Министерство образования Республики Беларусь

Белорусский Национальный Технический Университет

Факультет информационных технологий и робототехники

Кафедра «Программное обеспечение вычислительной техники

и автоматизированных систем»

**Отчёт**

по лабораторной работе № 9

по дисциплине ***«РАЗРАБОТКА ПРИЛОЖЕНИЙ В ВИЗУАЛЬНЫХ СРЕДАХ»***

тема: **«РАЗРАБОТКА МНОГОПОТОЧНЫХ ПРИЛОЖЕНИЙ»**

|  |  |  |
| --- | --- | --- |
| Исполнитель: |  | студент группы 10701217  Голованов Павел Андреевич |
| Преподаватель: |  | Гурский Николай Николаевич |

2019 учебный год

**ЛАБОРАТОРНАЯ РАБОТА № 9**

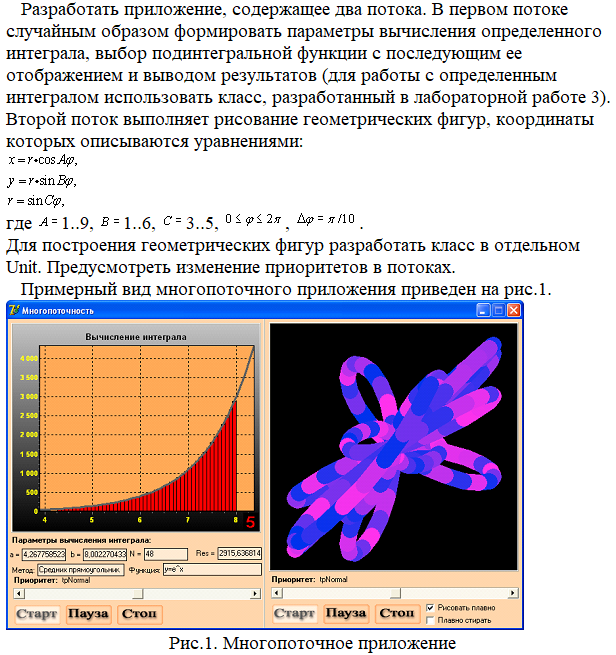
**РАЗРАБОТКА МНОГОПОТОЧНЫХ ПРИЛОЖЕНИЙ**

Цель работы:

Изучить принципы организации параллельных вычислений

в отдельных потоках.

Задание:



Результаты выполнения задания:

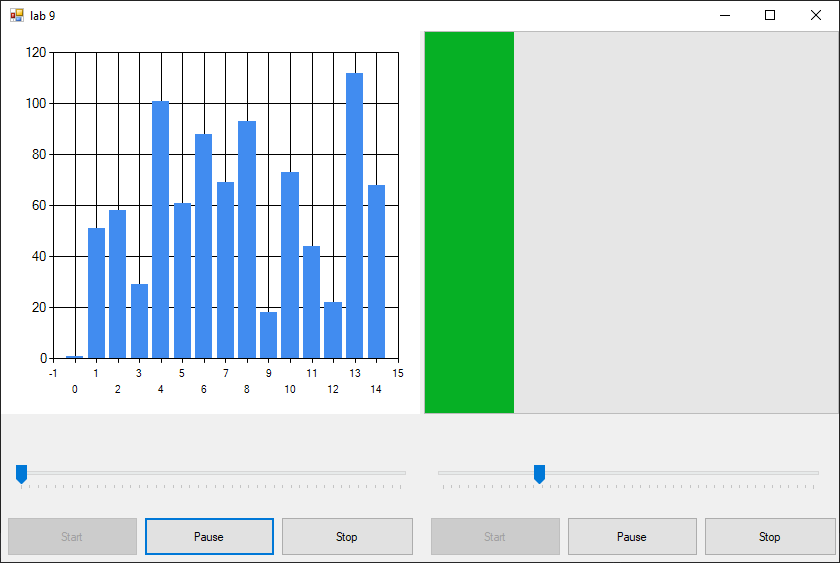


Рисунок 1 – Два одновременно работающих потока.

*ПРИЛОЖЕНИЕ A*

Листинг исходных кодов программ

**Файл Form1.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab\_9

{

public partial class Form1 : Form

{

Thread myThread1;

Thread myThread2;

bool thred1\_life;

int waitingtime1 = 1000;

bool thred2\_life;

int waitingtime2 = 1000;

public Form1()

{

InitializeComponent();

progressBar1.MarqueeAnimationSpeed = 1;

//progressBar1.Value += 50;

}

private void drawDonut()

{

Random random = new Random();

int counter = 0;

while (thred1\_life)

{

chart1.Invoke(new Action(() => { chart1.Series[0].Points.AddXY(counter++, random.Next(counter, counter+100)); }));

Thread.Sleep(waitingtime1);

}

}

private void th1\_start\_Click(object sender, EventArgs e)

{

//MessageBox.Show(" new thread ");

th1\_start.Enabled = false;

thred1\_life = true;

myThread1 = new Thread(new ThreadStart(drawDonut));

myThread1.Start();

th1\_stop.Enabled = true;

}

private void th1\_pause\_Click(object sender, EventArgs e)

{

if (myThread1.ThreadState == ThreadState.Suspended) { myThread1.Resume(); }

else { myThread1.Suspend(); }

}

private void th1\_stop\_Click(object sender, EventArgs e)

{

th1\_stop.Enabled = false;

thred1\_life = false;

chart1.Series[0].Points.Clear();

th1\_start.Enabled = true;

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

waitingtime1 = 1000 - trackBar1.Value;

}

private void trackBar2\_Scroll(object sender, EventArgs e)

{

waitingtime2 = 1000 - trackBar2.Value;

}

private void start\_bt\_Click(object sender, EventArgs e)

{

start\_bt.Enabled = false;

thred2\_life = true;

myThread2 = new Thread(new ThreadStart(drawProgressBar));

myThread2.Start();

stop\_bt.Enabled = true;

}

private void pause\_bt\_Click(object sender, EventArgs e)

{

if (myThread2.ThreadState == ThreadState.Suspended) { myThread2.Resume(); }

else { myThread2.Suspend(); }

}

private void stop\_bt\_Click(object sender, EventArgs e)

{

stop\_bt.Enabled = false;

thred2\_life = false;

chart1.Series[0].Points.Clear();

start\_bt.Enabled = true;

}

private void drawProgressBar()

{

while (thred2\_life)

{

progressBar1.Invoke(new Action(() => {

if (progressBar1.Value == 100) { progressBar1.Value = 0; Thread.Sleep(waitingtime2); }

progressBar1.Value++;

}));

Thread.Sleep(waitingtime2);

}

}

}

}