Лабораторная работа №8

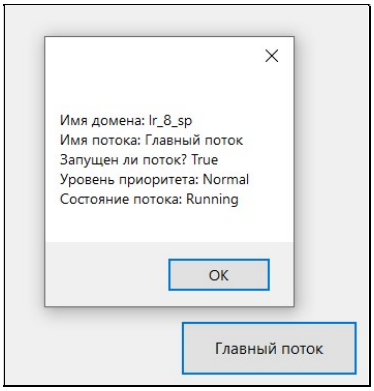
Процессы и потоки

А-13а-19 Кутдусов Р.К.

Ход работы

1. Программа должна выдать информацию о первичном (главном) потоке, создать еще три потока.

   Информация о первичном (главном потоке):




```
private void button1_Click(object sender, EventArgs e)
        {
            Thread main_thr = Thread.CurrentThread;
            main_thr.Name = "Главный поток";
            MessageBox.Show("Имя домена: " + Thread.GetDomain().FriendlyName +
                "\nИмя потока: " + main_thr.Name + "\nЗапущен ли поток? " +
                            main_thr.IsAlive.ToString() +
                "\nУровень приоритета: " + main_thr.Priority.ToString() +
                "\nСостояние потока: " + main_thr.ThreadState.ToString()
                );}
```

   Создание потоков:

```
firstPicThr = new Thread(firstPicStart);
secondPicThr = new Thread(secondPicStart);
systemInfoThr = new Thread(showSystemInfo);
```

2. Два вторичных потока должны быть созданы на основе двух разных функций, каждая из которых реализует метод случайного представления фрагментов одной из двух картинок (**каждая картинка режется на квадратики – не меньше девяти**). Поток должен завершиться, если «соберет» свою картинку в правильном порядке.

   Возьмём две картинки с достопримечательностями города Ульяновска. Разобьём наши картинки на 25 квадратиков. Будем генерировать рандомную перестановку, потом запустим алгоритм сортировки – для первой картинки – пузырьковая сортировка, для второй – сортировка вставками. Будем замерять время.

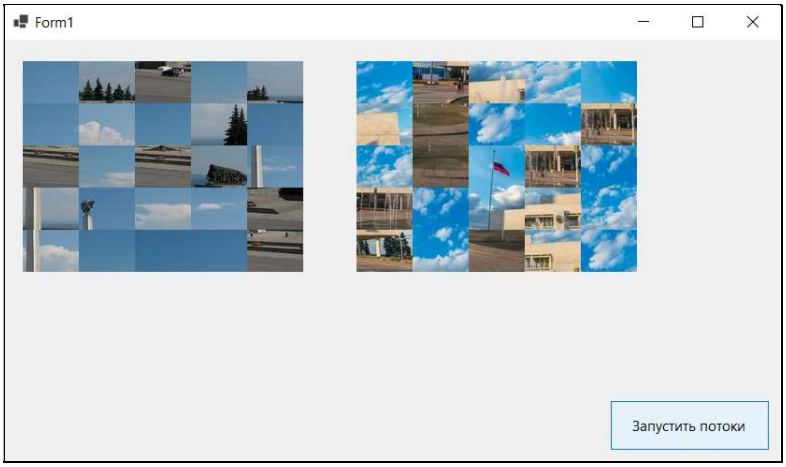
```
private Graphics firstPicGR; // объект типа graphics для каждой картинки свой picturebox
        private Graphics secondPicGR;

        private List<int> sorted; // отсортированный список

        private int width = 60; // длина и высота кусочка
        private int height = 45;
```

