Лабораторная работа №4 Асылбек уулу Бакыта из группы ИСП-308

Процессы и потоки

1) Написать программу, внутри которой:

1. Создать процесс через CreateProcess

2. Дождаться завершения созданного процесса и вывести «код завершения» пользователю (`WaitForSingleObject`)

#include <windows.h>

#include <stdio.h>

#include <tchar.h>

void main(int argc, TCHAR\* argv[])

{

STARTUPINFO si;

PROCESS\_INFORMATION pi;

ZeroMemory(&si, sizeof(si));

si.cb = sizeof(si);

ZeroMemory(&pi, sizeof(pi));

if (argc != 2)

{

printf("Usage: %s [cmdline]\n", argv[1]);

return;

}

// Запустите дочерний процесс.

if (!CreateProcess(NULL, // Нет имени модуля (используйте командную строку)

argv[1], // Командная строка

NULL, // Дескриптор процесса не наследуется

NULL, // Дескриптор потока не наследуется

FALSE, // Установите для наследования дескриптора значение FALSE

0, // Нет флагов создания

NULL, // Используйте родительский блок окружения

NULL, // Используйте начальный каталог родителя

&si, // Указатель на структуру STARTUPINFO

&pi) // Указатель на структуру PROCESS\_INFORMATION

)

{

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

return;

}

// Подождите, пока дочерний процесс завершит работу.

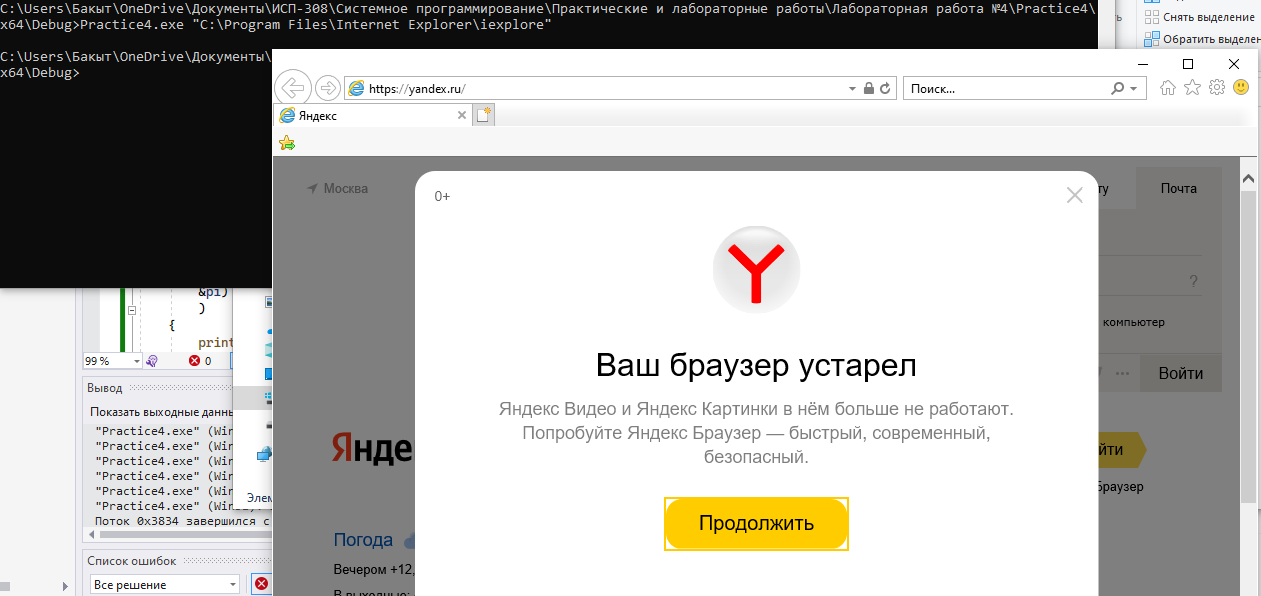
WaitForSingleObject(pi.hProcess, INFINITE);

// Закройте дескрипторы процесса и потока.

CloseHandle(pi.hProcess);

CloseHandle(pi.hThread);

}



2) Написать программу, внутри которой:

1. Перебираются всё процессы (`EnumProcesses`)

2. Пользователю выводится идентификатор и имя для каждого процесса (либо GetProcessImageFileName, либо `GetModuleBaseName`)

#include <windows.h>

#include <stdio.h>

#include <tchar.h>

#include <psapi.h>

// Чтобы обеспечить правильное разрешение символов, добавьте Psapi.lib в TARGETLIBS и скомпилируйте с -DPS API\_VERSION=1

void PrintProcessNameAndID(DWORD processID)

{

TCHAR szProcessName[MAX\_PATH] = TEXT("<unknown>");

// Получить управляемый процесс

HANDLE hProcess = OpenProcess(PROCESS\_QUERY\_INFORMATION | PROCESS\_VM\_READ, FALSE, processID);

// Получить название процесса.

if (NULL != hProcess)

{

HMODULE hMod;

DWORD cbNeeded;

if (EnumProcessModules(hProcess, &hMod, sizeof(hMod), &cbNeeded))

{

GetModuleBaseName(hProcess, hMod, szProcessName, sizeof(szProcessName) / sizeof(TCHAR));

}

}

\_tprintf(TEXT("%s (PID: %u)\n"), szProcessName, processID);// Вывести название процесса и идентификатор

CloseHandle(hProcess);// Отпустите управляемый процесса.

}

int main(void)

{

DWORD aProcesses[1024], cbNeeded, cProcesses;

unsigned int i;

if (!EnumProcesses(aProcesses, sizeof(aProcesses), &cbNeeded))// Получите список идентификаторов процесса.

{

return 1;

}

cProcesses = cbNeeded / sizeof(DWORD);// Подсчитайте, сколько идентификаторов процесса было возвращено.

for (i = 0; i < cProcesses; i++)

{

if (aProcesses[i] != 0)

{

PrintProcessNameAndID(aProcesses[i]);// Выведите название и идентификатор процесса для каждого процесса.

}

}

return 0;

}

