**Федеральное агентство связи**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Поволжский государственный университет телекоммуникаций и информатики»**

Факультет **Информационных систем и технологий**

Направление

(специальность) **Информатика и вычислительная техника**

Кафедра **Прикладной информатики**

**Лабораторная работа №1**

**Управление процессами и потоками в ОС.**

Руководитель преподаватель С.В. Малахов

Выполнил студент группы ИВТ-93 А.К. Михалев

Самара 2021

**Задание:**

Решить задачу производитель – потребитель с точки зрения синхронизации по общему буферу. Производитель и потребитель представлены порожденными потоками. Сделать возможным устанавливать значение размера буфера.

**Выполнение работы:**

Интерфейс был сгенирирован с QTDisigner. Для определение приоритета потока созданны радио кнопки. Если нужен максимальный приоритет то кнопку нужно выбрать, в противном случа у потока будет минимальный приоритет.

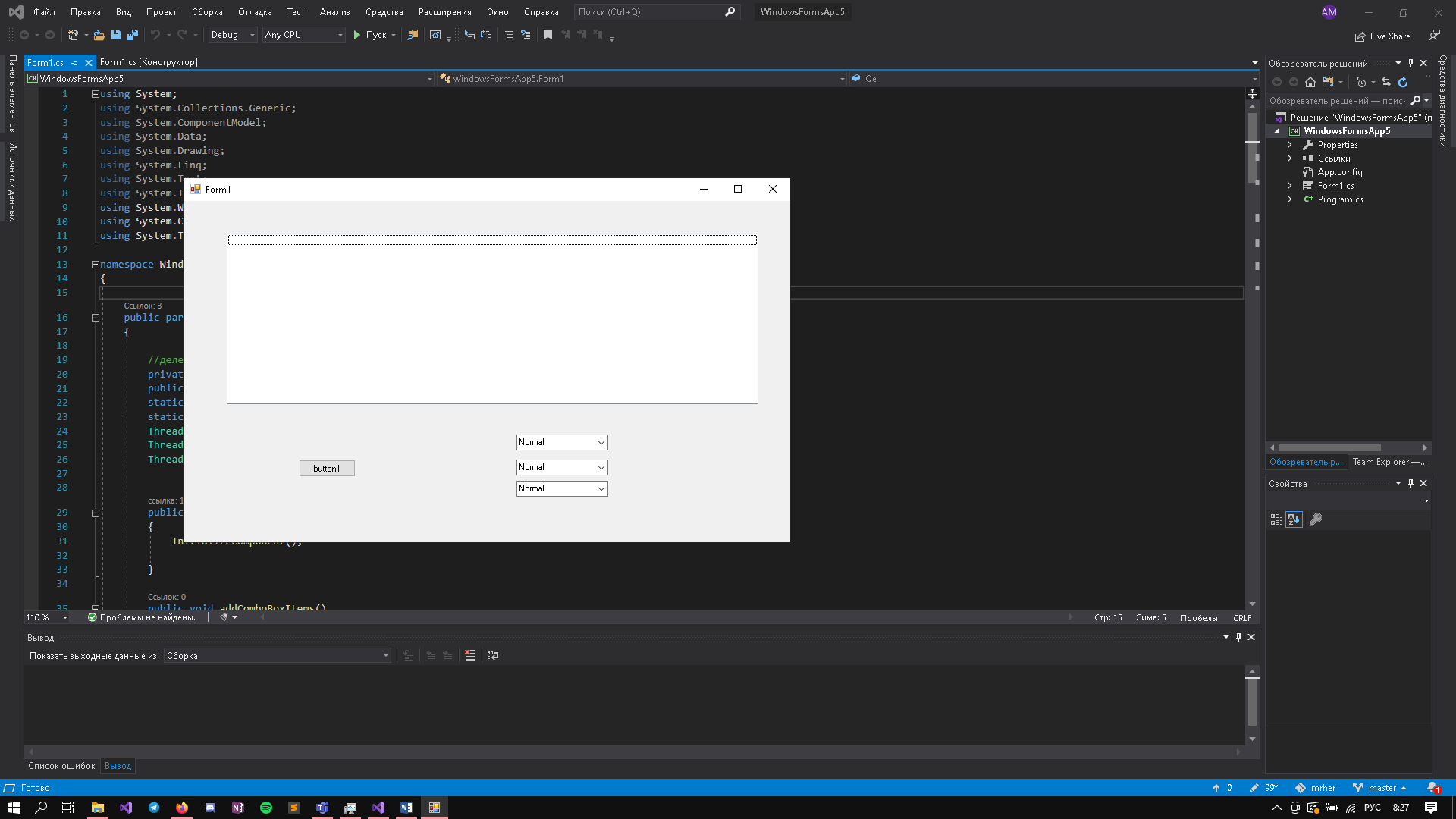


Рисунок 1. Скриншот графического интерфейса.

Прописывается необходимый функционал. При создание объекта сразу создается новый поток, в котором будет решать задачу потребитель производитель

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.Collections;

using System.Threading;

namespace WindowsFormsApp5

{

public partial class Form1 : Form

{

//делегат для связи с основным потоком

private delegate void DisplayGraphDelegate(int x);

public Queue Qe = new Queue();

static Semaphore s1 = new Semaphore(1, 2);

static Semaphore s2 = new Semaphore(1, 1);

Thread p1;

Thread p2;

Thread p3;

public Form1()

{

InitializeComponent();

}

public void addComboBoxItems()

{

string[] items =

{

ThreadPriority.Lowest.ToString(),

ThreadPriority.BelowNormal.ToString(),

ThreadPriority.Normal.ToString(),

ThreadPriority.AboveNormal.ToString(),

ThreadPriority.Highest.ToString()

};

comboBox1.Items.AddRange(items);

