**Федеральное агентство связи**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

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

**Факультет информационных систем и технологий**

**Кафедра программного обеспечения и управления в технических системах**

Лабораторная работа №1 Управление процессами и потоками в

ОС

Выполнил студент

группы МОИС-01

Богрянцев Игорь

Проверил: к.т.н., доцент

Бадыков Р. И.

Самара, 2022 г.

str[0] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text0.txt";

str[1] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text1.txt";

str[2] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text2.txt";

str[3] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text3.txt";

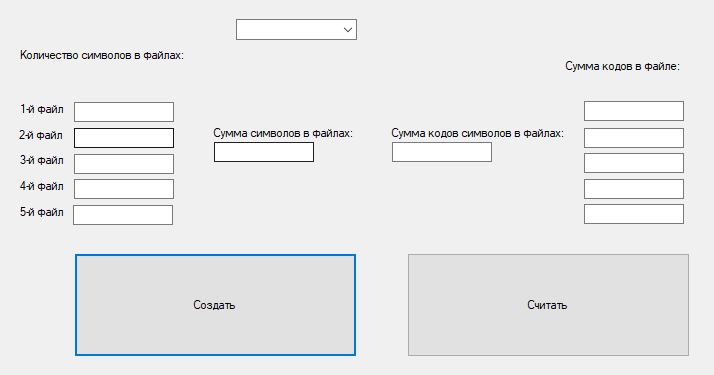
str[4] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text4.txt";Лабораторная работа №1 Управление процессами и потоками в ОС

Цель лабораторной работы Освоение навыков работы с потоками ОС Windows.

Вариант 3. Для 5-ти файлов (разного размера) требуется вычислить контрольную сумму (сумму кодов всех символов файла). Обработка каждого файла выполняется в отдельном потоке.

Ход работы:

Так как наше приложение должно иметь графический интерфейс, необходимо его создать:



1 )Чтобы создать и записывать в файлы данные, мы используем функцию WriteNow, передаем в нее аргумент Object a. В функции прописаны пути создания файла, задание размера создаваемого файла и его заполнение. Также производится вывод результатов с помощью Text.Box.

void WriteNow(Object a)

{

int j = (int)a;

Convert.ToInt32(j);

String[] str = new string[5];

str[0] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text0.txt";

str[1] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text1.txt";

str[2] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text2.txt";

str[3] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text3.txt";

str[4] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text4.txt"; string writePath = str[j];

int r = rand.Next(100, 1000);//Создаем рандомное количесвто символов в файле

char[] ch = new char[r];//Создаем массив символво размера r

string s = r.ToString();//Объявляем строку и переводим число в нее, нужно для вывода в TextBox

try

{

using (StreamWriter sw = new StreamWriter(writePath, false, System.Text.Encoding.Default))

{

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

{

ch[i] = (char)rand.Next(0x0410, 0x44F);

sw.Write(ch[i]);//Запись символов в файл

}

}

}

catch (Exception eb)

{

MessageBox.Show(eb.Message);

}

if (j == 0) { textBox2.Invoke(new Action(() => textBox2.Text = s)); }

else if (j == 1) { textBox3.Invoke(new Action(() => textBox3.Text = s)); }

else if (j == 2) { textBox4.Invoke(new Action(() => textBox4.Text = s)); }

else if (j == 3) { textBox5.Invoke(new Action(() => textBox5.Text = s)); }

else if (j == 4)

{

textBox1.Invoke(new Action(() => textBox1.Text = s));

int zig = Convert.ToInt32(textBox2.Text) + Convert.ToInt32(textBox3.Text) + Convert.ToInt32(textBox4.Text) + Convert.ToInt32(textBox5.Text) + Convert.ToInt32(textBox1.Text);

string sam = zig.ToString();

textBox6.Invoke(new Action(() => textBox6.Text = sam));

MessageBox.Show("Запись файлов выполнена. Количество символов:" + sam);

}

}

2)Функция ReadNow(), предназначена для считывания файла из заданного пути, узнаем результат преобразования символов файла в ASCII-код и производим вывод проделанной работы в TextBox.

private void ReadNow(object a)

{

String[] str = new string[5];

str[0] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text0.txt";

str[1] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text1.txt";

str[2] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text2.txt";

str[3] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text3.txt";

str[4] = @"C:\Users\smerc\OneDrive\Рабочий стол\test\text4.txt"; int j = (int)a;

