Министерство науки и образования Российской Федерации

Пензенский государственный университет

Кафедра «Системы автоматизированного проектирования»

**Отчет**

По лабораторной работе №1

По курсу «Операционные системы реального времени»

На тему «Порождение и завершение процессов и потоков»

Выполнил студент гр.20ВВС1

Горбунов Н.А.

Проверил:

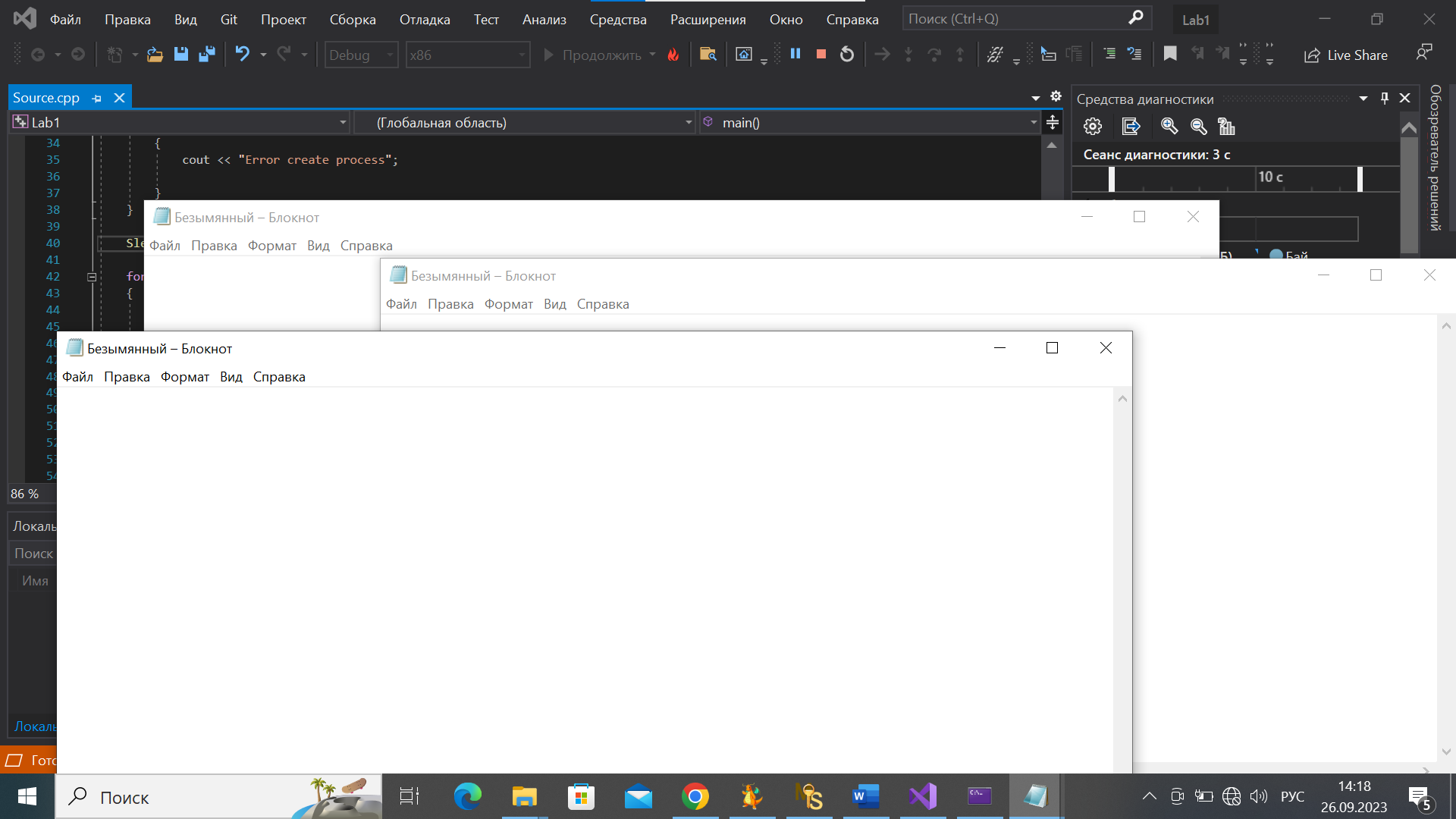
Гудков П.А.

Пенза, 2023

Цель: изучение методов и средств порождения процессов и нитей, изучение способов синхронизации процессов и нитей через ожидание окончания их выполнения.

Вариант 2: породить несколько процессов из одного. Все процессы должны завершаться по завершению первого.

Написал программу, которая порождает процессы приложения «блокнот»



И завершает процесс всех порождённых процессов через 5 секунд.

**Листинг:**

#include <windows.h>

#include <String>

#include <vector>

#include <exception>

#include <iostream>

typedef struct ProcessPGU {

STARTUPINFO si;

PROCESS\_INFORMATION pi;

bool complete;

} ProcessStruct;

using namespace std;

// =============================== Prototypes

ProcessStruct CreateNewProcess();

// ===============================

vector<ProcessStruct> arrayProcesses;

wstring path = L"C:\\Windows\\system32\\notepad.exe";

int main() {

for (int i = 0; i < 3; i++)

{

ProcessStruct newprocess = CreateNewProcess();

if (newprocess.complete)

{

arrayProcesses.push\_back(newprocess);

}

else

{

cout << "Error create process";

}

}

Sleep(5000);

for (int i = 0; i < arrayProcesses.size(); i++)

{

if (!arrayProcesses[i].complete) continue;

TerminateProcess(arrayProcesses[i].pi.hProcess, 1);

CloseHandle(arrayProcesses[i].pi.hProcess);

CloseHandle(arrayProcesses[i].pi.hThread);

}

return 1;

}

ProcessStruct CreateNewProcess()

{

STARTUPINFO si;

PROCESS\_INFORMATION pi;

ZeroMemory(&si, sizeof(si));

si.cb = sizeof(si);

ZeroMemory(&pi, sizeof(pi));

// Start the child process.

if (!CreateProcess(NULL, // No module name (use command line)

(LPWSTR)path.data(), // Command line

NULL, // Process handle not inheritable

NULL, // Thread handle not inheritable

FALSE, // Set handle inheritance to FALSE

0, // No creation flags

NULL, // Use parent's environment block

NULL, // Use parent's starting directory

&si, // Pointer to STARTUPINFO structure

&pi) // Pointer to PROCESS\_INFORMATION structure

)

{

printf("CreateProcess failed (%d).\n", GetLastError());

ProcessStruct a;

a.complete = false;

return a;

}

else {

ProcessStruct a;

a.complete = true;

a.pi = pi;

a.si = si;

return a;

}

}

**Вывод:** Научились взаимодействовать с процессами через WIN API.