comboBox2.Items.AddRange(items);

comboBox3.Items.AddRange(items);

}

private void thread1()

{

var thread = new Thread(Producer);

thread.IsBackground = true;

foreach (ThreadPriority value in Enum.GetValues(typeof(ThreadPriority)))

{

if (value.ToString() == comboBox1.Text)

thread.Priority = value;

}

thread.Start();

}

private void thread2()

{

var thread = new Thread(Producer);

thread.IsBackground = true;

foreach (ThreadPriority value in Enum.GetValues(typeof(ThreadPriority)))

{

if (value.ToString() == comboBox2.Text)

thread.Priority = value;

}

thread.Start();

}

private void thread3()

{

var thread = new Thread(Consumer);

thread.IsBackground = true;

foreach (ThreadPriority value in Enum.GetValues(typeof(ThreadPriority)))

{

if (value.ToString() == comboBox3.Text)

thread.Priority = value;

}

thread.Start();

}

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

{

p1 = new Thread(Producer);

p2 = new Thread(Producer);

p3 = new Thread(Consumer);

p1.Start(0);

p2.Start(1);

p3.Start();

}

private void AddEntries(int x)

{

listBox1.Items.Add(x);

}

private void Producer(object i) // Производитель

{

while (true)

{

s1.WaitOne();

Monitor.Enter(Qe);

if (Qe.Count < 20)

Qe.Enqueue(i);

else

listBox1.Invoke(new DisplayGraphDelegate(AddEntries), Qe.Dequeue());

Thread.Sleep(1);

Monitor.Exit(Qe);

s1.Release();

}

}

public void Consumer() // Потребитель

{

string y = "get";

while (true)

{

s2.WaitOne();

if (Qe.Count < 1)

{

Thread.Sleep(100);

}

else

listBox1.Invoke(new Action(() => listBox1.Items.Add(y)));

Thread.Sleep(100);

s2.Release();

int BUFF = 10000; // буфер

byte[] but = new byte[BUFF]; // кладем буфер

}

}

[Serializable]

public class Customer

{

private string nameValue = string.Empty;

public Customer(String name)

{

nameValue = name;

}

public string Name

{

get { return nameValue; }

set { nameValue = value; }

}

public void Buff(int value)