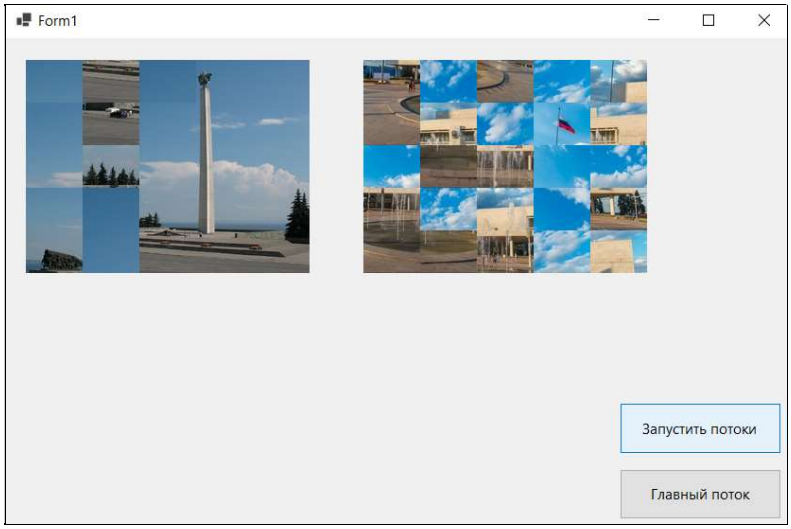
```
        private readonly List<Bitmap> firstPics; // контейнеры для кусочков
        private readonly List<Bitmap> secondPics;
        private List<int> firstPicsShuffle; // перестановка
        private List<int> secondPicsShuffle;

        // для измерения времени
        private Stopwatch firstPicStopwatch;
        private Stopwatch secondPicStopwatch;
        private Stopwatch systemInfoStopwatch;
        private TimeSpan firstPicSpan;
        private TimeSpan secondPicSpan;
        private TimeSpan systemInfoSpan;


        public Form1()
        {
            InitializeComponent();
            firstPicGR = pictureBox1.CreateGraphics();
            secondPicGR = pictureBox2.CreateGraphics();
            sorted = new List<int>();
            for (int i = 0; i < 25; i++) sorted.Add(i);
            firstPics = new List<Bitmap>();
            secondPics = new List<Bitmap>();
            firstPicsShuffle = new List<int>();
            secondPicsShuffle = new List<int>();
            firstPicStopwatch = new Stopwatch();
            secondPicStopwatch = new Stopwatch();
            systemInfoStopwatch = new Stopwatch();
        }

        private void LoadAndSplit() // загрузка изображений и разрезание на 25 кусков (картинки
одинакового размера)
        {
            var firstPic = Image.FromFile("C:\\Users\\user\\source\\repos\\lr_8_sp\\1.jpg");
            var secondPic = Image.FromFile("C:\\Users\\user\\source\\repos\\lr_8_sp\\2.jpg");
            for (int i = 0; i < 5; i++) {
                for (int j = 0; j < 5; j++) {
                    // var index = i * 5 + j;
                    firstPics.Add(new Bitmap(width, height));
                    secondPics.Add(new Bitmap(width, height));
                    var graphics1 = Graphics.FromImage(firstPics.Last<Bitmap>());
                    var graphics2 = Graphics.FromImage(secondPics.Last<Bitmap>());
                    graphics1.DrawImage(firstPic, new Rectangle(0, 0, width, height), new
Rectangle(i * width, j * height, width, height), GraphicsUnit.Pixel);
                    graphics2.DrawImage(secondPic, new Rectangle(0, 0, width, height), new
Rectangle(i * width, j * height, width, height), GraphicsUnit.Pixel);
                    graphics1.Dispose();
                    graphics2.Dispose();
                }
            }
        }

        private void Shuffle(ref List<int> indexes) // перемешивание элементов упорядоченного списка
        {
            var random = new Random();
            for (var i = indexes.Count - 1; i >= 0; i--)
            {
                var j = random.Next(i + 1);
                var tmp = indexes[i];
                indexes[i] = indexes[j];
                indexes[j] = tmp;
            }
        }

        private void bubbleSort(List<int> arr) // алгоритм пузырьковой сортировки (для первой)
        {
            for (int i = 0; i < arr.Count - 1; ++i)
            {
                bool swapped = false;
```

