МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

Рекурсия и головоломки

(наименование работы)

ОТЧЕТ

по лабораторной работе №4

по дисциплине

Технологии программирования

(наименование дисциплины)

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Капранов С.Н.

(подпись) (фамилия, и., о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Русинова С.Д.

(подпись) (фамилия, и., о.)

18-ИСТ-4

(шифр группы)

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород, 2020

**ЗАДАНИЕ**

Реализовать игру Кен-кен.

Кен-кен – это математическая и логическая головоломка. Необходимо заполнить сетку цифрами так, чтобы в каждой строке и в каждом столбце они не повторялись. Число в углу каждого выделенного блока является результатом арифметической операции над цифрами в этом блоке. В отличие от судоку-убийцы (сум-до-ку), цифры внутри блока могут повторяться.

**Описание функций**

* Init – Открывает и читает файл, вызывает функции CreateMap и GenParts;
* CreateMap – Создает игровое поле (задает размер поля, добавляет клеточки, определяет расположение элементов);
* GenParts – Формирует области игрового поля;
* MakeRule – Создает правило заполнения для области;
* CompleteButton\_Click – Проверяет верность ответа;
* AnswerButton\_Click – Показывает ответ.

**Листинг программы**

**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.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace KENKENNN

{

public partial class Form1 : Form

{

// Экземпляр класса рандомизатора

public static Random r = new Random();

// Словарь с различными цветами

public static Dictionary<int, Color> colors = new Dictionary<int, Color>

{

{0, Color.FromArgb(181, 199, 247 )},

{1, Color.FromArgb(196, 181, 247 )},

{2, Color.FromArgb(229, 181, 247)},

{3, Color.FromArgb(247, 181, 232)},

{4, Color.FromArgb(247, 181, 199)},

{5, Color.FromArgb(247, 196, 181)},

{6, Color.FromArgb(181, 247, 228 )},

{7, Color.FromArgb(181, 233, 247)},

{8, Color.FromArgb(181, 247, 191)},

{9, Color.FromArgb(014,203,005)},

{10, Color.FromArgb(247, 181, 199)},

{11, Color.FromArgb(247, 196, 181)},

{12, Color.FromArgb(181, 247, 228 )},

{13, Color.FromArgb(181, 233, 247)},

};

// Задаем размер матрицы

public static int mapSize=4;

// Создаем массив текстбоксов

public static TextBox[][] textCell = new TextBox[mapSize][];

// Создем массив, в который будем хранить информацию о том

// Задействованы ли клеточки в областях

public static int[,] map;

// Размер одной клетки

int cellSize = 50;

// Создаем массив с ответами

public static int[,] correct;

public Form1()

{

InitializeComponent();

Init();

}

// Функиця при запуске

public void Init()

{

// Заполнение массива для областей

map = new int[mapSize, mapSize];

for (int i = 0; i < mapSize; i++)

{

for (int j = 0; j < mapSize; j++)

{

map[i, j] = 0;

}

}

correct = new int[mapSize,mapSize];

// Открытие и прочтение файла, на основе которого генерируется игра

string[] lines = File.ReadAllLines(@"..\123.txt");

string[][] cells = new string[lines.Length][];

for (int i = 0; i < cells.Length; i++)

{

cells[i] = lines[i].Split(new char[] { ' ', '\t' }, StringSplitOptions.RemoveEmptyEntries);

}

// Заполнение матрицы ответов

for (int i = 0; i < mapSize; i++)

{

for (int j = 0; j < mapSize; j++)

{

correct[i, j] = Int32.Parse(cells[i][j]);

}

}

//

CreateMap();

//

GenParts();

}

// Функиця создания поля игры

public void CreateMap()

{

// Задаем размер окна

this.Width = (mapSize + 1 ) \* (cellSize-6)+3;

this.Height = (mapSize + 3) \* cellSize-10;

// Создаем клеточки игрового поля

for (int i = 0; i < mapSize; i++)

{

textCell[i] = new TextBox[mapSize];

for (int j = 0; j < mapSize; j++)

{

textCell[i][j] = new TextBox();

textCell[i][j].Size = new Size(cellSize, cellSize);

textCell[i][j].Multiline = true;

textCell[i][j].Location = new Point(j \* cellSize, i \* cellSize);

textCell[i][j].Text = correct[i, j].ToString();

textCell[i][j].Font = new Font(textCell[i][j].Font.FontFamily, 16);

this.Controls.Add(textCell[i][j]);

}

}

// Задаем располодение кнопок

CompleteButton.Location = new Point(this.Width/2-67,mapSize\*50+10);

AnswerButton.Location = new Point(this.Width / 2 - 67, mapSize \* 50 + 50);

}

// Функиця генерации разделения игрового поля на области

public static void GenParts()

{

// Счетчик количества областей

int count = 0;

// Начинается проход массива

for (int i = 0; i < mapSize; i++)

{

for (int j = 0; j < mapSize; j++)

