Белорусско-Российский университет

Кафедра «ПОИТ»

Отчёт по лабораторной работе №3 по дисциплине «Операционные системы»

**Работа с процессами и потоками в ОС Windows**

Выполнил Проверил

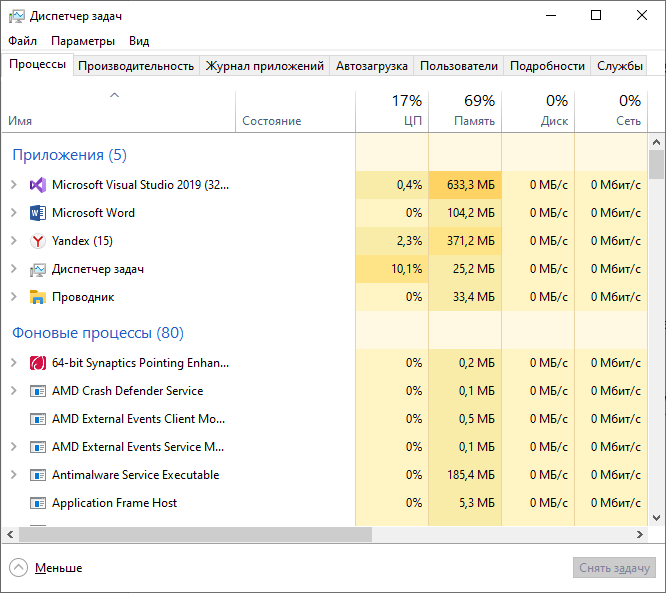
Ст. гр.АСОИР -181 Преподаватель

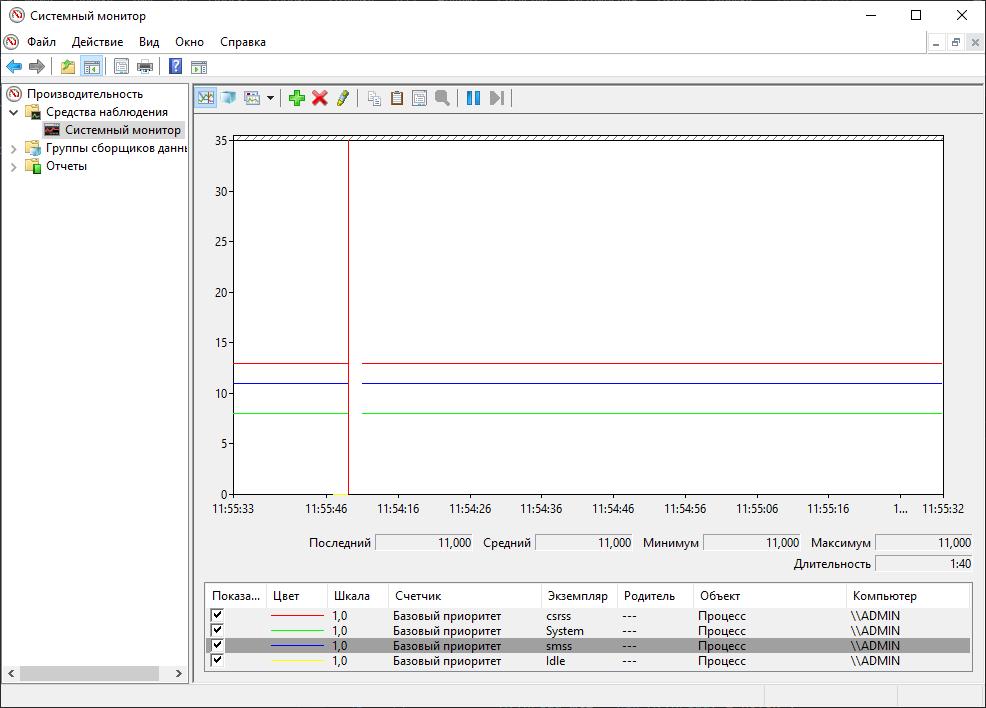
Ковалевский ТВ Зайченко Е. А.