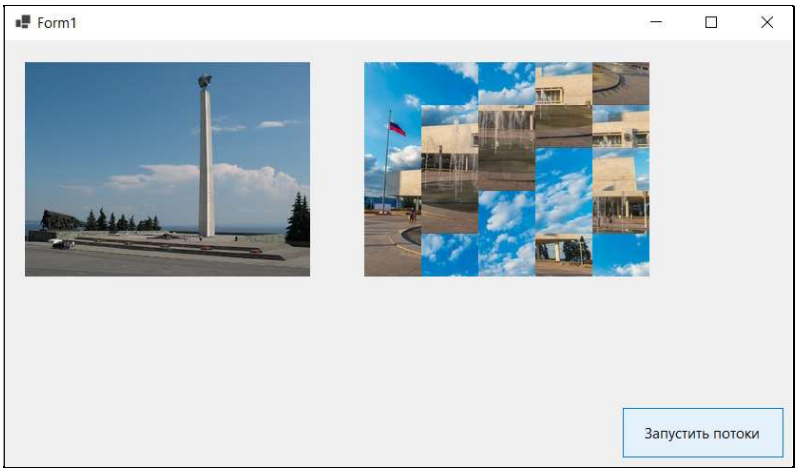
```
            for (int j = 1; j < arr.Count - i; ++j)
            {
                if (arr[j - 1] > arr[j])
                {
                    int buf = arr[j - 1];
                    arr[j - 1] = arr[j];
                    arr[j] = buf;
                    swapped = true;
                }
            }
            DrawImage(firstPicGR, firstPics, arr); // отрисовываем
            Thread.Sleep(200);
            if (!swapped)
                break;
        }
    }

    private void insertionSort(List<int> arr) // алгоритм сортировки вставками (для второй)
    {
        int x;
        int j;
        for (int i = 1; i < arr.Count; i++)
        {
            x = arr[i];
            j = i;
            while (j > 0 && arr[j - 1] > x)
            {
                int buf = arr[j];
                arr[j] = arr[j - 1];
                arr[j - 1] = buf;
                j -= 1;
            }
            arr[j] = x;
            DrawImage(secondPicGR, secondPics, arr); // отрисовываем
            Thread.Sleep(200);
        }
    }

    private void firstPicStart() // функция для первого потока
    {
        firstPicStopwatch.Restart();
        firstPicsShuffle.Clear();
        firstPicsShuffle.AddRange(sorted);
        Shuffle(ref firstPicsShuffle);
        DrawImage(firstPicGR, firstPics, firstPicsShuffle);
        bubbleSort(firstPicsShuffle);
        firstPicSpan = firstPicStopwatch.Elapsed;
    }

    private void secondPicStart() // функция для второго потока
    {
        secondPicStopwatch.Restart();
        secondPicsShuffle.Clear();
        secondPicsShuffle.AddRange(sorted);
        Shuffle(ref secondPicsShuffle);
        DrawImage(secondPicGR, secondPics, secondPicsShuffle);
        insertionSort(secondPicsShuffle);
        secondPicSpan = secondPicStopwatch.Elapsed; }
```

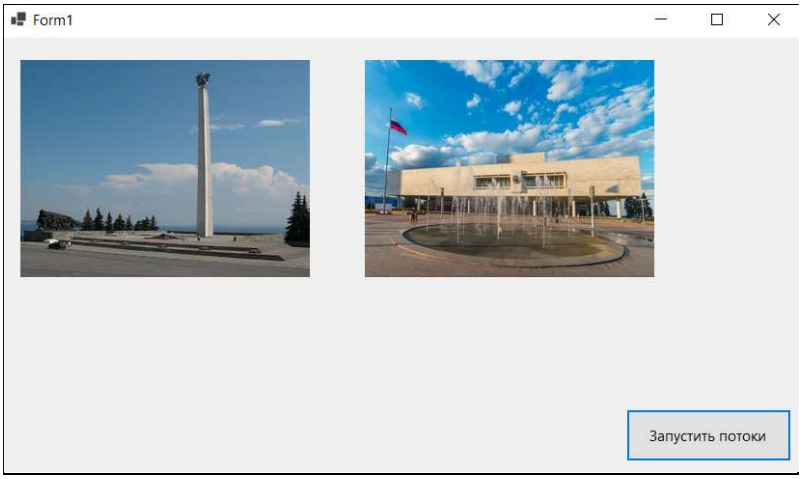
Далее можем видеть, что сортировка пузырьком, собирающая первую картинку, справилась чуть быстрее, чем сортировка вставками.

3. Приложение должно выводить на экран эти «разрезанные» картинки на экран (в выбранном в потоке порядке).


