Проект „Какурасу“

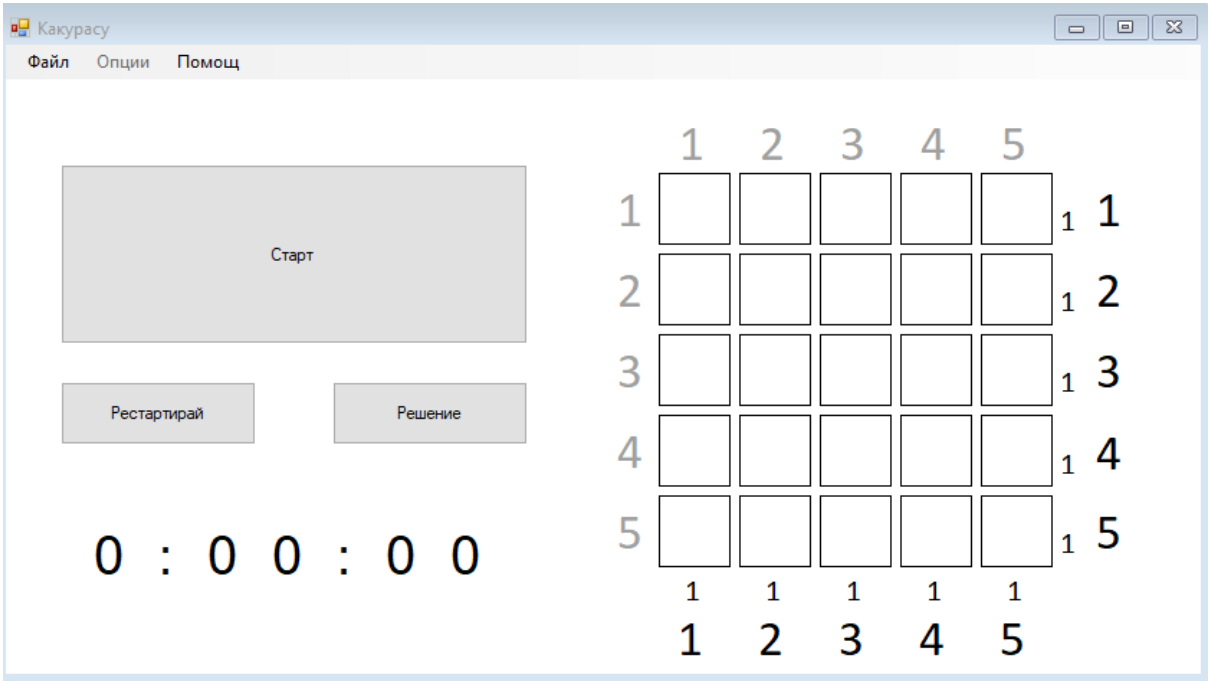
Правила на играта

Какурасу е логически пъзел с прости правила. Играе се на квадратна решетка. Някои клетки трябва да са черни по следните правила:

1. Сумата от всички черни клетки на всеки ред трябва да отговаря на числото в дясно от реда.
2. Сумата от всички черни клетки на всяка колона трябва да отговаря на числото под колоната.
3. Ако черна клетка е първа на реда или колоната нейната стойност е 1. Ако е втора стойността и е 2, ако е трета – 3 и т.н.

Реализация на програмата

Визуализация



Компоненти

1. **Игрална решетка** – 25 бутона (5x5); 20 labels (10 сиви, с които са номерирани редовете и колоните и 10 черни със стойността, която трябва да бъде направена).

2. Броячи – 10 labels, отговарящи на направената или оставащата сума (визуализирани са с 1-ци).

3. Бутон „Старт - Готово“ – дава начало и край на играта.

4. Бутон „Рестартирай“ – премахва всички черни квадратчета от игралната решетка и занулява броячите.

5. Бутон „Решение“ – решава играта.

6. Таймер – 7 labels, които измерват времето за решаване на играта.

