**Лабораторная работа №7**

**Управление файлами II**

**Цель работы:** Получение практических навыков в работе с файлами отображенными в память.

Перед выполнением задания прочитайте его полностью. Схематично представьте себе, что от вас требуется и только после этого выполняйте.

**Темы для предварительного изучения:**

* Объект «section» ([ссылка](https://learn.microsoft.com/en-us/windows-hardware/drivers/kernel/section-objects-and-views));
* FileMapping ([ссылка](https://learn.microsoft.com/en-us/windows/win32/memory/file-mapping)).

**Постановка задачи для Windows:**

**Приложение Lab-07a:**

Разработайте консольное приложение **Lab-07** дляработы со списком студентов, доступ к которому будет осуществляться через механизм FileMapping.

По сути, с точки зрения программы список студентов будет представлять собой массив структур следующего вида:

struct Student

{

char Name[64];//Имя

char Surname[128];//Фамилия

unsigned char Course; //Курс

unsigned char Group; //Группа

char ID[8]; //Номер студенческого

};

В приложении создайте следующие функции:

* **OpenMapping**(LPSTR filePath, LPINT listSize);
* **AddRow**(HANDLE hFileMapping, struct Student row, INT pos);
* **RemRow**(HANDLE hFileMapping, INT pos);
* **PrintRow**(HANDLE hFileMapping, INT pos);
* **PrintRows(**HANDLE hFileMapping);
* **CloseMapping**(HANDLE hFileMapping).

Хранение дескриптора маппинга, с которым работает процесс, а также всех необходимых буферов для данных осуществляется на глобальном уровне (глобальные переменные). Взаимодействие с маппингом происходит через упомянутые ранее функции. Вызовы функций приложение обеспечивает через консоль в виде меню. Данные для функций вводятся с консоли (дескрипторы не вводятся с консоли, а берутся из глобальной переменной). Предусмотреть обработку ошибок ввода данных.

Поведение функций должно быть следующим:

**OpenMapping –** должнапроизводить попытку открытия существующего маппинга для заданного файла или создания нового объект маппинга, если он отсутствует в системе, с последующей записью дескриптора маппинга в глобальную переменную. В случае если файл не существует, то возвращает ошибку.

Открытие маппинга должно происходить по имени маппинга (последний параметр функции CreateFileMapping), которое должно быть задано при его создании. В качестве имени маппинга используется параметр **filePath**.

Второй параметр используется только при создании маппинга для задания максимального размера списка студентов. Информация о размере списка должна храниться в самом файле и при открытии маппинга извлекаться из файла для инициализации необходимых буферов хранения.

**AddRow** – должна производить попытку добавления студента в список, при этом:

* Работают обычные правила работы с массивом по индексам;
* При указании отрицательного значения место для вставки вычисляется с конца массива (т. е. -1 – последний элемент массива, -2 – предпоследний элемент и т. д.);
* Если позиция занята или не существует функция должна завершаться с ошибкой;
* Также ошибка должна быть возвращена если глобальная переменная не содержит корректного дескриптора.

**RemRow** – должна производить попытку удаления студента из списка, при этом:

* Правила работы с индексами аналогичны **AddRow**;
* Если позиция пуста или не существует функция должна завершаться с ошибкой;
* Также ошибка должна быть возвращена если глобальная переменная не содержит корректного дескриптора.

**PrintRow** – должна производить попытку вывода студента из списка с заданной позицией на консоль, при этом:

* Правила работы с индексами аналогичны **AddRow**;
* Если позиция пуста или не существует функция должна завершаться с ошибкой;
* Также ошибка должна быть возвращена если глобальная переменная не содержит корректного дескриптора.

**PrintRows –** должна производить попытку вывода всего списка студентов, при этом, если глобальная переменная не содержит корректного дескриптора, то должна быть возвращена ошибка.

**CloseMapping –** должна производить попытку перевода дескриптора из глобальной переменной в недействительное состояние, а в случае если переменная не содержит корректного дескриптора возвращать ошибку. Также данная функция должна освобождать динамически выделенные ресурсы для работы с маппингом.

В моменты работы приложения просмотреть информацию об открытом маппинге в программе Process Explorer. Посмотреть информацию до открытия, после открытия и после закрытия маппинга.

**Постановка задачи для Linux:**

**Приложение Lab-07b:**

Разработайте консольное приложение, являющееся полным аналогом **Lab-07а**.

**Требования к выполненной работе:**

* Разработанные приложения должны быть скомпилированы и протестированы как на Windows, так и на ОС из семейства Linux.
* Для генерации и сборки проекта использовать средство CMake.
* Для компиляции файлов при сборке проекта использовать компилятор Clang.

**Вопросы для контроля:**

1. Что такое файл, отображенный в память?
2. Как работает механизм FileMapping в Windows? В Linux?
3. Какие функции входят в API для работы с файлами, отображенными в память в WinAPI? POSIX?
4. В чем преимущества использования файлов, отображенных в память?

Что такое файл, отображенный в память?

То есть в этом случае файл или его часть рассматривается как набор виртуальных страниц процесса, которые имеют последовательные логические адреса. Файл, отображенный в адресное пространство процесса, называется представлением или видом файла (**file view**). После отображения файла в адресное пространство процесса доступ к виду может осуществляться с помощью указателя, как к обычным данным в адресном пространстве процесса.

Как работает механизм FileMapping в Windows? В Linux?

* открыть файл, который будет отображаться в память
* создать объект ядра, который выполняет отображение файла
* отобразить файл или его часть в адресное пространство процесса
* выполнить необходимую работу с видом файла
* отменить отображение файла
* закрыть объект ядра для отображения файла
* закрыть файл, который отображался в память
* Получить дескриптор файла (обычно с помощью вызова open)
* Передать этот файловый дескриптор вызову mmap в виде аргумента fd

Какие функции входят в API для работы с файлами, отображенными в память в WinAPI? POSIX?

CreateFileMapping, MapViewOfFile, UnmapViewOfFile, FlushViewOfFile.

mmup, munmap, msync

Когда использовать File Mapping:

* Когда вы работаете с большими файлами
* Когда требуется произвольный доступ
* Когда производительность критична
* Когда память ограничена

Когда не использовать File Mapping:

* Когда вы работаете с маленькими файлами
* Когда требуется последовательный доступ
* Частые изменения файла