Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики»

Лабораторная №2

«Процессы и потоки»

Выполнила студентка

группы БВТ1501

Козлова Анна

8 вариант

МОСКВА, 2018

**Цель работы:**

Получение практических навыков при использовании Win32 API для исследования процессов.

**Код:**

#include <windows.h>

#include <iostream>

#include <iomanip>

#include <conio.h>

#include <sstream>

#include <tlhelp32.h>

#define TCHAR wchar\_t

#define cout wcout

#define cin wcin

#define TEXT(p) L##p

using namespace std;

//идти от конца строки, когда встретится \,

//запомнить позицию \ и обрезать от нее до конца

wstring Name(const wchar\_t\* path)

{

wstring name = path;

int i;

for (i = size(name) - 1;i >= 0;i--)

if (name[i] == '\\') break;

name = name.substr(i + 1, size(name) - i);

return name;

}

int main() {

setlocale(LC\_ALL, "Russian");

TCHAR buffer[512];

DWORD size = 512;

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

//для текущего процесса

if (!GetModuleFileName(GetModuleHandle(NULL), buffer, size))

cout << TEXT("error of GetModuleFileName");

else {

SetConsoleTextAttribute(console, 11);

cout << TEXT("Задаем - дескриптор модуля\n");

SetConsoleTextAttribute(console, 15);

cout << TEXT("Полное имя модуля: \t") << buffer << endl;

cout << TEXT("Имя файла: \t\t") << Name(buffer) << endl;

}

if (!GetModuleHandle(TEXT("Lab1OS.exe")))

cout << TEXT("error of GetModuleHandle");

else {

HMODULE hModule = GetModuleHandle(TEXT("Lab1OS.exe"));

SetConsoleTextAttribute(console, 11);

cout << TEXT("Задаем - имя модуля\n");

SetConsoleTextAttribute(console, 15);

cout << TEXT("Дескриптор модуля: \t") << UINT(hModule) << endl;

GetModuleFileName(hModule, buffer, size);

cout << TEXT("Полное имя модуля: \t") << buffer << endl;

}

if (!GetModuleHandle(TEXT("Lab1OS.exe")))

cout << TEXT("error of GetModuleHandle");

else {

HMODULE hModule = GetModuleHandle(

TEXT("C://Users//Spanri//documents//visual studio 2017//Projects//Debug//Lab1OS.exe"));

SetConsoleTextAttribute(console, 11);

cout << TEXT("Задаем - полное имя модуля\n");

SetConsoleTextAttribute(console, 15);

cout << TEXT("Дескриптор модуля: \t") << UINT(hModule) << endl;

GetModuleFileName(hModule, buffer, size);

cout << TEXT("Имя модуля: \t\t") << Name(buffer) << endl;

}

//Для библиотеки kernel

SetConsoleTextAttribute(console, 11);

cout << TEXT("Для kernel32.dll\n");

if (!GetModuleHandle(TEXT("kernel32.dll")))

cout << TEXT("error of GetModuleHandle");

else {

HMODULE hModule = GetModuleHandle(TEXT("kernel32.dll"));

cout << TEXT("Задаем - имя модуля\n");

SetConsoleTextAttribute(console, 15);

cout << TEXT("Дескриптор модуля: \t") << UINT(hModule) << endl;

GetModuleFileName(hModule, buffer, size);

cout << TEXT("Полное имя модуля: \t") << buffer << endl;

}

//вторая часть

SetConsoleTextAttribute(console, 11);

cout << TEXT("Вторая часть:\n");

SetConsoleTextAttribute(console, 15);

DWORD id = GetCurrentProcessId();

cout << TEXT("Идентификатор текущего процесса: \t") << UINT(id) << endl;

HANDLE ps = GetCurrentProcess();

cout << TEXT("Псевдодескриптор текущего процесса: \t") << UINT(ps) << endl;

HANDLE buf1;

DuplicateHandle(ps, ps, ps, &buf1, 0, FALSE, DUPLICATE\_SAME\_ACCESS);

cout << TEXT("Дескриптор с DublicateHandle: \t\t") << UINT(buf1) << endl;

HANDLE buf2 = OpenProcess(PROCESS\_ALL\_ACCESS, FALSE, id);

cout << TEXT("Дескриптор с OpenProcess: \t\t") << UINT(buf2) << endl;

if (CloseHandle(buf1))

cout << TEXT("Закрыли DublicateHandle.") << endl;

else cout << TEXT("DublicateHandle не закрывается...") << endl;

if (CloseHandle(buf2))

cout << TEXT("Закрыли OpenProcess.") << endl;

else cout << TEXT("OpenProcess не закрывается...") << endl;

