Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Объектно-ориентированные технологии программирования и стандарты проектирования (ООТПиСП)

ОТЧЕТ

по лабораторной работе №2

Тема работы: Работа с многопоточностью. Дескрипторы.

Выполнил

студент: гр. 951003 Антонов М.А.

Проверил: Деменковец Д.В.

Минск 2021

# Исходный код программы

**Задание 1:**

**Program.cs:**

using System;

using System.Threading;

namespace lab2

{

class Program

{

static void Main(string[] args)

{

int N = 10;

var mutex = new Mutex();

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

{

Thread thread = new Thread(() =>

{

mutex.Lock();

Console.WriteLine("current Thread id: " +

Thread.CurrentThread.ManagedThreadId + " lock

Thread");

Thread.Sleep(400);

Console.WriteLine("current Thread id: " +

Thread.CurrentThread.ManagedThreadId + " un

lock Thread");

mutex.Unlock();

}

);

thread.Start();

}

Console.ReadLine();

}

}

}

**Mutex.cs:**

using System.Threading;

namespace lab2

{

class Mutex

{

private int curId = -1;

public void Lock()

{

var id = Thread.CurrentThread.ManagedThreadId;

while (Interlocked.CompareExchange(ref this.curId,

id, -1) != -1)

{

Thread.Sleep(10);

}

}

public void Unlock()

{

var id = Thread.CurrentThread.ManagedThreadId;

Interlocked.CompareExchange(ref this.curId,

-1, id);

}

}

}

**Задание 2:**

**Program.cs:**

using System;

using System.Runtime.InteropServices;

namespace descriptors

{

class Program

{

public const int STD\_OUTPUT\_HANDLE = -11;

[DllImport("kernel32.dll", SetLastError = true)]

private static extern IntPtr GetStdHandle(int nStdHan

dle);

static void Main(string[] args)

{

OSHandle handle = new OSHandle(GetStdHandle(STD\_OUT

PUT\_HANDLE));

handle.Dispose();

}

}

}

**OSHandle.cs:**

using System;

namespace descriptors

{

public class OSHandle : IDisposable

{

[System.Runtime.InteropServices.DllImport("Kernel32")]

private extern static Boolean CloseHandle(IntPtr han

dle);

public IntPtr handle { get; set; }

private bool isDisposed;

public OSHandle(IntPtr handle)

{

this.handle = handle;

}

~OSHandle()

{

if (!isDisposed && handle != IntPtr.Zero)

{

Console.WriteLine("Closed handle: {0}",

this.handle);

CloseHandle(handle);

}

}

public void Dispose()

{

if (!isDisposed && handle != IntPtr.Zero)

{

GC.SuppressFinalize(this);

isDisposed = true;

Console.WriteLine("Closed handle: {0}",

this.handle);

CloseHandle(handle);

}

}

}

}