{

int BUFF = value; // буфер

byte[] but = new byte[BUFF]; // кладем буфер

}

}

}

}

**Пример выполнения программы:**

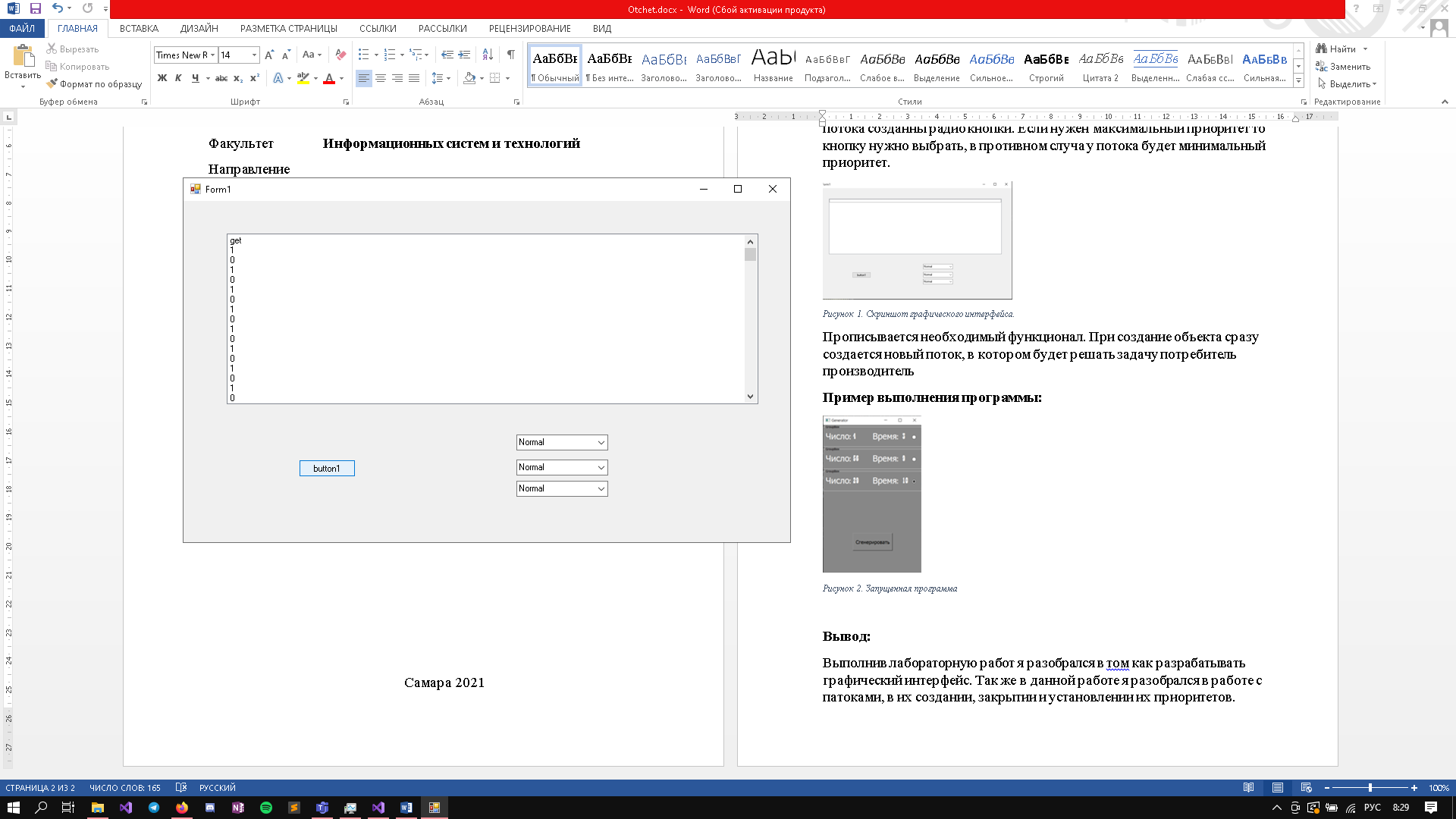


Рисунок 2. Запущенная программа

**Вывод:**

Выполнив лабораторную работ я разобрался в том как разрабатывать графический интерфейс. Так же в данной работе я разобрался в работе с патоками, в их создании, закрытии и установлении их приоритетов.