6. Меню – състои се от 3 подменюта:

- „Файл“ – нова игра и излизане;
- „Опции“ – спиране на броячите и 2 опции за пускане на броячите (направена или оставаща сума);
- „Помощ“ – правила и относно;

Алгоритъм за създаване на игра

Алгоритъмът за създаването на играта е прост. Използваме матрица game[5,5], на който отначало даваме стойности 0, два масива column[5] и row[5], в които ще запазваме стойностите за всеки ред и колона, и btncol[5] и btnrow[5], в които впоследствие ще записваме стойностите за реда/колоната, когато играем. За да може всеки път да ни изкарва различни игри ще си създадем една функция, която ще ни връща всеки път различно число:

```
Random r = new Random();
int rand()
	{
		return r.Next() % 5;
	}
```

След това ще направим един цикъл от 0 до 4 (i) – броят на редовете. Ще си създадем енда променлива sqrow, на която ще задаваме различна стойност от 1 до 5. Тя отговаря на броя запълнени квадратчета на ред. После създаваме нов цикъл, който ще се върти sqrow пъти. В него всеки път ще намираме число n от 0 до 4, което ще отбелязваме като запълнено и ще променяме стойността на game[i,n] = 1;. Ако game[i,n] е равно на 1 ще се намира ново число n и така sqrow пъти. След това преброяваме на всеки ред и на всяка колона каква стойност имаме и я записваме в

column[] и row[]. Даваме стойностите на масивите като текстове на label и правим съответния label видим. Source code:

```
void igra()
    {
        int sqrow, n, sumarow, sumacol;
        for (int i = 0; i < 5; i++)
        {
            for (int j = 0; j < 5; j++) game[i, j] = 0;
            btncol[i] = 0;
            btnrow[i] = 0;
        }
        for (int i = 0; i < 5; i++)
        {
            sqrow = rand() + 1;
            while (sqrow > 0)
            {
                do n = rand();
                while (game[i, n] == 1);
                game[i, n] = 1; sqrow--;
            }
        }
        for(int i = 0; i < 5; i++)
        {
            sumacol = 0; sumarow = 0;
            for (int j = 0; j < 5; j++)
            {
                if (game[i, j] == 1) sumacol = sumacol + j + 1;
                if (game[j, i] == 1) sumarow = sumarow + j + 1;
            }
            row[i] = sumarow;
            column[i] = sumacol;
        }
        label15.Text = column[0].ToString(); label15.Visible = true;
        label14.Text = column[1].ToString(); label14.Visible = true;
        label13.Text = column[2].ToString(); label13.Visible = true;
        label12.Text = column[3].ToString(); label12.Visible = true;
        label11.Text = column[4].ToString(); label11.Visible = true;
        label20.Text = row[0].ToString(); label20.Visible = true;
        label19.Text = row[1].ToString(); label19.Visible = true;
        label18.Text = row[2].ToString(); label18.Visible = true;
        label17.Text = row[3].ToString(); label17.Visible = true;
        label16.Text = row[4].ToString(); label16.Visible = true;

       }
```

Алгоритъм за пресмятане на стойностите

При натискането на всеки един бутон с ляво копче, той трябва да промени цвета си в черно или бяло, а с дясно копче се появчва хикс(ако знаем, че то няма да бъде маркирано). Тъй като всеки бутон изпълнява една и съща функция, за всеки на опцията за MouseUp задаваме една и съща функция Button_Click.Използваме тази опция, за да можем да имаме различни