SetConsoleTextAttribute(console, 11);

cout << TEXT("Все процессы, потоки, модули и их свойства в системе:\n");

cout << setiosflags(ios::left) << setw(33) << TEXT("Имя процесса")

<< setw(10) << TEXT("PID")

<< setw(15) << TEXT("Число потоков")

<< setw(15) << TEXT("Приоритет") << endl;

SetConsoleTextAttribute(console, 15);

//третья часть

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

if (all == NULL) {

return 0;

cout << TEXT("Всё плохо.");

}

PROCESSENTRY32 proc;

proc.dwSize = sizeof(PROCESSENTRY32);

if (Process32First(all, &proc)) {

do {

cout << setw(33) << proc.szExeFile;

cout << setw(10) << proc.th32ProcessID

<< setw(15) << proc.cntThreads

<< setw(15) << proc.pcPriClassBase << endl;

} while (Process32Next(all, &proc));

}

DWORD PID;

SetConsoleTextAttribute(console, 11);

cout << TEXT("Введите PID процесса: ");

cin >> PID;

HANDLE thread = CreateToolhelp32Snapshot(TH32CS\_SNAPTHREAD, PID);

THREADENTRY32 thr;

thr.dwSize = sizeof(THREADENTRY32);

if (Thread32First(thread, &thr))

{

cout << setw(10) << TEXT("PID")

<< setw(10) << TEXT("TID")

<< setw(10) << TEXT("Приоритет") << endl;

SetConsoleTextAttribute(console, 15);

do

{

cout << setw(10) << thr.th32OwnerProcessID

<< setw(10) << thr.th32ThreadID

<< setw(10) << thr.tpBasePri << endl;

} while (Thread32Next(thread, &thr));

cout << endl;

}

HANDLE module = CreateToolhelp32Snapshot(TH32CS\_SNAPMODULE, PID);

MODULEENTRY32 mod;

mod.dwSize = sizeof(MODULEENTRY32);

if (Module32First(module, &mod))

{

SetConsoleTextAttribute(console, 11);

cout << setw(25) << TEXT("Модуль процесса")

<< setw(10) << TEXT("Дескриптор")

<< endl;

SetConsoleTextAttribute(console, 15);

do

{

cout << setw(25) << Name(mod.szExePath)

<< setw(10) << (UINT)mod.hModule << endl;

} while (Module32Next(module, &mod));

cout << endl;

}

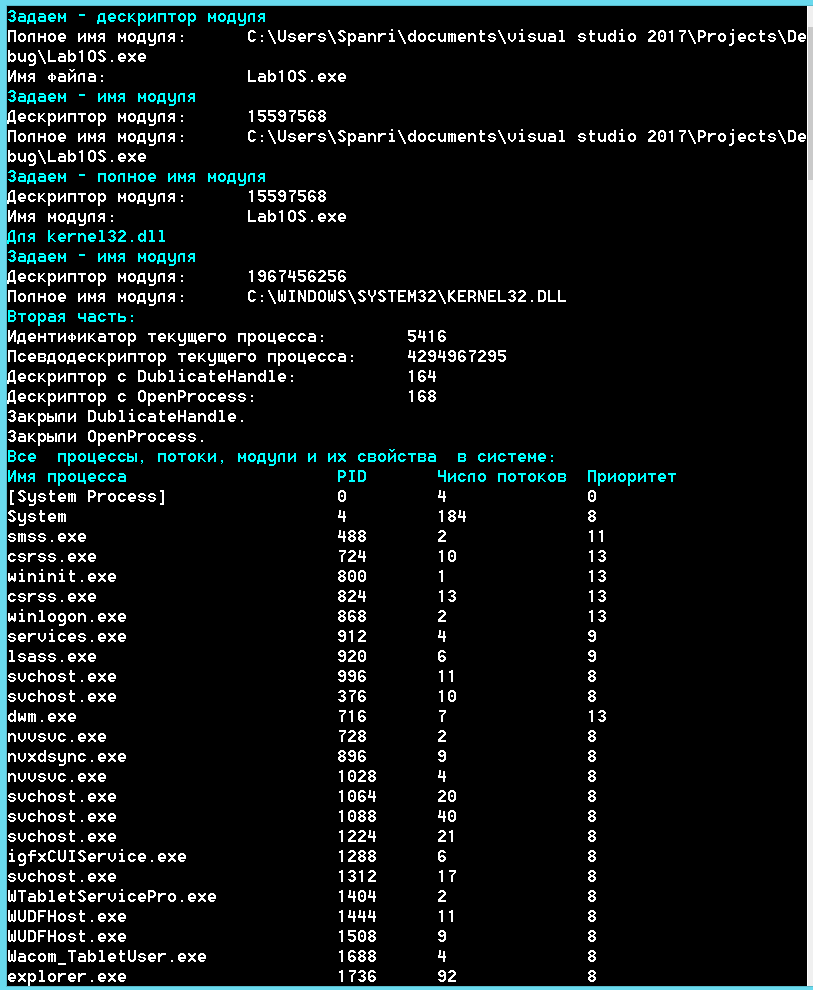
SetConsoleTextAttribute(console, 11);