{

// Если клетка не относится ни к одной из областей, то

if (map[i, j] == 0)

{

// Данное значение отвечает за основную клетку области

int a = correct[i, j];

// Клетка рядом

int ai = 123;

// Клетка рядом

// Пока эти клетки пустые, данные значения будут заполнены флагами'123'

int aj = 123;

// Красим клетку в цвет, соответствующий

textCell[i][j].BackColor = colors[colors.Keys.ElementAt(count)];

// Помечаем клетку как задействованную в области

map[i, j] = 1;

// Если клетка не на границе

if (i!=mapSize-1)

{

// И если клетка рядом свободна

if(map[i+1,j] == 0)

{

// Помечаем значение соседней клетки, записываем туда

// Правильный ответ в этой клетке

ai = correct[i + 1, j];

// Красим в цвет основной клетки

textCell[i+1][j].BackColor = textCell[i][j].BackColor;

textCell[i+1][j ].Text = null;

// Помечаем клетку как задействованную в области

map[i+1, j] = 1;

}

}

// Если клетка не на границе

if (j != mapSize - 1)

{

// И если клетка рядом свободна

if (map[i , j+1] == 0)

{

// Помечаем значение соседней клетки, записываем туда

// Правильный ответ в этой клетке

aj = correct[i, j + 1];

// Красим в цвет основной клетки

textCell[i][j+1].BackColor = textCell[i][j].BackColor;

textCell[i][j + 1].Text = null;

// Помечаем клетку как задействованную в области

map[i, j+1] = 1;

}

}

//Создание правила для сформированной области

MakeRule(a,ai,aj,i,j);

count++;

}

}

}

}

// Создание правила для области игрового поля

// Сюда передаются значения основной и соседних клеточек, а так же их индексы

private static void MakeRule(int a, int ai, int aj,int i,int j)

{

// Рандомизатор целых чисел, выдает число от 1 до 4х

int f = r.Next(1, 4);

// Если область состоит только из одной клеточки, то заполняем

// ее числом из ответа, как указано в правилах игры

if (ai == 123 && aj == 123)

{

textCell[i][j].Text = a.ToString();

return;

}

// Если рандомизатор выдает число 1, то правило для области - сложение

if (f == 1)

{

if (ai == 123)

textCell[i][j].Text = (a + aj).ToString() + "+";

else if (aj == 123)

textCell[i][j].Text = (a + ai).ToString() + "+";

else

textCell[i][j].Text = (a + ai + aj).ToString() + "+";

}

// Если рандомизатор выдает число 1, то правило для области - вычитание

if (f == 2)

{

if (ai == 123)

textCell[i][j].Text = (a - aj).ToString() + "-";

else if (aj == 123)

textCell[i][j].Text = (a - ai).ToString() + "-";

else

textCell[i][j].Text = (a - ai - aj).ToString() + "-";

}

// Если рандомизатор выдает число 1, то правило для области - произведение

if (f == 3)

{

if (ai == 123)

textCell[i][j].Text = (a \* aj).ToString() + "\*";

else if (aj == 123)

textCell[i][j].Text = (a \* ai).ToString() + "\*";

else

textCell[i][j].Text = (a \* ai \* aj).ToString() + "\*";

}

}

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

{

}

// При нажатии на кнопку завершения

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

{

int f = 0;

string[,] answer = new string[mapSize, mapSize];

for(int i = 0; i < mapSize; i++)

{

for (int j = 0; j < mapSize; j++)

{

answer[i, j] = (textCell[i][j].Text);

}

}

for (int i = 0; i < mapSize; i++)

{

for (int j = 0; j < mapSize; j++)

{

if (answer[i, j] != correct[i, j].ToString())

f = 1;

}

}

if (f == 1)

{

MessageBox.Show("Incorrect!");

}

else

MessageBox.Show("Correct!");

}

// При нажатии на кнопку показа ответов

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

{

for (int i = 0; i < mapSize; i++)

{

for (int j = 0; j < mapSize; j++)

{

textCell[i][j].Text = correct[i, j].ToString();

}

}

}

}

}

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace KENKENNN

{

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());

}

}

}

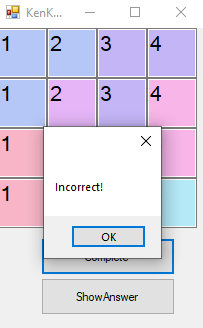
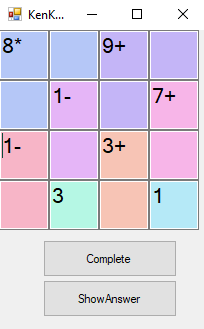


Рисунок 1 - Игровое поле после запуска Рисунок 2 – Вывод, если ответ неверный

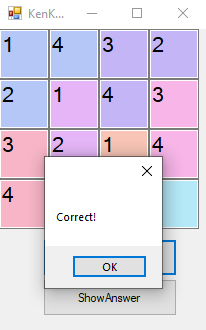


Рисунок 3 - Вывод, если ответ верный