възможности при натискането на ляв и десен бутон. В нея ще броим след натискането на бутон каква е неговата стойност и ще променяме btncol[] и btnrow[]. За разбиране кой бутон е натиснат, предварително трябва да променим стойностите на TabIndex за всеки един. Source code:

```
private void Button_Click(object sender, MouseEventArgs e)
 {
     Button b = (Button)sender;
     if (e.Button == MouseButtons.Right)
     {
         b.Image = Properties.Resources.Screenshot_2;
     }
     else
     {
         b.Image = null;
         b.Refresh();
         if (b.BackColor != Color.Black)
         {
             b.BackColor = Color.Black;
                    btncol[(b.TabIndex - 1) / 5] = btncol[(b.TabIndex - 1)
/ 5] + ((b.TabIndex - 1) % 5) + 1;
             btnrow[(b.TabIndex - 1) % 5] = btnrow[(b.TabIndex - 1) % 5] +
((b.TabIndex - 1) / 5) + 1;
         }
         else
         {
             b.BackColor = Color.White;
                    btncol[(b.TabIndex - 1) / 5] = btncol[(b.TabIndex - 1)
/ 5] - ((b.TabIndex - 1) % 5) - 1;
             btnrow[(b.TabIndex - 1) % 5] = btnrow[(b.TabIndex - 1) % 5] -
((b.TabIndex - 1) / 5) - 1;
         }
     }
     label1.Focus();
 }
```

Бутон „Старт – готово“

Декларираме си булева променлива start, на която ще даваме стойност false, когато играта не е започнала, и true, когато е. След като щракнем върху бутона „Старт“, текстът му се променя на „Готово“, таймерът се пуска, бутоните се отключват и се изпълнява функцията igra(), която създава играта. Проверява се дали броячите са пуснати или не. При натискане на бутона „Готово“ програмата проверява дали стойностите на btncol[] са равни на column[] и стойностите на btnrow[] и row[] също да са равни. Ако не са – изкарва съобщение, че имате грешки. Ако обаче са, таймерът се спира, бутоните се блокират и на екрана излиза съобщение, че играчът е спечелил. Source code:

```
private void Button26_Click(object sender, EventArgs e)
        {
            if (start == false)
            {
                igra();
                start = true;
                button26.Text = "Готово";
                timer1.Enabled = true;
                try
                {
                    foreach (var button in this.Controls.OfType<Button>())
                    {
                        button.Enabled = true;
                    }
                }
                catch { }
                опцииToolStripMenuItem.Enabled = true;
                createbtn();
            }
            else
                if (Enumerable.SequenceEqual(column, btncol) == true &&
Enumerable.SequenceEqual(row, btnrow) == true)
            {
                timer1.Enabled = false;
                MessageBox.Show("Поздравления !");
                try
                {
                    foreach (var button in this.Controls.OfType<Button>())
                    {
                        button.Enabled = false;
                    }
                }
                catch { }
            }
            else MessageBox.Show("Имате грешки");
            label1.Focus();
        }
```

Бутони „Рестарирай“ и „Решение“

Бутонът „Рестартирай“ ти премахва всички маркирани квадратчета (независимо дали са с хикс или черни) и нулира btncol[] и btnrow[], както и броячите. За целта използваме функция clear(bool f), където параметърът f е дали бутоните трябва да се активират или да се деактивират. Source code:

```
private void Button27_Click(object sender, EventArgs e)
        {
            clear(true);
            for (int i = 0; i < 5; i++)
            {
                btncol[i] = 0;
                btnrow[i] = 0;
            }
        }
```

Бутонът **„Решение“** решава играта, спира таймерът и прави броячите със стойности равни на реда или колоната. За целта първо изчистваме всички полета с функцията clear(bool f). След това проверяваме чрез вложени цикли проверяваме дали стойността на матрицата game[i,j] е равна на 1 и ако е, правим съответното квадратче черно използвайки TabIndex-а му. Деактивираме всеки един бутон. **Source code:**

```
private void Button28_Click(object sender, EventArgs e)
        {
            clear(true);
            for (int i = 0; i < 5; i++)
                for(int j = 0; j < 5; j++)
                    if(game[i,j] == 1)
                    {
                        try
                        {
                            foreach (var button in this.Controls.OfType<Button>())
                            {
                                if (button.TabIndex == i * 5 + j + 1) button.BackColor
= Color.Black;
                                button.Enabled = false;
                            }
                        }
                        catch { }
                    }
            for(int i = 0; i < 5; i++)
            {
                btncol[i] = column[i];
                btnrow[i] = row[i];
            }

            timer1.Enabled = false;
       }
```