Form1
Запустить потоки


Form1
Запустить потоки
Главный поток

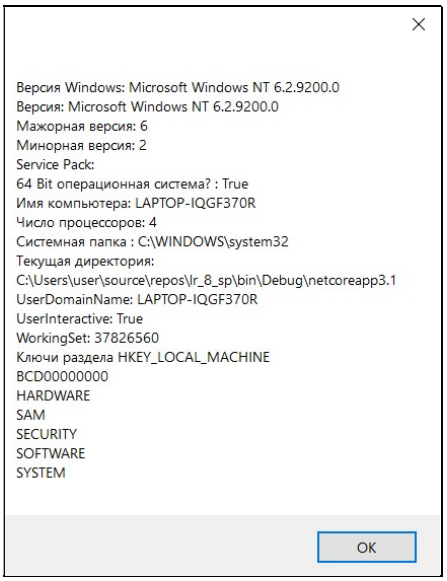

Form1
Запустить потоки


Form1
Запустить потоки

Отрисовка происходит при помощи функции:

```
private static void DrawImage(Graphics graphics, IReadOnlyList<Bitmap> image, IReadOnlyList<int>
permutation)
        {
            for (int i = 0; i < 5; i++)
                for (int j = 0; j < 5; j++)
                    graphics.DrawImage(image[permutation[i * 5 + j]], new Point(i * 60, j * 45))
        }
```

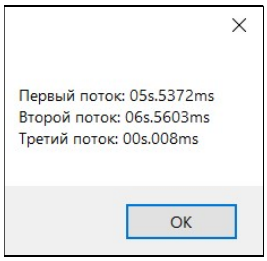
4. Третий поток должен собрать информацию о компьютерной системе (не менее 10-ти параметров), подготовить картинку для демонстрации этой информации на экране (**в виде рекламы или презентации**) и ждать события о завершении двух предыдущих потоков (информация о системе должна быть представлена пользователю, когда произойдет завершение двух потоков).




```
  private void showSystemInfo()
        {
            var info = GetSystemInfo();
            firstPicThr.Join();
            secondPicThr.Join();
            MessageBox.Show(info);
        }

        private string GetSystemInfo()
        {
            systemInfoStopwatch.Restart();
            var output = new StringBuilder();
            OperatingSystem os = Environment.OSVersion;
            output.AppendFormat("Версия Windows: {0}\n", os);
            output.AppendFormat("Версия: {0}\n", os.VersionString);
            output.AppendFormat("Мажорная версия: {0}\n", os.Version.Major);
            output.AppendFormat("Минорная версия: {0}\n", os.Version.Minor);
            output.AppendFormat("Service Pack: {0}\n", os.ServicePack);
            output.AppendFormat("64 Bit операционная система? : {0}\n",
                Environment.Is64BitOperatingSystem.ToString());
            output.AppendFormat("Имя компьютера: {0}\n", Environment.MachineName);
            output.AppendFormat("Число процессоров: {0}\n", Environment.ProcessorCount);
            output.AppendFormat("Системная папка : {0}\n", Environment.SystemDirectory);
            output.AppendFormat("Текущая директория: {0}\n", Environment.CurrentDirectory);
            output.AppendFormat("UserDomainName: {0}\n", Environment.UserDomainName);
            output.AppendFormat("UserInteractive: {0}\n", Environment.UserInteractive);
            output.AppendFormat("WorkingSet: {0}\n", Environment.WorkingSet);

            Microsoft.Win32.RegistryKey rk = Microsoft.Win32.Registry.LocalMachine;
```

```
            string[] keys = rk.GetSubKeyNames();
            output.AppendFormat("Ключи раздела {0}\n", rk.Name);
            foreach (string s in keys)
                output.AppendFormat("{0} \n", s);

            systemInfoSpan = systemInfoStopwatch.Elapsed;
            return output.ToString()
  }
```

5. Программа должна также предоставить пользователю информацию о времени, в течение которого выполнялись потоки.




```
private void button2_Click(object sender, EventArgs e)
        {
            LoadAndSplit();

            firstPicThr = new Thread(firstPicStart);
            secondPicThr = new Thread(secondPicStart);
            systemInfoThr = new Thread(showSystemInfo);
            firstPicThr.Start();
            secondPicThr.Start();
            systemInfoThr.Start();

            firstPicThr.Join();
            secondPicThr.Join();
            systemInfoThr.Join();

            var worktime = $"Первый поток:
{firstPicSpan.TotalSeconds:00}s.{firstPicSpan.TotalMilliseconds:000}ms\n";
            worktime += $"Второй поток:
{secondPicSpan.TotalSeconds:00}s.{secondPicSpan.TotalMilliseconds:000}ms\n";
            worktime += $"Третий поток:
{systemInfoSpan.TotalSeconds:00}s.{systemInfoSpan.TotalMilliseconds:000}ms\n";
            MessageBox.Show(worktime);
        }
```