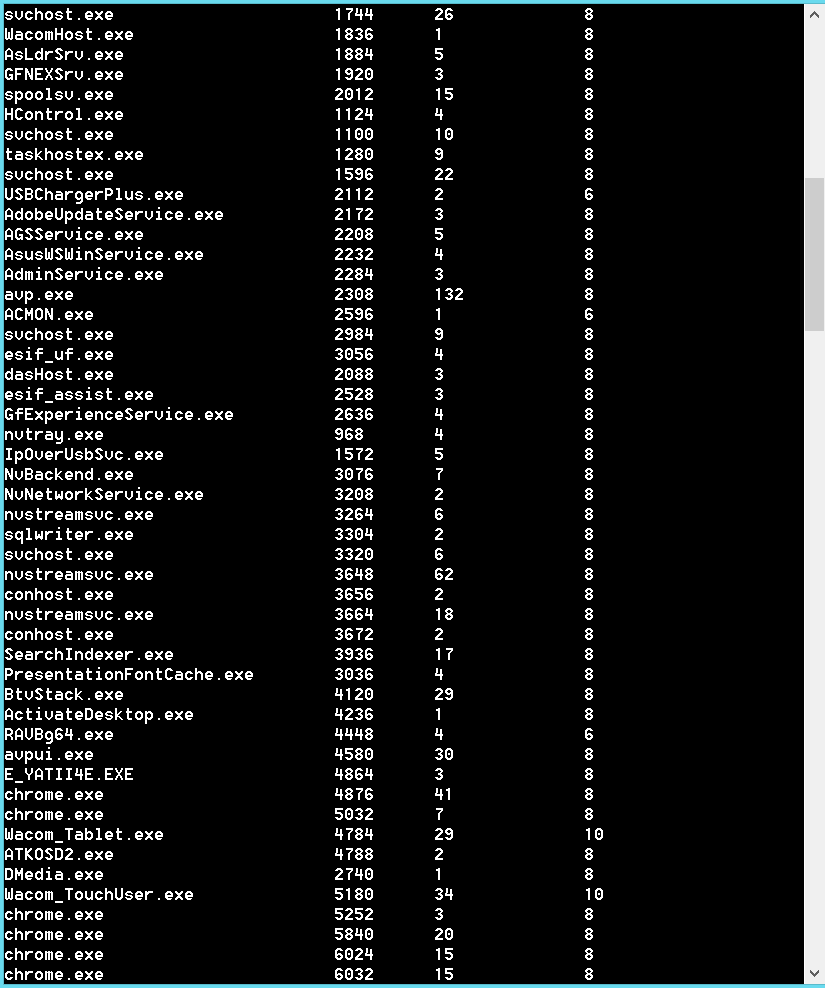
system("pause");

