МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УО «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Индивидуальная практическая работа №2

«Файловая система»

по дисциплине «Системное программирование»

Вариант 1

Студент

ФИНО, ИИТП, 793551, 3 курс И. В. Галкин

Преподаватель С. И. Сиротко

МИНСК 2019

# **Постановка задачи**

С клавиатуры вводится имя файла. Необходимо найти запись для данного файла в корневом каталоге; составить карту его размещения (список кластеров, в которых записано содержимое этого файла); покластерно прочитать его. Содержимое файла вывести на экран или (по выбору) сохранить в файл.

# **Описание целевых компонентов проекта**

**SpSecond.cpp** – файл содержащий основной код программы.

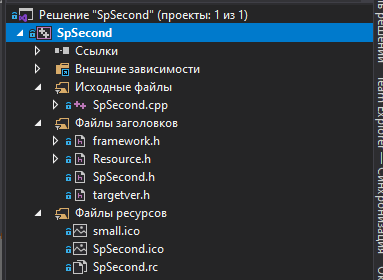


Рисунок 1 Структура проекта

# **Описание выполнения программы**

Входной точкой программы является метод wWinMain, который принимает 4 входных параметра:

* hInst – так называемый дескриптор экземпляра приложения. Операционная система использует этот дескриптор для идентификации исполняемого файла, когда он загружается в память.
* hPrevInstance – использовался в 16-битной Windows. Всегда равен нулю.
* lpCmdLine – содержит аргументы командной строки.
* nCmdShow – указывает на то, будет ли главное окно приложения свернуто, развернуто или показано нормально.

Первым шагом идет регистрация класса окна приложения через метод RegisterWindowClass, в котором мы задаем основные его параметры. Здесь задается указатель на оконную процедуру (определяет основное поведение окна), задается название класса окна и т.д. После этого вызывается метод InitializeWindow, задачей которого является создание окна и отображение его на UI.

Далее идет обработка сообщений операционной системы. В приложении, в основном, ключевую роль играют два сообщения:

* WM\_CREATE – оконная процедура нового окна принимает это сообщение после того, как окно создано, но до того, как окно становится видимым. ОС посылает данное сообщение при вызове CreateWindow. В обработчике сообщения вызываются методы для инициализации меню и инициализируются все элементы управления окна.
* WM\_KEYDOWN – обработчик этого сообщения привязан к полю ввода пути к файлу. При нажатии кнопки «Ввод» (VK\_RETURN) вызывается метод, который выполняет всю работу по анализу выбранного файла.

При запуске приложения пользователю предлагается ввести путь к файлу (Рисунок 2).

