void ПравилаToolStripMenuItem\_Click(object sender, EventArgs e)

{

FormHelp Hlp = new FormHelp();

Hlp.Show();

}

public bool exitFromMenu = false;

private void ВыходToolStripMenuItem\_Click(object sender, EventArgs e)

{

exitFromMenu = true;

Application.Exit();

}

private void НовичокToolStripMenuItem\_Click(object sender, EventArgs e)

{

MR = 10; MC = 10; MinNum = 10;

pole = new int[MR+2,MC+2];

CreatePole();

this.NewGame();

this.ClientSize = new Size(20 \* MC + 1, 20 \* MR + 23);

НовичокToolStripMenuItem.Checked = true;

СреднийToolStripMenuItem.Checked = false;

ПрофессионалToolStripMenuItem.Checked = false;

СвоиНастройкиToolStripMenuItem.Checked = false;

this.Invalidate();

}

private void СреднийToolStripMenuItem\_Click(object sender, EventArgs e)

{

MR = 16; MC = 16; MinNum = 40;

pole = new int[MR+2,MC+2];

CreatePole();

this.NewGame();

this.ClientSize = new Size(20 \* MC + 1, 20 \* MR + 23);

СреднийToolStripMenuItem.Checked = true;

НовичокToolStripMenuItem.Checked = false;

ПрофессионалToolStripMenuItem.Checked = false;

СвоиНастройкиToolStripMenuItem.Checked = false;

this.Invalidate();

}

private void ПрофессионалToolStripMenuItem\_Click(object sender, EventArgs e)

{

MR = 16; MC = 30; MinNum = 99;

pole = new int[MR + 2, MC + 2];

CreatePole();

this.NewGame();

this.ClientSize = new Size(20 \* MC + 1, 20 \* MR + 23);

ПрофессионалToolStripMenuItem.Checked = true;

НовичокToolStripMenuItem.Checked = false;

СреднийToolStripMenuItem.Checked = false;

СвоиНастройкиToolStripMenuItem.Checked = false;

this.Invalidate();

}

private void СвоиНастройкиToolStripMenuItem\_Click(object sender, EventArgs e)

{

FormOsoblyvi FO = new FormOsoblyvi();

FO.ShowDialog();

pole = new int[MR + 2, MC + 2];

CreatePole();

this.NewGame();

this.ClientSize = new Size(20 \* MC + 1, 20 \* MR + 23);

СвоиНастройкиToolStripMenuItem.Checked = true;

НовичокToolStripMenuItem.Checked = false;

СреднийToolStripMenuItem.Checked = false;

ПрофессионалToolStripMenuItem.Checked = false;

this.Invalidate();

}

private void ИграToolStripMenuItem\_Click(object sender, EventArgs e)

{

}

private void menuStrip1\_ItemClicked(object sender, ToolStripItemClickedEventArgs e)

{

}

private void Form1\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

Close();

MessageBox.Show("Poka");

}

private void Form1\_Load(object sender, EventArgs e)

{

}

public static void GetFromOsoblyvi(int mr, int mc, int minnum)

{

MR = mr; MC = mc; MinNum = minnum;

}

private void ОПрограммеStripMenuItem\_Click(object sender, EventArgs e)

{

MessageBox.Show("Программа \"Сапёр\" \nАвтор: Лужецкий Вадим 4ПКС-114");

}

private void Form1\_Resize(object sender, EventArgs e)

{

this.ClientSize = new Size(20 \* MC + 1, 20 \* MR + 23);

}

}

}

Программный модуль FormHelp.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace Saper

{

public partial class FormHelp : Form

{

public FormHelp()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void label2\_Click(object sender, EventArgs e)

{

}

}

}

Программный модуль FormSettings.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace Saper

{

public partial class FormSettings : Form

{

public FormSettings()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Form1.GetFromOsoblyvi(trackBarHeight.Value, trackBarWidth.Value, trackBarMineCount.Value);

this.Close();

}

private void button2\_Click(object sender, EventArgs e)

{

this.Close();

}

private void trackBarHeight\_Scroll(object sender, EventArgs e)

{

trackBarMineCount.Maximum = trackBarHeight.Value \* trackBarWidth.Value;

label1.Text = "Высота: " + trackBarHeight.Value;

label2.Text = "Количество мин: " + trackBarMineCount.Value;

}

private void trackBarWidth\_Scroll(object sender, EventArgs e)

{

trackBarMineCount.Maximum = trackBarHeight.Value \* trackBarWidth.Value;

label3.Text = "Ширина: " + trackBarWidth.Value;

label2.Text = "Количество мин: " + trackBarMineCount.Value;

}

private void trackBarMineCount\_Scroll(object sender, EventArgs e)

{

label2.Text = "Количество мин: " + trackBarMineCount.Value;

}

}

}

Программный модуль Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace Saper

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

Приложение Б

Титульный лист презентации к пояснительной записке.

