В файле «Vipusknaya3» дополнение предыдущий версии, «Vipusknaya2»;

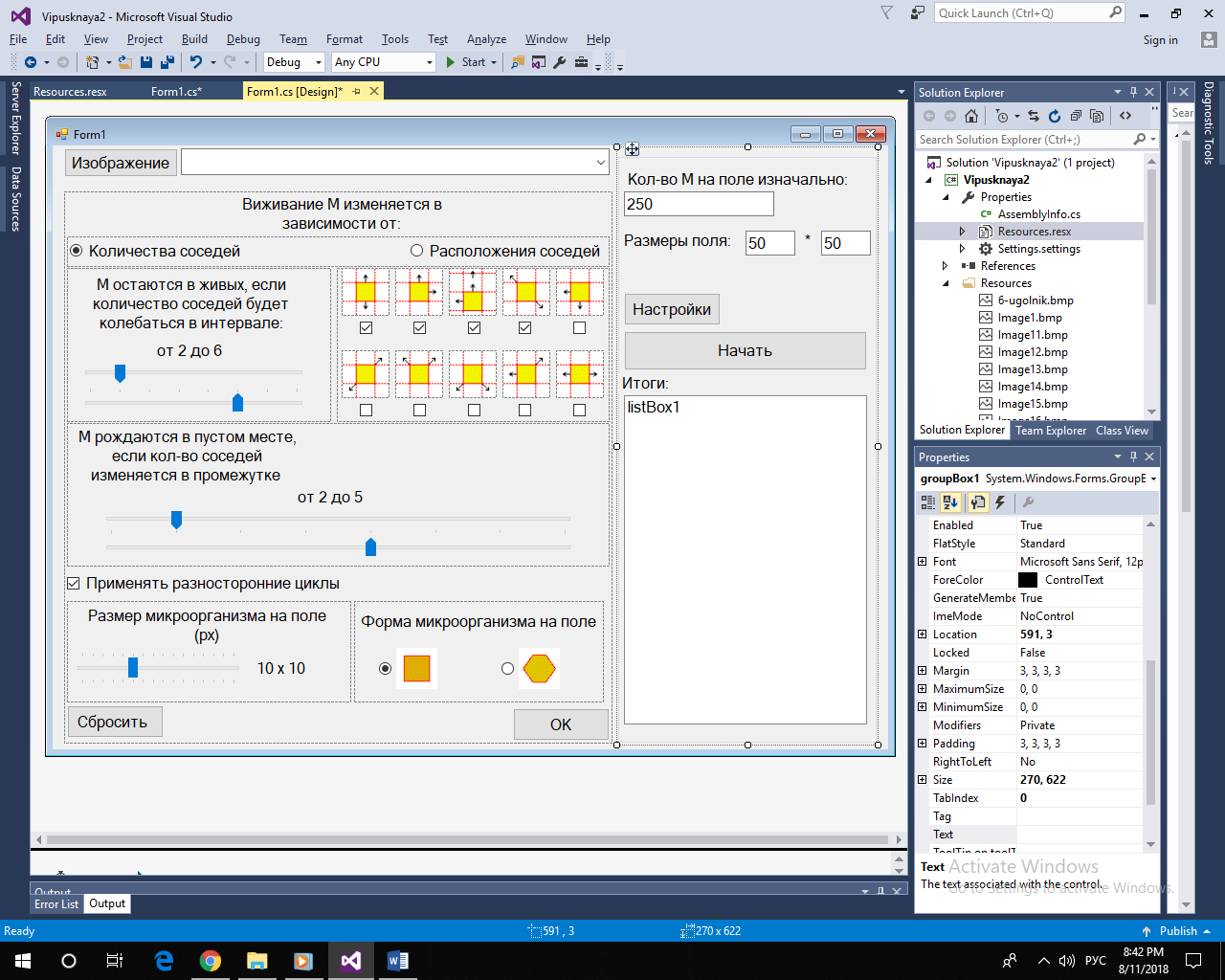
Здесь добавилось:

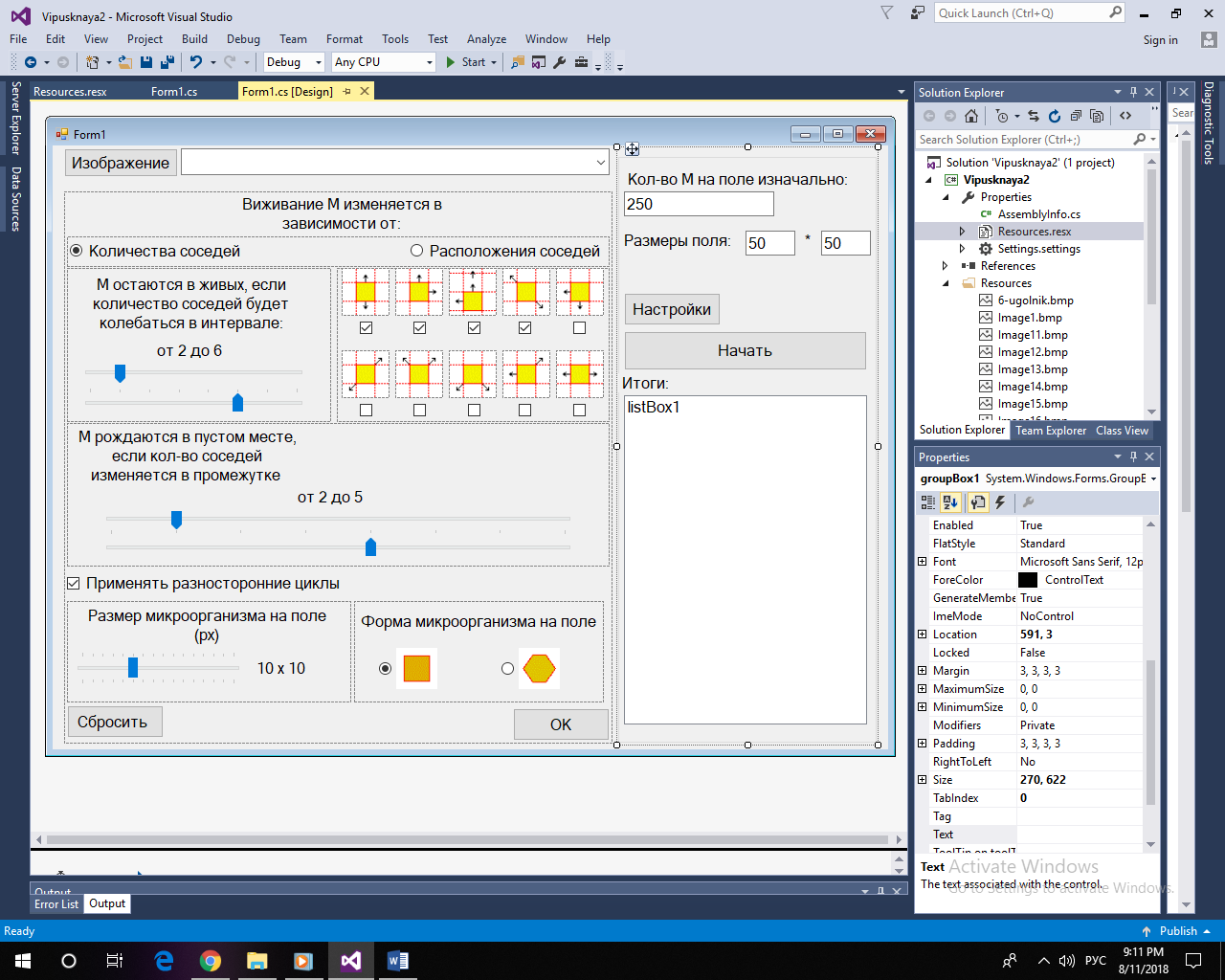
1. Настройки

**1.Выбор выживания:**

Если выбрано **выживание в зависимости от количества соседей**, то становится доступной настройка «М остаются в живых, если количество соседей будет колебаться в интервале».

