**БУ ВО Ханты-Мансийского автономного округа – Югры**

**«Сургутский государственный университет»**

**Политехнический институт**

**Кафедра автоматики и компьютерных систем**

**ОТЧЁТ**

**по лабораторной работе №2**

**по дисциплине: «Операционные системы»**

**Выполнил: студент группы №609-21**

**Шумилов И. Д.**

**Принял: старший преподаватель кафедры АиКС**

**Кривицкая М. А.**

**Сургут**

**2024г.**

Цель работы: изучить функции и структуры данных Windows API, используемые для создания и зазвершения процессов, исследовать возможности получения доступа к управлению процессами.

**Задание на лабораторную работу:**

Разработать и отладить прогрпмму, выполняющую следующие функции:

1. Создание процесса с помощью функции CreateProcess.
2. Определение дескриптора созданного процесса по идентификатору, определенному соответствующим полем структуры PROCESS\_INFORMATION (сравнить это значение со значением идентификатора из “Диспетчера процессов”).
3. Проверка уникальности дескрипторов процессов.
4. Получение значений временных характеристик процесса.
5. Завершение процесса и закрытие открытых раннее дескрипторов.
6. Проверка успешности выполнения операций над процессами.

**Листинг 1. Код программы на С++:**

**#include <Windows.h>**

**#include <iostream>**

**#include <string>**

**int DoCreateProcess();**

**int main(int argc, char\*\* argv) {**

**if (!DoCreateProcess()) {**

**std::cout << "Was unable to create process" << std::endl;**

**return 1;**

**}**

**return 0;**

**}**

**int DoCreateProcess() {**

**STARTUPINFO si;**

**PROCESS\_INFORMATION pi;**

**ZeroMemory(&si, sizeof(si));**

**ZeroMemory(&pi, sizeof(pi));**

**std::wstring CommandLine(L"notepad.exe");**

**LPWSTR lpwCmdlLine = &CommandLine[0];**

**BOOL b = NULL;**

**b = CreateProcessW(NULL, lpwCmdlLine, NULL, NULL, FALSE, 0, NULL, NULL, &si, &pi);**

**if (b) {**

**std::cout << "Process created" << std::endl;**

**std::cout << "Process Id: " << GetProcessId(pi.hProcess) << std::endl;**

**HANDLE processHandle = OpenProcess(PROCESS\_ALL\_ACCESS, FALSE, pi.dwProcessId);**

**std::cout << "Process handle opened" << std::endl;**

**FILETIME creationTime, exitTime, kernelTime, userTime;**

**SYSTEMTIME sCreationTime, sExitTime, sKernelTime, sUserTime;**

**GetProcessTimes(processHandle, &creationTime, &exitTime, &kernelTime, &userTime);**

**std::cout << "Got process times (1/4)" << std::endl;**

**FileTimeToSystemTime(&creationTime, &sCreationTime);**

**std::cout << "Creation time: " << sCreationTime.wHour << ":" << sCreationTime.wMinute << ":" << sCreationTime.wSecond << std::endl;**

**std::cout << "Close notepad when ready" << std::endl;**

**WaitForSingleObject(processHandle, INFINITE);**

**GetProcessTimes(processHandle, &creationTime, &exitTime, &kernelTime, &userTime);**

**std::cout << "Got process times (4/4)" << std::endl;**

**FileTimeToSystemTime(&exitTime, &sExitTime);**

**std::cout << "Exit time: " << sExitTime.wHour << ":" << sExitTime.wMinute << ":" << sExitTime.wSecond << std::endl;**

**std::cout << "Kernel time: " << kernelTime.dwLowDateTime / 100000.0 << " crocodiles" << std::endl;**

**std::cout << "User time: " << userTime.dwLowDateTime / 100000.0 << " crocodiles" << std::endl;**

**std::cout << "Press any key to terminate process and continue..." << std::endl;**

**system("pause > nul");**

**TerminateProcess(processHandle, 1);**

**CloseHandle(processHandle);**

**CloseHandle(pi.hProcess);**

**CloseHandle(pi.hThread);**

**} else {**

**return 0;**

**}**

**return 1;**

**}**

**Пояснение к программе:**

Получение дескриптора процесса выполняется с помощью функции Windows API OpenProcess, которая возвращает HANDLE (т.е. сам дескриптор процесса). В качестве аргумента OpenProcess принимает идентификатор процесса, который извлекается из структуры PROCESS\_INFORMATION и уровень доступа (уровни доступа приведены на Рис. 1). После чего уже с помощью самого идентификатора можно получить временные характеристики и завершить процесс. После получения временных характеристик и завершения процесса происходит закрытие открытых ранее дескрипторов.

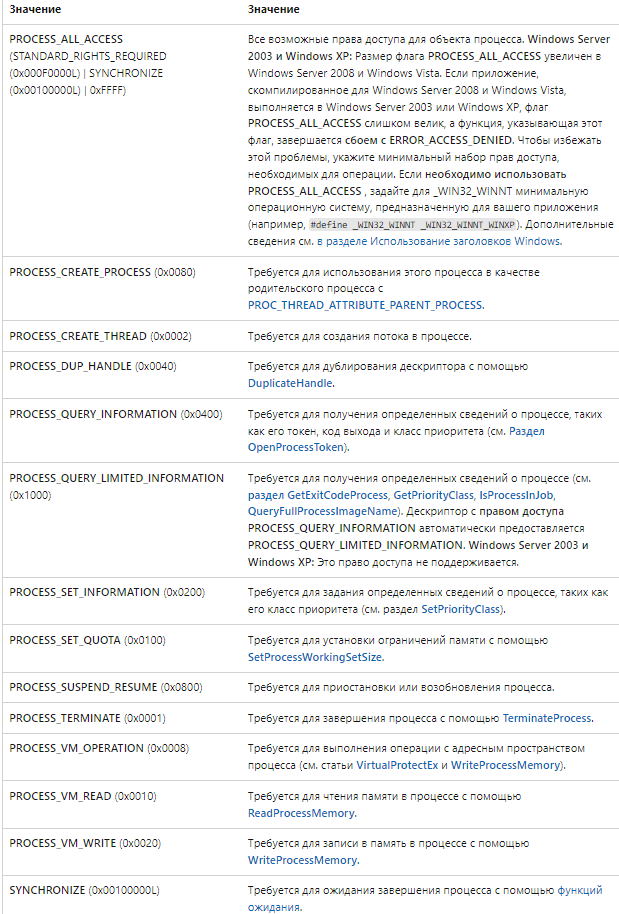


Рис. 1 - таблица уровней доступа к процессу.

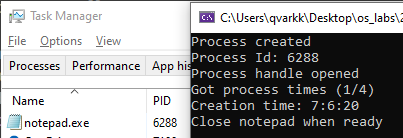


Рис. 2 - проверка совпадения идентификатора процесса.

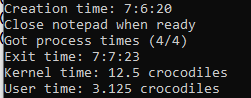


Рис. 3 - получение временных характеристик процесса.

**Вывод:** были изучены некоторые функции и структуры данных библиотеки Windows в С++, используемые для создания и завершения процессов. Также были иследованы возможности получения доступа к управлению процессами.