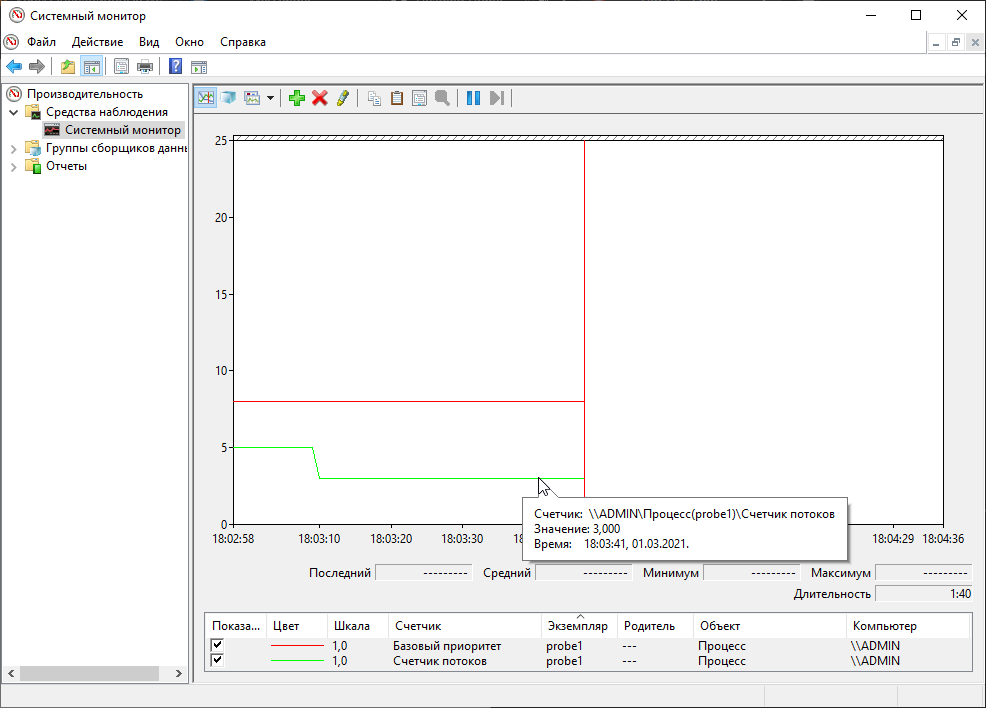
Могилев 2021

Цель: Овладеть навыками работы с процессами и потоками в ОС Windows с помощью WIN 32 API.

Запущенные в системе процессы:



Базовые приоритеты системных процессов: 

Информация о процессе probe1.exe. Базовый приоритет – 8 (нормальный), количество потоков при запуске – 5, затем – 3.

Исходный код probe1:

using System;

namespace probe1

{

class Program

{

static void Main(string[] args)

{

double a = 0, b = 0, c = 0, s = 0;

try

{

a = TryDoubleParse();

}

catch (ArgumentException e) { Console.WriteLine(e.Message); }

try

{

b = TryDoubleParse();

}

catch (ArgumentException e) { Console.WriteLine(e.Message); }

try

{

c = TryDoubleParse();

}

catch (ArgumentException e) { Console.WriteLine(e.Message); }

s = Math.Sqrt(a \* b \* c) / c;

Console.WriteLine(s);

}

static double TryDoubleParse()

{

double result = 0;

if (double.TryParse(Console.ReadLine(), out result))

{

return result;

}

else throw new ArgumentException("Input value should be a number!");

}

}

}

Задание: Создать приложение, запускающее два дочерних потока. Дожидаться в главном потоке завершения работы одного из запущенных потоков. Каждый поток выполняет ведение счетчика, предельное значение которого введено в главном потоке. Понизить приоритет одного из потоков на 2, а другого повысить на 2. Приостановить на некоторое время поток, имеющий наименьший приоритет. Вывести значение счетчика каждого из потоков.

*Исходный код:*

using System;

using System.Threading;

namespace Lab\_3

{

class Program

{

static int limit = 1000000;

static void Main(string[] args)

{

(MyThread mt1, MyThread mt2) = CreateThreads();

mt1.thread.Start();

mt2.thread.Start();

Console.WriteLine(PrintThreadInfo(mt1));

Console.WriteLine(PrintThreadInfo(mt2));

mt1.thread.Priority = ThreadPriority.Highest;

mt2.thread.Priority = ThreadPriority.Lowest;

Console.WriteLine(PrintThreadInfo(mt1));

Console.WriteLine(PrintThreadInfo(mt2));

PauseThread(mt2.thread, 1000);

mt2.thread.Join();

Console.WriteLine();

Console.WriteLine(PrintThreadInfo(mt1));

Console.WriteLine(PrintThreadInfo(mt2));

Console.ReadKey();

}

public static (MyThread, MyThread) CreateThreads()

{

MyThread mt1 = new MyThread("1 поток")

{

LimitValueOfVounter = limit

};

MyThread mt2 = new MyThread("2 поток")

{

LimitValueOfVounter = limit

};

return (mt1, mt2);

}

public static void PauseThread(Thread thread, int time)

{

int count = 0;

thread.Suspend();

do

{

count++;

}

while (count < time);

thread.Resume();

}

public static string PrintThreadInfo(MyThread myThread)

{

string defaultOut = $"{myThread.thread.Name} досчитал до {myThread.Сount}.";

return myThread.thread.IsAlive ? $"{defaultOut} Приоритет: {myThread.thread.Priority}" : defaultOut;

}

}

class MyThread

{

public int Сount { get; set; }

public int LimitValueOfCounter { get; set; }

public Thread thread;

static bool stop = false;

static string currentName;

public MyThread(string name)

{

Сount = 0;

thread = new Thread(new ThreadStart(Run))

{

Name = name

};

currentName = name;

}

void Run()

{

Console.WriteLine(thread.Name + " стартовал.");

do

{

Сount++;

if (currentName != thread.Name)

{

currentName = thread.Name;

Console.WriteLine("Текущий поток - " + currentName);

}

} while (!stop && (Сount < LimitValueOfVounter));

stop = true;

Console.WriteLine(thread.Name + " завершен.");

}

}

}

Результат:

