Белорусский государственный университет

информатики и радиоэлектроники

Факультет компьютерных технологий

Лабораторная работа №2

«Процессы»

Выполнил ст. гр. 980161: Алейчик И.Д.

Приняла: Герман Ю.О.

Минск 2020

**Цель лабороторной работы:**

* изучить способы работы с процессами на языке С/С++.

**ОБЩЕЕ ЗАДАНИЕ:**

1. Набрать и выполнить приведенные файлы.
2. На основании представленных программок создать собственную программу, которая отыскивает exe-файл по части имени, запускает его на выполнение, затем проверяет, выполняется ли он, и выдает соответствующее сообщение.

**Ход выполнения работы:**

1А. ПОИСК ФАЙЛА (-ОВ) ПО ИМЕНИ

#include <iostream>

#include <tlhelp32.h>

#include "stdafx.h"

#include "windows.h"

#include "string.h"

int \_tmain(int argc, \_TCHAR\* argv[])

{

WIN32\_FIND\_DATA findFileData;

HANDLE hFind = INVALID\_HANDLE\_VALUE;

TCHAR directorySearch[] = TEXT("D:\\\*");

TCHAR fileSearch[] = TEXT("D:\\kat\*.txt");

hFind = FindFirstFile(fileSearch, &findFileData);

if (hFind == INVALID\_HANDLE\_VALUE) {

printf("Invalid file handle. Error is %u. \n", GetLastError());

}

else

{

wprintf(L"Found: %s \n", findFileData.cFileName);

while (FindNextFile(hFind, &findFileData)) {

wprintf(L"Found: %s \n", findFileData.cFileName);

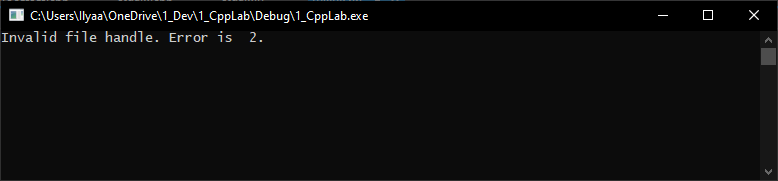
}

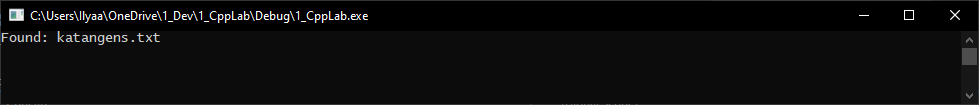
FindClose(hFind);

}

getchar();

}





1Б. ЗАПУСК ВНЕШНЕГО ПРОЦЕССА WIN API CreateProcess.

#include <iostream>

#include <windows.h>

#include "stdafx.h"

#include "string.h"

int \_tmain(int argc, \_TCHAR\* argv[])

{

STARTUPINFO cif;

PROCESS\_INFORMATION pi;

TCHAR buf[]=TEXT("mspaint.exe");

ZeroMemory(&cif,sizeof(STARTUPINFO));

if(!(CreateProcess(NULL, buf, NULL,NULL,FALSE,0,NULL,NULL,&cif,&pi) ))

std::wcout << "Error " << '\n';

getchar();

return 0;

}

ЗАДАЧА. ОБЪЕДИНИТЬ 1А и 1Б. НАЙТИ ФАЙЛ EXE И ЗАПУСТИТЬ ЕГО НА ВЫПОЛНЕНИЕ.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <windows.h>

#include <tchar.h>

#include "stdafx.h"

void startProcess(WCHAR str[]) {

STARTUPINFO cif;

PROCESS\_INFORMATION pi;

ZeroMemory(&cif, sizeof(STARTUPINFO));

if (!(CreateProcess(NULL, str, NULL,NULL,FALSE,0,NULL,NULL,&cif,&pi))){

printf("%s", "Error \n");}

getchar();

}

void findFileWithExecute() {

WIN32\_FIND\_DATA findFileData;

HANDLE hFind = INVALID\_HANDLE\_VALUE;

TCHAR directorySearch[] = TEXT("C:\\Windows\\system32\\\*");

TCHAR fileSearch[] = TEXT("C:\\Windows\\system32\\mspa\*.exe");

hFind = FindFirstFile(fileSearch, &findFileData);

if (hFind == INVALID\_HANDLE\_VALUE) {

printf("Invalid file handle. Error is %u. \n", GetLastError());

}

else

{

wprintf(L"Found: %s \n", findFileData.cFileName);

while (FindNextFile(hFind, &findFileData)) {

wprintf(L"Found: %s \n", findFileData.cFileName);

}

startProcess(findFileData.cFileName);

FindClose(hFind);

}

}

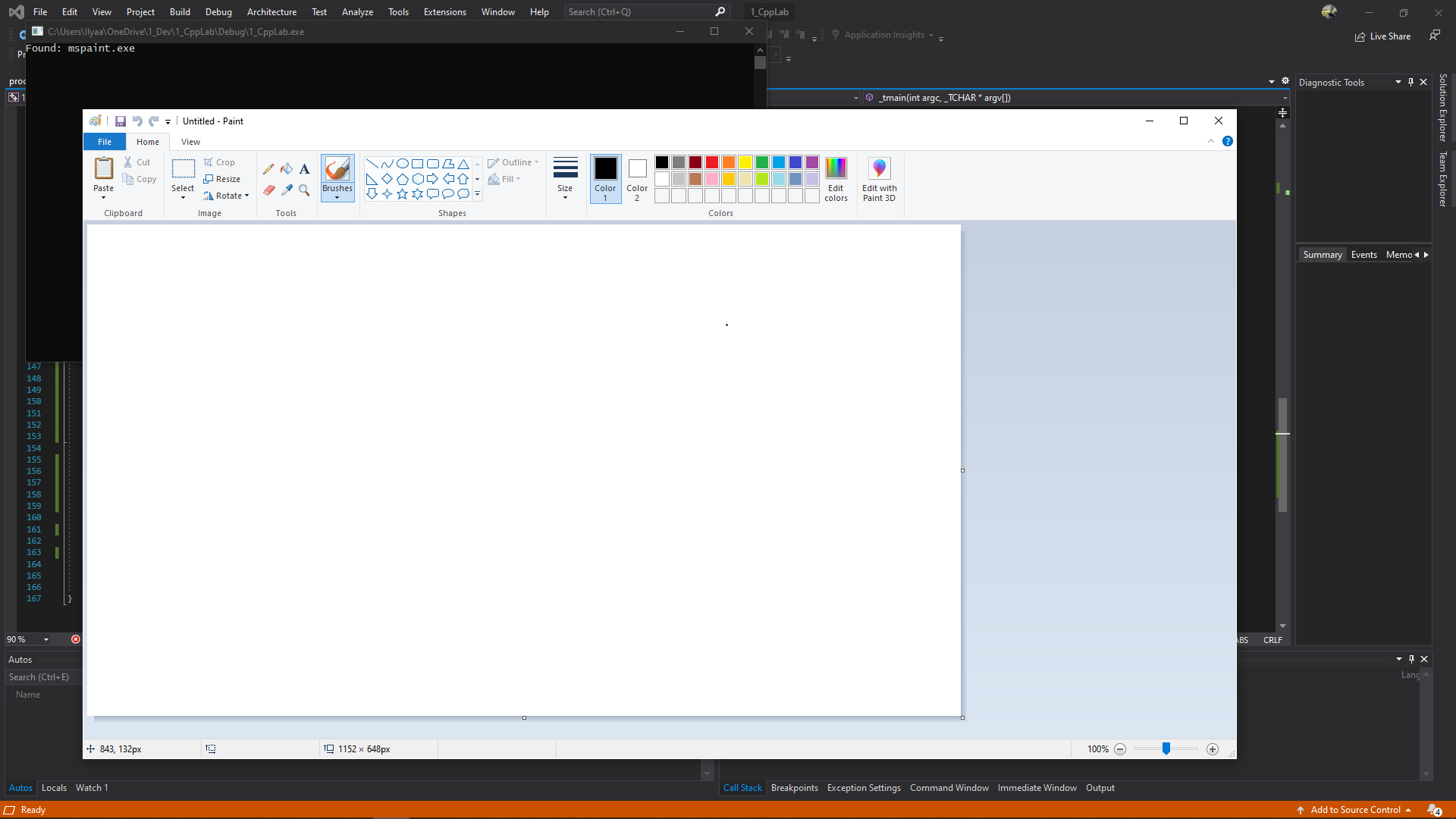
int \_tmain(int argc, \_TCHAR\* argv[]) {

findFileWithExecute();

system("pause");

return 0;

}



1В. Запуск процесса с выводом на консоль его системного номера (заголовка)

#include <iostream>

#include <windows.h>

#include "stdafx.h"

#include "string.h"

int \_tmain(int argc, \_TCHAR\* argv[])

{

DWORD processID;

STARTUPINFO cif;

PROCESS\_INFORMATION pi;

TCHAR buf[] = TEXT("mspaint.exe");

ZeroMemory(&cif, sizeof(STARTUPINFO));

processID=CreateProcess(NULL,buf,NULL,NULL,FALSE,0,NULL,NULL,&cif,&pi);

std::wcout << "Running with PID:" << pi.dwProcessId << '\n';

HANDLE hProcess = GetCurrentProcess(); //OpenProcess( PROCESS\_QUERY\_INFORMATION |

//PROCESS\_VM\_READ,

//FALSE, processID );

// Get the process name.

if (NULL != hProcess){

Sleep(2000);

BOOL result = TerminateProcess(hProcess, 0);

CloseHandle(hProcess);

}

else{

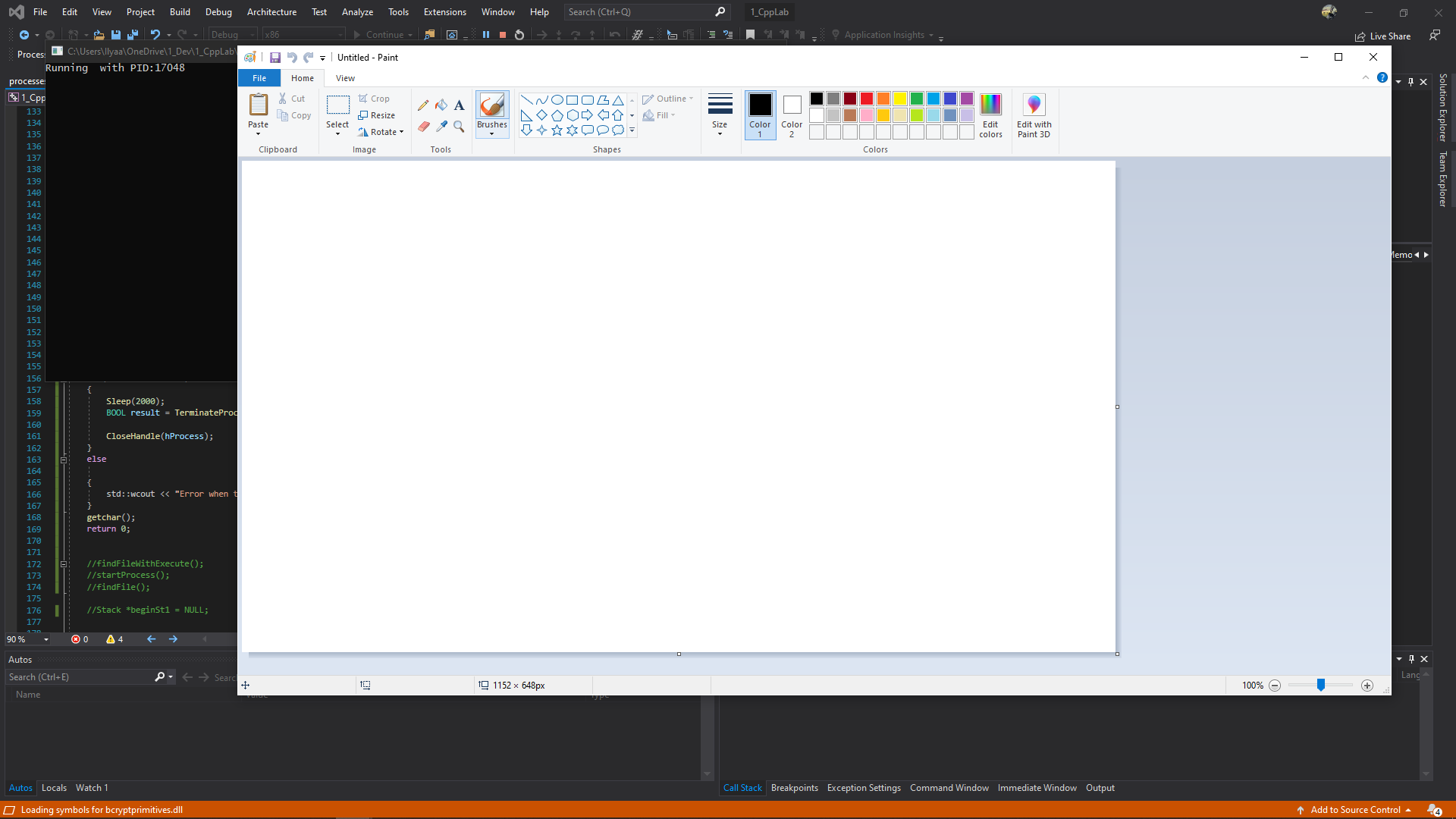
std::wcout << "Error when terminating " << '\n';

}

getchar();

return 0;

}



2. Запуск процесса из консоли с удалением консольного окна через 0.2 секунды.

#include <windows.h>

#include <iostream>

#include "stdafx.h"

#include "string.h"

int \_tmain(int argc, \_TCHAR\* argv[])

{

DWORD processID;

STARTUPINFO cif;

PROCESS\_INFORMATION pi;

TCHAR buf[] = TEXT("mspaint.exe");

ZeroMemory(&cif, sizeof(STARTUPINFO));

if (!(processID = CreateProcess(NULL, buf, NULL, NULL, FALSE, 0, NULL, NULL, &cif, &pi)))

std::wcout << "Running " << '\n';

HANDLE hProcess = GetCurrentProcess(); //получаем pID консольного приложен.

if (NULL != hProcess)

{

Sleep(200);

BOOL result = TerminateProcess(hProcess, 0);

CloseHandle(hProcess);

}

else

{

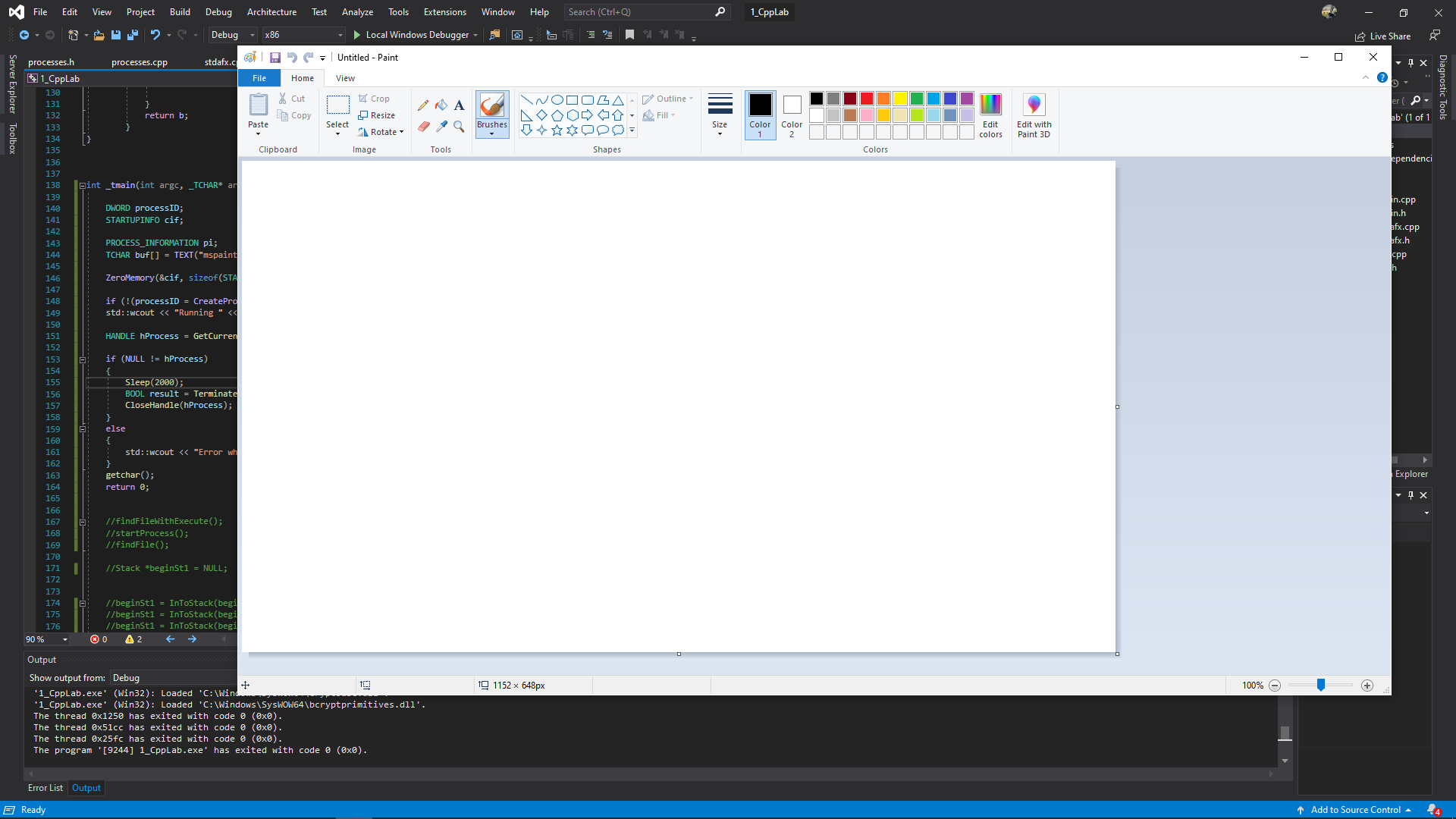
std::wcout << "Error when terminating " << '\n';

}

getchar();

return 0;

}



3. ВЫВОД ИМЕН ВСЕХ ЗАПУЩЕННЫХ ПРОЦЕССОВ

#include "stdafx.h"

#include "windows.h"

#include <tlhelp32.h>

#include "string.h"

#include <iostream>

int \_tmain(int argc, \_TCHAR\* argv[])

{

HANDLE hStdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

HANDLE hSnapshot = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0);

SetConsoleTextAttribute(hStdout, FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_INTENSITY | BACKGROUND\_BLUE);

if (hSnapshot){

PROCESSENTRY32 pe32;

pe32.dwSize = sizeof(PROCESSENTRY32);

if (Process32First(hSnapshot, &pe32)) {

do {

std::wcout << "Running ProcName :" << pe32.szExeFile << '\n';

} while (Process32Next(hSnapshot, &pe32));

}

CloseHandle(hSnapshot);

}

getchar();

return 0;

}



4. ПРОВЕРЯЕТ, ВЫПОЛНЯЕТСЯ ЛИ ПРОЦЕСС

// GetProcessName.cpp : Defines the entry point for the console application.

//

Следующая программа проверяет, запущен ли процесс (exe-файл) на выполнение или нет. Проверку реализует метод IsProcessRun(). Там имя проверяемого процесса задано жестко - GetProcessName.exe. Это имя самой данной с++-программы.

#define STRLEN(x) (sizeof(x)/sizeof(TCHAR) - 1)

#include "stdafx.h"

#include "windows.h"

#include <tlhelp32.h>

#include "string.h"

#include <iostream>

bool AreEqual(const TCHAR\* a, const TCHAR\* b)

{

while (\*a == \*b)

{

if (\*a == \_TEXT('\0'))return true;

a++; b++;

}

return false;

}

bool IsProcessRun()

{

bool RUN;

TCHAR buf[] = TEXT("mspaint.exe");

HANDLE hSnapshot = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0);

PROCESSENTRY32 pe;

pe.dwSize = sizeof(PROCESSENTRY32);

Process32First(hSnapshot, &pe);

if (AreEqual(pe.szExeFile, buf))

{

RUN = true;

return RUN;

}

else{ RUN = false; }

while (Process32Next(hSnapshot, &pe))

{

if (AreEqual(pe.szExeFile, buf))

{

RUN = true;

return RUN;

}

else

RUN = false;

}

return RUN;

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

HANDLE hStdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hStdout, FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_INTENSITY | BACKGROUND\_BLUE);

if (IsProcessRun())

{

std::wcout << "Running" << '\n';

}

else

{

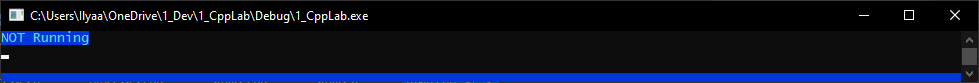
std::wcout << "NOT Running" << '\n';

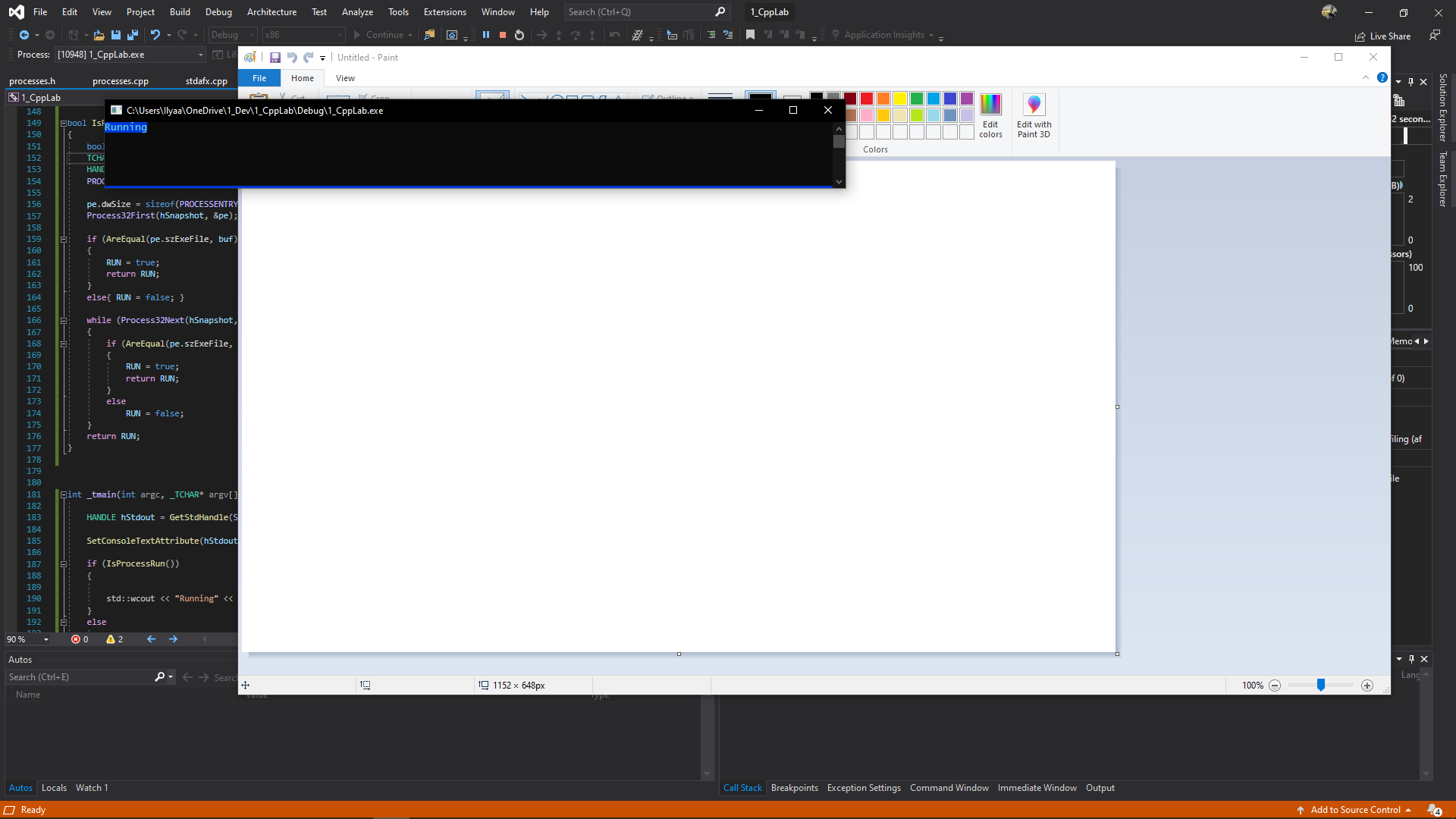
}

getchar();

return 0;

}





5. Собственная программа которая отыскивает exe-файл по части имени, запускает его на выполнение, затем проверяет, выполняется ли он, и выдает соответствующее сообщение

bool AreEqual(const TCHAR\* a, const TCHAR\* b)

{

while (\*a == \*b){

if (\*a == \_TEXT('\0'))return true;

a++; b++;

}

return false;

}

bool IsProcessRun()

{

bool RUN;

TCHAR buf[] = TEXT("mspaint.exe");

HANDLE hSnapshot = CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS, 0);

PROCESSENTRY32 pe;

pe.dwSize = sizeof(PROCESSENTRY32);

Process32First(hSnapshot, &pe);

if (AreEqual(pe.szExeFile, buf)){

RUN = true;

return RUN;

}

else { RUN = false; }

while (Process32Next(hSnapshot, &pe)){

if (AreEqual(pe.szExeFile, buf)){

RUN = true;

return RUN;

}

else RUN = false;

}

return RUN;

}

void startProcess(WCHAR str[]){

STARTUPINFO cif;

PROCESS\_INFORMATION pi;

ZeroMemory(&cif, sizeof(STARTUPINFO));

if (!(CreateProcess(NULL, str, NULL,NULL,FALSE,0,NULL,NULL,&cif,&pi))){

printf("%s", "Error \n");}

HANDLE hProcess = GetCurrentProcess(); //получаем pID консольного приложен.

if (IsProcessRun()){

std::wcout << "Running" << '\n';

}

else{std::wcout << "NOT Running" << '\n';}

getchar();

}

void findFileWithExecute() {

WIN32\_FIND\_DATA findFileData;

HANDLE hFind = INVALID\_HANDLE\_VALUE;

TCHAR directorySearch[] = TEXT("C:\\Windows\\system32\\\*");

TCHAR fileSearch[] = TEXT("C:\\Windows\\system32\\mspa\*.exe");

hFind = FindFirstFile(fileSearch, &findFileData);

if (hFind == INVALID\_HANDLE\_VALUE) {

printf("Invalid file handle. Error is %u. \n", GetLastError());

}

else

{

wprintf(L"Found: %s \n", findFileData.cFileName);

while (FindNextFile(hFind, &findFileData)) {

wprintf(L"Found: %s \n", findFileData.cFileName);

}

startProcess(findFileData.cFileName);

FindClose(hFind);

}

}