Таймер

Таймерът е изграден от 7 labels, два от които са „:“. Декларираме си 5 променливи – ms01, ms10, s01, s10 и m. Задаваме им стойности 0. Техните стойности отговарят на съответния label. От toolbox-а вземаме timer. В свойствата му задаваме стойността на интервала да е 1 (1 милисекунда). След което във фукцията на таймер започваме за всеки tick да променяме стойността на ms01 с 1.66 (Попринцип трябва да увеличваме с 1, но тогава времето не отговаря на реално изминатото, затова използваме 1.66). Ако стойността на ms01 надвиши 10, увеличаваме с 1 стойността на ms10 и така за останалите 3 променливи. След това променяме текстовете на съответните labels със новите стойности (използваме функцията timer()). Тъй като стойността на ms01 е дробно число използваме вградената функция **`Math.Round(ms01, MidpointRounding.ToEven)`**, за да закръглим стойността и. **Source code:**

```
void timer()
        {
            label21.Text = Math.Round(ms01, MidpointRounding.ToEven).ToString();
            label22.Text = ms10.ToString();
            label24.Text = s01.ToString();
            label25.Text = s10.ToString();
            label27.Text = m.ToString();
        }

private void Timer1_Tick(object sender, EventArgs e)
        {
            ms01= ms01 + 1.66;
            if(ms01 >= 9.7)
            {
                ms01 = 0;
                ms10++;
            }
            if(ms10 == 10)
            {
                ms10 = 0;
                s01++;
            }
            if(s01 == 10)
            {
                s01 = 0;
                s10++;
            }
            if(s10 == 6)
            {
                s10 = 0;
                m++;
            }
            timer();
        }
```

Меню

Менюто е разделено на 3 подменюта – **„Файл“**, „Опции“ и „Помощ“. „Файл“ съдържа нови две подменюта – „Нова игра“ и „Изход“. Подменюто Нова игра нулира всички броячи и променливи, скрива всички labels, премахва решението от игровата решетка чрез функцията clear(bool f) и деактивира всички бутони освен „Старт“. Подменюто Изход просто затваря програмата.

Source code:

```
private void ПравилаToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        } // изход

private void НоваИграToolStripMenuItem_Click(object sender, EventArgs e)
        {
            label15.Visible = false;
            label14.Visible = false;
            label13.Visible = false;
            label12.Visible = false;
            label11.Visible = false;
```

```
            label20.Visible = false;
            label19.Visible = false;
            label18.Visible = false;
            label17.Visible = false;
            label16.Visible = false;
            clear(false);
            start = false;
            timer1.Enabled = false;
            button26.Text = "Старт"; button26.Enabled = true;
            ms01 = 0; ms10 = 0; s01 = 0; s10 = 0; m = 0; timer();
            button27.Enabled = false;
            button28.Enabled = false;
            broyachi(false);
        } // нова игра
```

Подменюто „Опции“ съдържа 3 нови подменюта – **„Изключени броячи“** (Маркирано по подразбиране отначалото), **„Броячи за оставащи“**, **„Броячи за направени“.** С тези броячи играчът си улеснява играта, защото не трябва да следи какав стойност е направил за всяка ред/колона. След натискане на подменю то се маркира и в зависимост от това кое е маркирано се визуализира до игралната решетка. Създаваме функцията broyachi(bool f), която ни показва броячите, както и функциите ost() (за оставащите) и napr() (за направените).

```
private void БроячЗаОставащиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            броячЗаОставащиToolStripMenuItem.Checked = true;
            броячЗаНаправениToolStripMenuItem.Checked = false;
            изключенБроячToolStripMenuItem.Checked = false;
            broyachi(true); ost();

        }

private void БроячЗаНаправениToolStripMenuItem_Click(object sender, EventArgs e)
        {
            броячЗаОставащиToolStripMenuItem.Checked = false;
            броячЗаНаправениToolStripMenuItem.Checked = true;
            изключенБроячToolStripMenuItem.Checked = false;
            broyachi(true); napr();
        }

private void ИзключенБроячToolStripMenuItem_Click(object sender, EventArgs e)
        {
            броячЗаОставащиToolStripMenuItem.Checked = false;
            броячЗаНаправениToolStripMenuItem.Checked = false;
            изключенБроячToolStripMenuItem.Checked = true;
            broyachi(false);
        }
```