Это означает, что при каждой итерации промежуток количества соседей будет изменятся в отведенных рамках. Например,

Если верхний trackBarы расположены следующим образом, то **самое малое условие выживания** колеблется от 2 (включительно) до среднего арифметического между макс. и мин. значениями, то есть до 4 (не включительно) , а **самое большое условие** – от 4 (не включительно) до 6 (включительно); В данном случае М живет, если кол-во соседей больше [2;4) и меньше чем (4;6];

Если выживание зависит от расположения соседей, то доступна следующая настройка:

Тут я полагаю все очевидно: те checkBoxы, которые отмечены – те расположения поочередно и проверяются;

В коде для исключения невыбранных комбинаций есть ограничитель:

void f\_limit()

{

if (f == 1 && checkBox2.Checked == false)

f++;

if (f == 2 && checkBox3.Checked == false)

f++;

if (f == 3 && checkBox4.Checked == false)

f++;

if (f == 4 && checkBox5.Checked == false)

f++;

if (f == 5 && checkBox6.Checked == false)

f++;

if (f == 6 && checkBox7.Checked == false)

f++;

if (f == 7 && checkBox8.Checked == false)

f++;

if (f == 8 && checkBox9.Checked == false)

f++;

if (f == 9 && checkBox10.Checked == false)

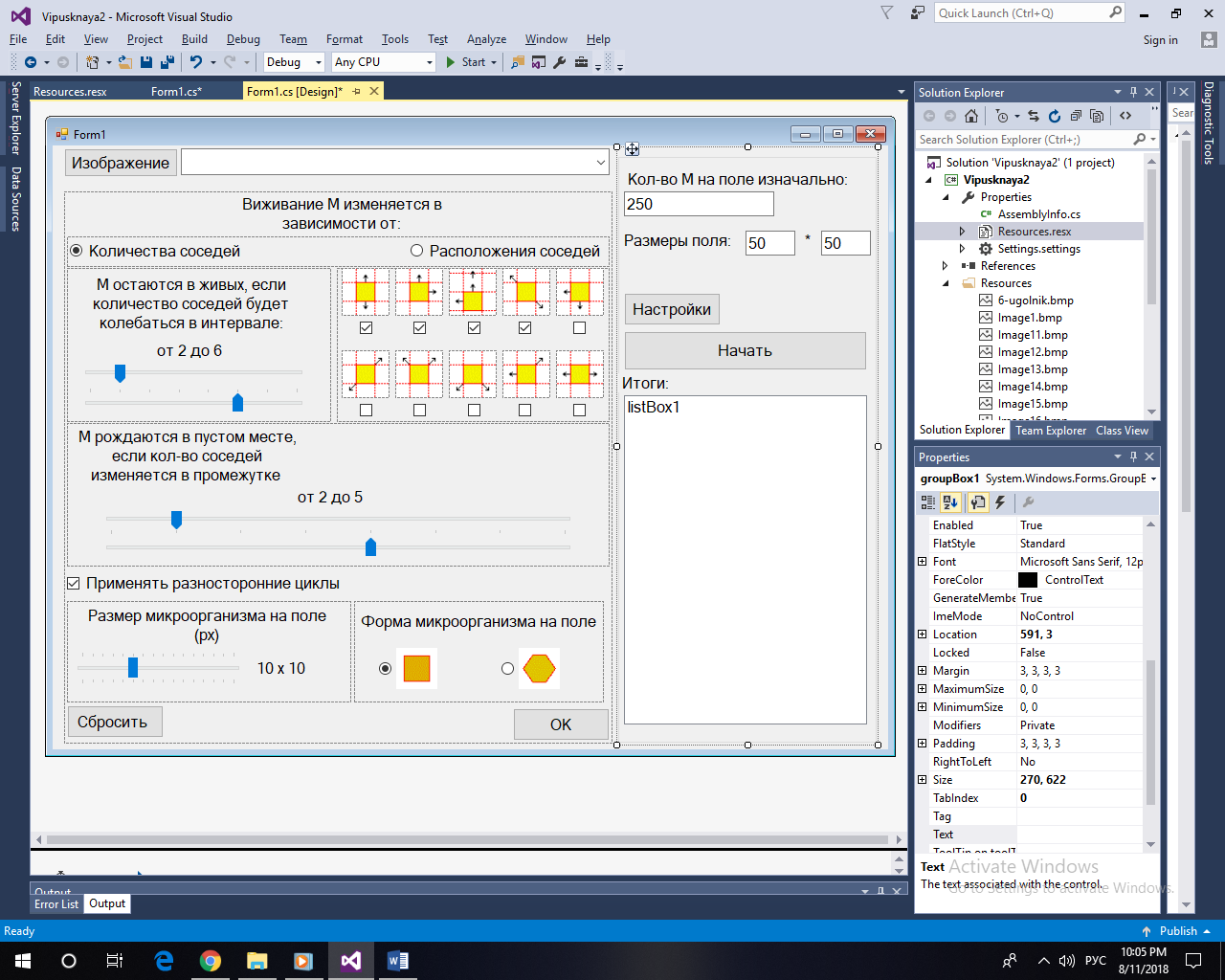
f++;

