МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Полоцкий государственный университет»

Факультет информационных технологий Кафедра технологий программирования

# Лабораторная работа №11

по дисциплине: «Системное программирование»

на тему:

«Отладка программ в режиме ядра. Системные механизмы ядра»

Вариант №12

Выполнил Студент группы 21-ИТ-1

Чиникайло А.П.

Проверил Преподаватель

Сергеев М.А.

Полоцк 202

**Задания**

1. В соответствии с заданием определить необходимые системные функции;

2. Создать дескриптор на файл используя функцию CreateFile;

3. Получить размер файла с помощью функции GetFileSize, используя полученный дескриптор;

4. Создать файловое отображение файла, сохранить полученный на него дескриптор, используя дескриптор на файл;

5. Отобразить объект файлового отображения в адресное пространство текущего процесса с помощью функции MapViewOfFile, которая возвращает начальный адрес отображения в памяти;

6. Скопировать в символьный массив отображенные данные с помощью функции CopyMemory, где первым параметром является указатель на временный массив, вторым - указатель на отображение файла в памяти;

7. В соответствии с заданием провести необходимые операции с данными, полученными из файлового отображения;

8. Скопировать результат из временного массива обратно в отображенную память с помощью функции CopyMemory;

9. Освободить память с помощью функции UnmapViewOfFile;

10. Завершить работу с файловым отображением и файлом с помощью функции CloseHandle.

**Задание 1:**

#include <Windows.h>

#include <iostream>

#include <string>

using namespace std;

void LogAdd(const string& logMessage, HANDLE logFile) {

DWORD bytesWritten;

WriteFile(logFile, logMessage.c\_str(), static\_cast<DWORD>(logMessage.length()), &bytesWritten, nullptr);

}

int main() {

HANDLE logFile = CreateFile(L"FILE.log", GENERIC\_WRITE, 0, nullptr, CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, nullptr);

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

cout << "File" << endl;

return 1;

}

LogAdd("Log file was created!\n", logFile);

STARTUPINFO startupInfo = {};

PROCESS\_INFORMATION notepadProcessInfo = {};

if (CreateProcess(L"C:\\Windows\\notepad.exe", nullptr, nullptr, nullptr, FALSE, 0, nullptr, nullptr, &startupInfo, &notepadProcessInfo)) {

LogAdd("Notepad proccess was created!\n", logFile);

HANDLE notepadDescriptor = notepadProcessInfo.hProcess;

HANDLE logFileNew = logFile;

LogAdd("Main proccess gets decriptor.(loaded by new file desc)\n", logFileNew);

cout << "Choose type of priority: \n1. IDLE\_PRIORITY\_CLASS\n2. BELOW\_NORMAL\_PRIORITY\_CLASS\n3. ABOVE\_NORMAL\_PRIORITY\_CLASS\n4. HIGH\_PRIORITY\_CLASS\n\n";

int switch\_on;

cin >> switch\_on;

switch (switch\_on)

{

case 1:

SetPriorityClass(notepadDescriptor, IDLE\_PRIORITY\_CLASS);

cout << "Priority #" << switch\_on << " was set!\n";

LogAdd("Priority IDLE\_PRIORITY\_CLASS was set!(loaded by file desc)\n", logFileNew);

break;

case 2:

SetPriorityClass(notepadDescriptor, BELOW\_NORMAL\_PRIORITY\_CLASS);

cout << "Priority BELOW\_NORMAL\_PRIORITY\_CLASS was set!\n";

LogAdd("Priority BELOW\_NORMAL\_PRIORITY\_CLASS was set!(loaded by file desc)\n", logFileNew);

break;

case 3:

SetPriorityClass(notepadDescriptor, ABOVE\_NORMAL\_PRIORITY\_CLASS);

cout << "Priority ABOVE\_NORMAL\_PRIORITY\_CLASS was set!\n";

LogAdd("Priority ABOVE\_NORMAL\_PRIORITY\_CLASS was set!(loaded by file desc)\n", logFileNew);

break;

case 4:

SetPriorityClass(notepadDescriptor, HIGH\_PRIORITY\_CLASS);

cout << "Priority HIGH\_PRIORITY\_CLASS was set!\n";

LogAdd("Priority HIGH\_PRIORITY\_CLASS was set!(loaded by file desc)\n", logFileNew);

break;

default:

cout << "Priority wasnt set!\n";

LogAdd("Priority wasnt set!\n", logFileNew);

break;

}

LogAdd("Notepad2 descriptor was closed.(loaded by file desc)\n", logFileNew);

LogAdd("Notepad file descriptor was closed.\n", logFile);

CloseHandle(notepadDescriptor);

CloseHandle(logFile);

}

else {

cout << "Notepad" << endl;

CloseHandle(logFile);

return 1;

}

return 0;

}

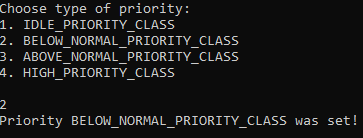


Рисунок 1 – Выбор приоритета процесса

**Задание 2:**

#include <Windows.h>

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

int main() {

// 2

HANDLE logFile = CreateFile(L"FILE2.log", GENERIC\_READ | GENERIC\_WRITE, 0, nullptr, CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, nullptr);

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

cout << "File" << endl;

return 1;

}

cout << "File descriptor: " << logFile << endl;

long long int num;

cout << "Enter numbers to sort (without spaces): ";

cin >> num;

wstring dataToWrite = to\_wstring(num);

DWORD bytesWritten;

WriteFile(logFile, dataToWrite.c\_str(), dataToWrite.size() \* sizeof(wchar\_t), &bytesWritten, nullptr);

DWORD fileSize = GetFileSize(logFile, nullptr);

cout << "File size: " << fileSize << endl;

// 3

HANDLE fileMapping = CreateFileMapping(logFile, nullptr, PAGE\_READWRITE, 0, 0, nullptr);

if (fileMapping == nullptr) {

cout << "Mapping" << endl;

CloseHandle(logFile);

return 1;

}

cout << "File mapping: " << fileMapping << endl;

// 4

LPVOID mapView = MapViewOfFile(fileMapping, FILE\_MAP\_READ | FILE\_MAP\_WRITE, 0, 0, 0);

if (mapView == nullptr) {

cout << "MapView" << endl;

CloseHandle(fileMapping);

CloseHandle(logFile);

return 1;

}

cout << "Map view: " << mapView << endl;

wstring dataObratno(reinterpret\_cast<wchar\_t\*>(mapView));

sort(dataObratno.begin(), dataObratno.end(), less<wchar\_t>()); // greater

CopyMemory(mapView, dataObratno.c\_str(), fileSize);

// 5

UnmapViewOfFile(mapView);

CloseHandle(fileMapping);

CloseHandle(logFile);

return 0;

}

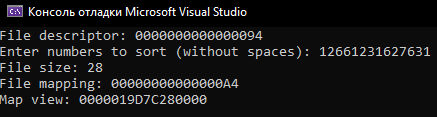


Рисунок 1 – Сортировка цифр

**Вывод:** в результате выполнения данной лабораторной работы были изучены способы наследования дескрипторов и построены приложение-родитель и дочернее приложение, демонстрирующие наследование дескрипторов.