Последното подменю е „Помощ“ съдържа две нови подменюта – **„Правила“** и **„Относно“**. „Правила“ изкарва прозорец, на който са написани правилата на играта, „Относно“ показва кой е направил играта.

Цялостен source code

```
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;

namespace WindowsFormsApp1
{

    public partial class Form1 : Form
    {
        int[,] game = new int[5,5];
        int[] column = new int[5];
        int[] row = new int[5];
        int[] btncol = new int[5];
        int[] btnrow = new int[5];
        int  ms10 = 0, s01 = 0, s10 = 0, m = 0;
        double ms01 = 0;
        bool start = false;
        public Form1()
        {
            InitializeComponent();
        }
        Random r = new Random();
        int rand()
        {
            return r.Next() % 5;
        }
        void igra()
        {
            int sqrow, n, sumarow, sumacol;
            for (int i = 0; i < 5; i++)
            {
                for (int j = 0; j < 5; j++) game[i, j] = 0;
                btncol[i] = 0;
                btnrow[i] = 0;
            }
            for (int i = 0; i < 5; i++)
            {
                sqrow = rand() + 1;
                while (sqrow > 0)
                {
                    do n = rand();
                    while (game[i, n] == 1);
                    game[i, n] = 1; sqrow--;
                }
            }
            for(int i = 0; i < 5; i++)
            {
                sumacol = 0; sumarow = 0;
                for (int j = 0; j < 5; j++)
                {
                    if (game[i, j] == 1) sumacol = sumacol + j + 1;
                    if (game[j, i] == 1) sumarow = sumarow + j + 1;
                }
```

```
                row[i] = sumarow;
                column[i] = sumacol;
            }
            label15.Text = column[0].ToString(); label15.Visible = true;
            label14.Text = column[1].ToString(); label14.Visible = true;
            label13.Text = column[2].ToString(); label13.Visible = true;
            label12.Text = column[3].ToString(); label12.Visible = true;
            label11.Text = column[4].ToString(); label11.Visible = true;
            label20.Text = row[0].ToString(); label20.Visible = true;
            label19.Text = row[1].ToString(); label19.Visible = true;
            label18.Text = row[2].ToString(); label18.Visible = true;
            label17.Text = row[3].ToString(); label17.Visible = true;
            label16.Text = row[4].ToString(); label16.Visible = true;

        }

        void clear(bool f)
        {
            try
            {
                foreach (var button in this.Controls.OfType<Button>())
                {
                    if (button.BackColor == Color.Black) button.BackColor =
Color.White;
                    button.Enabled = f;
                    button.Image = null;
                    button.Refresh();
                }
            }
            catch { }
        }
        void timer()
        {
            label21.Text = Math.Round(ms01, MidpointRounding.ToEven).ToString();
            label22.Text = ms10.ToString();
            label24.Text = s01.ToString();
            label25.Text = s10.ToString();
            label27.Text = m.ToString();
        }

        private void Button26_Click(object sender, EventArgs e)
        {
            if (start == false)
            {
                igra();
                start = true;
                button26.Text = "Готово";
                timer1.Enabled = true;
                try
                {
                    foreach (var button in this.Controls.OfType<Button>())
                    {
                        button.Enabled = true;
                    }
                }
                catch { }
                if (бройчЗаОставащиToolStripMenuItem.Checked == true) { ost();
broyachi(true); }
                else if (бройчЗаНаправениToolStripMenuItem.Checked == true) { napr();
broyachi(true); }