if (f == 10 && checkBox11.Checked == false)

f = 1;

}//переменная 'f' - порядковый номер комбинации

**2.Настройка рождения:**



Количество соседей, для рождения на пустом месте, поочередно изменяется (в данном случае от 2 до 5 включительно);

**3.Разносторонние циклы:**

Если эта настройка активирована, то все проверки на выживание и рождения М будут проверятся из-под разных углов в следующей последовательности:

по строкам слева на право и сверху вниз,

по столбцам с сверху вниз и справа на лево,

по строкам справа на лево и снизу вверх,

по столбцам снизу вверх и слева на право,

по столбцам сверху вниз и слева на право,

по строкам слева на право и снизу вверх ,

по столбцам снизу вверх и справа на лево,

по строкам справа на лево и сверху вниз;

Если же настройка отключена, то используется «стандартный цикл» по столбцам сверху вниз и слева на право.

**4.Размер микроорганизма**

Эта функция влияет только на визуальную часть проекта: значение на trackBar означает стороны квадрата в пикселях, выводимого на экран.

2) Концентрация:

1.Концентрация высчитывается с помощью формулы: конц. = n \* 100 / (h \* w), где

n – количество М; h и w – высота и ширина

2.Концентрация используется для построения графика

3)txt файл:

1.Когда игра окончена, то в файл записуется информация об этой игре следующим образом (по символам через пробел):

1) порядковый номер записи

2) всего итераций

3,4) высота и ширина поля

5-6-7 и до конца) итерация ‘і’, количество М[і], концентрация[і]

В коде над этим процессом работает подпрограмма Record():

void Record()

{

//по окончанию в txt файл записуем данные о

string[] s\_n = new string[10000];

string[] s\_conc = new string[10000];

int j = -1;//счетчик всех итераций в данной игре

//считываем данные с listBox

for (int i = 0; i < listBox1.Items.Count - 1; i++)

{

s2 = (listBox1.Items[i].ToString()).Split(' ');

if (s2[0] == "Итерация")

{

j++;

}

if (s2[0] == "n")

{

s\_n[j] = s2[2];

}

if (s2[0] == "Концентрация")

{

s\_conc[j] = s2[2];

}

}

//открываем поток

var sw = new StreamWriter(path, true);

m++;//счетчик всех записей

sw.Write(m + " " + (j + 1).ToString() + " " + w + " " + h + " ");

for (int i = 0; i < j + 1; i++)

{

sw.Write(i + " " + s\_n[i] + " " + s\_conc[i] + " ");

}

sw.WriteLine();

sw.Close();

}