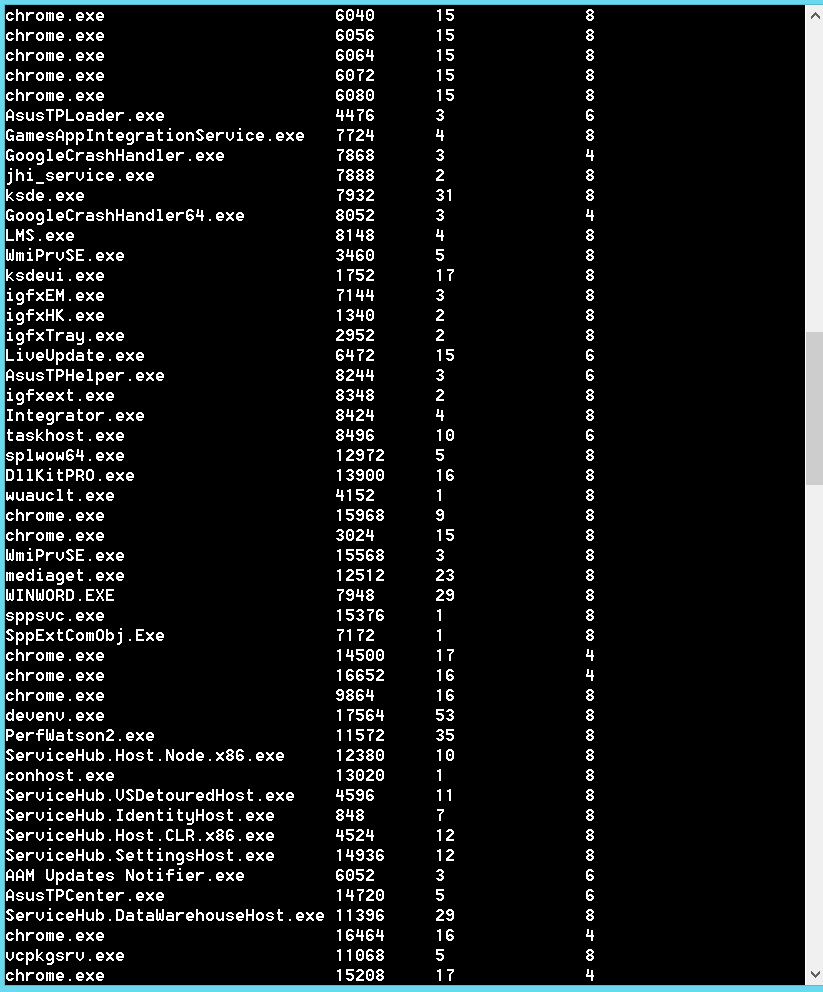
return 0;

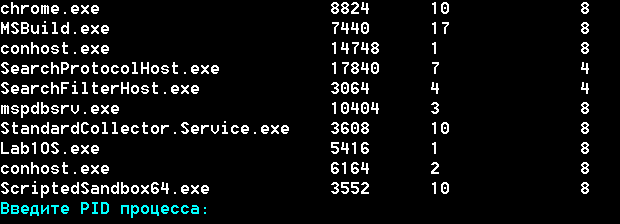
}

**Пример работы:**

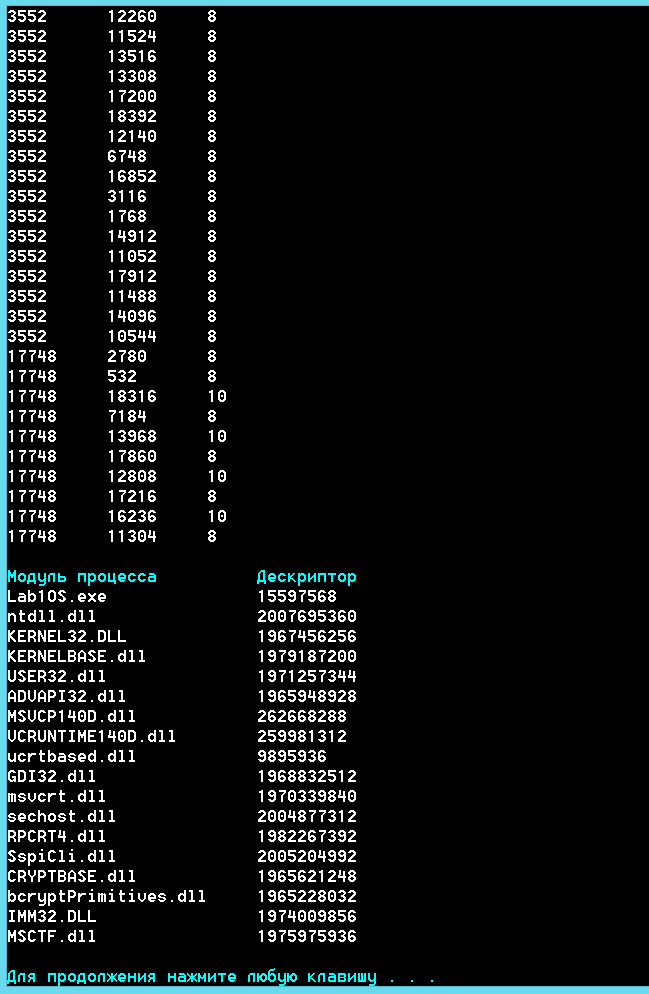








Вводим 5146 (сначала – все потоки, их PID, TID, приоритет, не вместились все в вывод).



**Вывод:**

В результате проделанной работы научились работать с процессами и потоками.