                опцииToolStripMenuItem.Enabled = true;
            }
```

```
            else
                if (Enumerable.SequenceEqual(column, btncol) == true &&
Enumerable.SequenceEqual(row, btnrow) == true)
            {
                timer1.Enabled = false;
                MessageBox.Show("Поздравления !");
                try
                {
                    foreach (var button in this.Controls.OfType<Button>())
                    {
                        button.Enabled = false;
                    }
                }
                catch { }
            }
            else MessageBox.Show("Имате грешки");
            label1.Focus();
        }

        private void ПравилаToolStripMenuItem2_Click(object sender, EventArgs e)
        {
            MessageBox.Show("Какурасу се играе на правоъгълна решетка. Някои клетки
трябва да са черни по следните правила: " +
                "\n1. Сумата от всички чирни клетки на всеки ред трябва да отговаря на
числото в дясно от реда." +
                "\n2. Сумата от всички чирни клетки на всяка колона трябва да отговаря
на числото под колоната." +
                "\n3. Ако черна клетка е първа на реда/колоната нейната стойност е 1.
Ако е втора стойността и е 2 и т.н.", "Правила");
        }

        private void ПравилаToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void Button27_Click(object sender, EventArgs e)
        {
            clear(true);
            for (int i = 0; i < 5; i++)
            {
                btncol[i] = 0;
                btnrow[i] = 0;
            }
            if (бройчЗаОставащиToolStripMenuItem.Checked == true) ost();
            else if (бройчЗаНаправениToolStripMenuItem.Checked == true) napr();
            label1.Focus();
        }
private void Button28_Click(object sender, EventArgs e)
        {
            clear(true);
            for (int i = 0; i < 5; i++)
                for(int j = 0; j < 5; j++)
                    if(game[i,j] == 1)
                    {
                        try
                        {
                            foreach (var button in this.Controls.OfType<Button>())
                            {
                                if (button.TabIndex == i * 5 + j + 1) button.BackColor
= Color.Black;
                                button.Enabled = false;
```

```
                    }
                }
                catch { }
            }
        for(int i = 0; i < 5; i++)
        {
            btncol[i] = column[i];
            btnrow[i] = row[i];
        }

        timer1.Enabled = false;
        if (бројчЗаОставащиToolStripMenuItem.Checked == true) ost();
        else if (бројчЗаНаправениToolStripMenuItem.Checked == true) napr();
    }

    void broyachi(bool f)
    {
        label26.Visible = f;
        label29.Visible = f;
        label31.Visible = f;
        label30.Visible = f;
        label32.Visible = f;
        label33.Visible = f;
        label34.Visible = f;
        label35.Visible = f;
        label36.Visible = f;
        label37.Visible = f;
    }

    void ost()
    {
        label26.Text = (column[0] - btncol[0]).ToString();
        label29.Text = (column[1] - btncol[1]).ToString();
        label31.Text = (column[2] - btncol[2]).ToString();
        label30.Text = (column[3] - btncol[3]).ToString();
        label32.Text = (column[4] - btncol[4]).ToString();
        label37.Text = (row[0] - btnrow[0]).ToString();
        label35.Text = (row[1] - btnrow[1]).ToString();
        label36.Text = (row[2] - btnrow[2]).ToString();
        label34.Text = (row[3] - btnrow[3]).ToString();
        label33.Text = (row[4] - btnrow[4]).ToString();
    }

    void napr()
    {
        label26.Text = btncol[0].ToString();
        label29.Text = btncol[1].ToString();
        label31.Text = btncol[2].ToString();
        label30.Text = btncol[3].ToString();
        label32.Text = btncol[4].ToString();
        label37.Text = btnrow[0].ToString();
        label35.Text = btnrow[1].ToString();