//int sum = 0;

string stroka = str[j];

string text = File.ReadAllText(stroka);

byte[] bytes = Encoding.ASCII.GetBytes(text);

int result = 0;

for (int k = 0; k < text.Length; k++)

{

result += ((int)bytes[k]);

}

string s = result.ToString();

if (j == 0) { textBox7.Invoke(new Action(() => textBox7.Text = s)); }

else if (j == 1) { textBox8.Invoke(new Action(() => textBox8.Text = s)); }

else if (j == 2) { textBox9.Invoke(new Action(() => textBox9.Text = s)); }

else if (j == 3) { textBox10.Invoke(new Action(() => textBox10.Text = s)); }

else if (j == 4)

{

textBox11.Invoke(new Action(() => textBox11.Text = s));

int zig = Convert.ToInt32(textBox7.Text) + Convert.ToInt32(textBox8.Text) + Convert.ToInt32(textBox9.Text) + Convert.ToInt32(textBox10.Text) + Convert.ToInt32(textBox11.Text);

string sam = zig.ToString();

textBox12.Invoke(new Action(() => textBox12.Text = sam));

}

}

3)Для вызова функции WriteNow и создания потоков, используем свойство «Button\_Click», передавая параметр Object a. Задается выбор приоритета потока с помощью «case» и передачи значения в «ListBox».

private void button1\_MouseClick(object sender, MouseEventArgs e)

{

Form1 tws = new Form1();

for (int i = 0; i < 5; i++)

{

int imdb = i;

Thread myThread = new Thread(new ParameterizedThreadStart(WriteNow));

switch (comboBox1.SelectedIndex)

{

case 0:

myThread.Priority = ThreadPriority.Highest;

break;

case 1:

myThread.Priority = ThreadPriority.AboveNormal;

break;

case 2:

myThread.Priority = ThreadPriority.Normal;

break;

case 3:

myThread.Priority = ThreadPriority.BelowNormal;

break;

case 4:

myThread.Priority = ThreadPriority.Lowest;

break;

default:

myThread.Priority = ThreadPriority.Normal;

break;

}

myThread.Start(imdb);

}

}

4) Для вызова функции ReadNow и создания ее потоков, используем свойство «Button\_Click», передавая параметр Object a. Задается выбор приоритета потока с помощью «case» и передачи значения в «ListBox».

private void button2\_MouseClick(object sender, MouseEventArgs e)

{

for (int j = 0; j < 5; j++)

{

int imdb = j;

Thread myThread = new Thread(new ParameterizedThreadStart(ReadNow));

switch (comboBox1.SelectedIndex)

{

case 0:

myThread.Priority = ThreadPriority.Highest;

break;

case 1:

myThread.Priority = ThreadPriority.AboveNormal;

break;

case 2:

myThread.Priority = ThreadPriority.Normal;

break;

case 3:

myThread.Priority = ThreadPriority.BelowNormal;

break;

case 4:

myThread.Priority = ThreadPriority.Lowest;

break;

default:

myThread.Priority = ThreadPriority.Normal;

break;

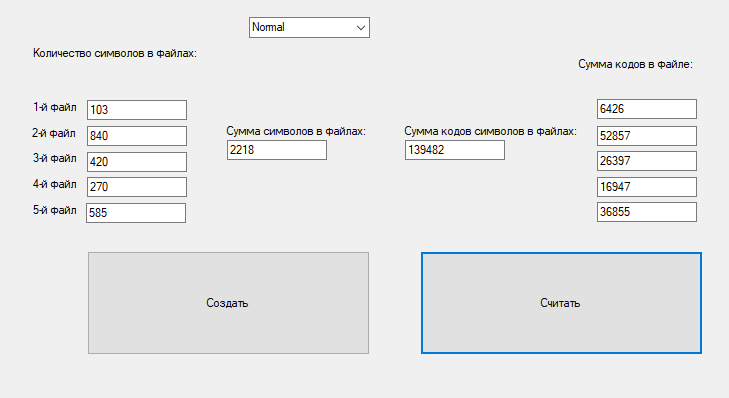
}

myThread.Start(imdb);

}

}

Финальный вид программы:



Вывод: в результате выполнения лабораторной работы, я научился создавать потоки(в том числе и параметризированные), задавать их приоритеты и т.д.