        label36.Text = btnrow[2].ToString();
        label34.Text = btnrow[3].ToString();
        label33.Text = btnrow[4].ToString();
    }

    private void БројчЗаОставащиToolStripMenuItem_Click(object sender, EventArgs
e)
    {
        бројчЗаОставащиToolStripMenuItem.Checked = true;
        бројчЗаНаправениToolStripMenuItem.Checked = false;
```

```
            изключенБроячToolStripMenuItem.Checked = false;
            broyachi(true); ost();

        }

      private void БроячЗаНаправениToolStripMenuItem_Click(object sender, EventArgs e)
        {
            броячЗаОставащиToolStripMenuItem.Checked = false;
            броячЗаНаправениToolStripMenuItem.Checked = true;
            изключенБроячToolStripMenuItem.Checked = false;
            broyachi(true); napr();
        }

        private void ИзключенБроячToolStripMenuItem_Click(object sender, EventArgs e)
        {
            броячЗаОставащиToolStripMenuItem.Checked = false;
            броячЗаНаправениToolStripMenuItem.Checked = false;
            изключенБроячToolStripMenuItem.Checked = true;
            broyachi(false);
        }

        private void Button_Click(object sender, MouseEventArgs e)
        {
            Button b = (Button)sender;
            if (e.Button == MouseButtons.Right)
            {
                b.Image = Properties.Resources.Screenshot_2;
            }
            else
            {
                b.Image = null;
                b.Refresh();
                if (b.BackColor != Color.Black)
                {
                    b.BackColor = Color.Black;
                    btncol[(b.TabIndex - 1) / 5] = btncol[(b.TabIndex - 1) / 5] +
((b.TabIndex - 1) % 5) + 1;
                    btnrow[(b.TabIndex - 1) % 5] = btnrow[(b.TabIndex - 1) % 5] +
((b.TabIndex - 1) / 5) + 1;
                }
                else
                {
                    b.BackColor = Color.White;
                    btncol[(b.TabIndex - 1) / 5] = btncol[(b.TabIndex - 1) / 5] -
((b.TabIndex - 1) % 5) - 1;
                    btnrow[(b.TabIndex - 1) % 5] = btnrow[(b.TabIndex - 1) % 5] -
((b.TabIndex - 1) / 5) - 1;
                }
            }
            if (броячЗаОставащиToolStripMenuItem.Checked == true) ost();
            else if (броячЗаНаправениToolStripMenuItem.Checked == true) napr();
            label1.Focus();
        }

        private void ОтносноToolStripMenuItem_Click(object sender, EventArgs e)
        {
            MessageBox.Show("Направено от Никола Красимиров Михайлов.", "Какурасу");
        }

        private void НоваИграToolStripMenuItem_Click(object sender, EventArgs e)
        {
            label15.Visible = false;
```

```
            label14.Visible = false;
            label13.Visible = false;
            label12.Visible = false;
            label11.Visible = false;
            label20.Visible = false;
            label19.Visible = false;
            label18.Visible = false;
            label17.Visible = false;
            label16.Visible = false;
            clear(false);
            start = false;
            timer1.Enabled = false;
            button26.Text = "Старт"; button26.Enabled = true;
            ms01 = 0; ms10 = 0; s01 = 0; s10 = 0; m = 0; timer();
            button27.Enabled = false;
            button28.Enabled = false;
            broyachi(false);
        }

        private void Timer1_Tick(object sender, EventArgs e)
        {
            ms01= ms01 + 1.66;
            if(ms01 >= 9.7)
            {
                ms01 = 0;
                ms10++;
            }
            if(ms10 == 10)
            {
                ms10 = 0;
                s01++;
            }
            if(s01 == 10)
            {
                s01 = 0;
                s10++;
            }
            if(s10 == 6)
            {
                s10 = 0;
                m++;
            }
            timer();
        }
    }
}
```

Изготвил: Никола Михайлов

12[в] клас

22.10.2019 г.