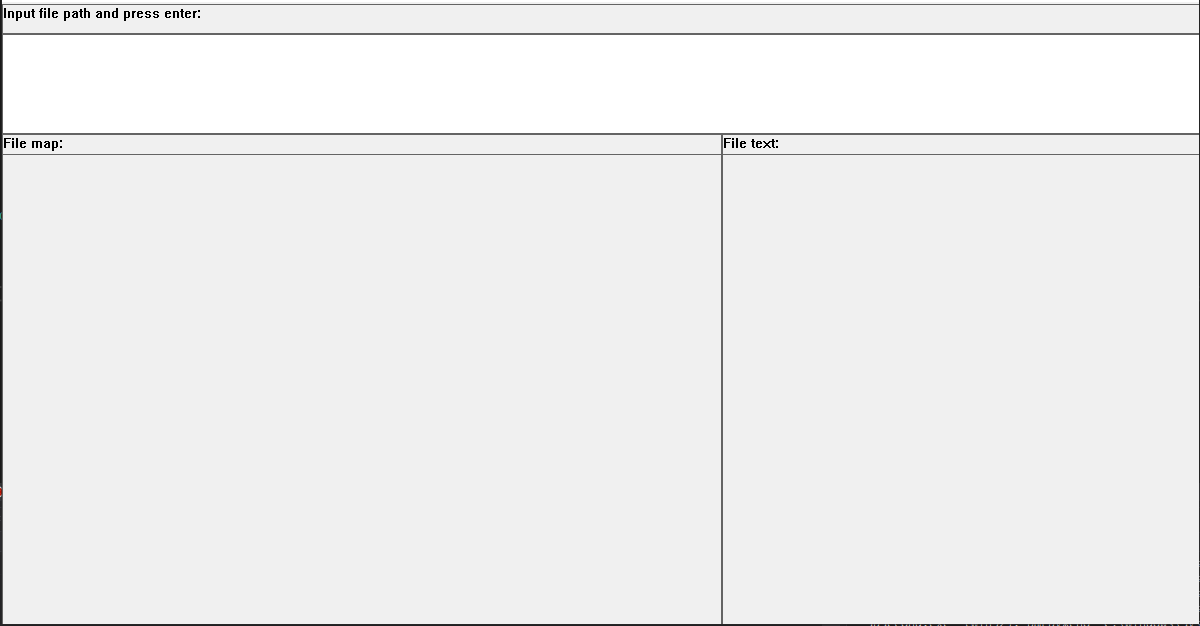


Рисунок 2 Начальный экран программы.

В том случае, если будет введен несуществующий путь, тогда программа выдаст соответствующее сообщение (Рисунок 3).

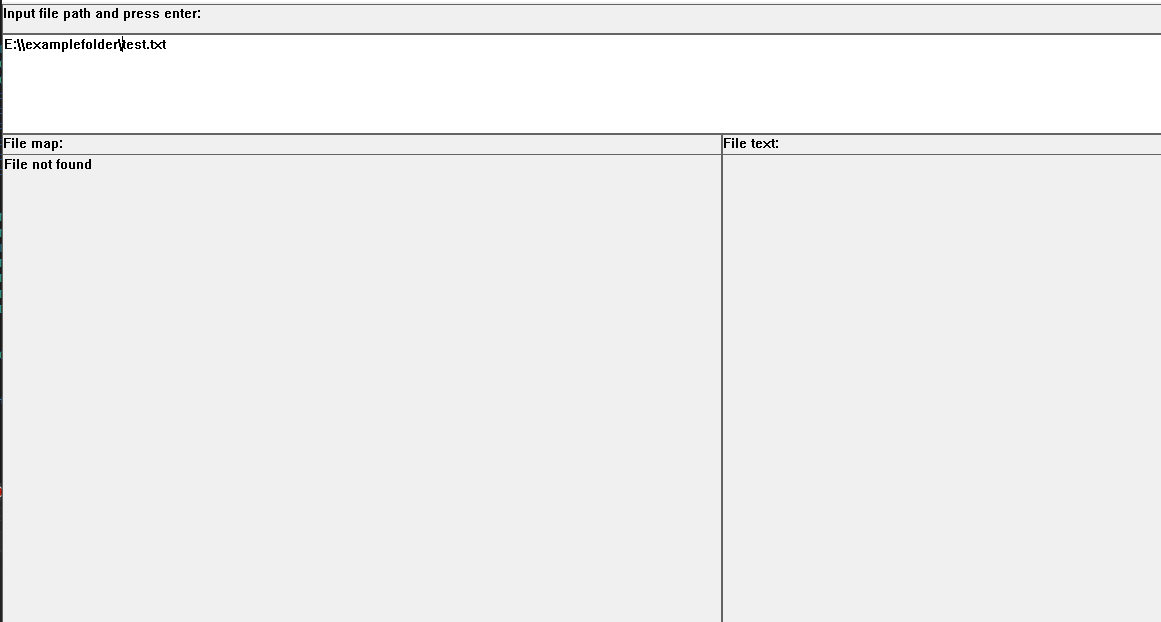


Рисунок 3 Обработка некорректного ввода.

В случае корректного ввода выводится информация о файле и его содержимое (Рисунок 4).



Рисунок 4 Обработка файла.

При вводе пути к файлу, программа пытается создать обработчик для работы с этим файлом. Если файл не находится – выводится ошибка.

После этого идет подсчет размера одного кластера на разделе в методе ScanFileSystem. Для подсчета вызывается функция GetVolumePathName с указанием пути к файлу, из которого получаем идентификатор раздела, а затем вызывается GetDiskFreeSpace, который возвращает количество секторов в кластере и количество байтов на сектор. Их произведение будет равно размеру одного кластера.

Далее, для получения схемы текущего размещения файла на диске вызывается управляющий код FSCTL\_GET\_RETRIEVAL\_POINTERS, который в буфере возвращает массив экстентов, по которому мы можем получить логический номер кластера (LCN, указание места расположения экстента) и ссылку на расположение следующего экстента (NEXTVCN).

В завершение, для того, чтобы прочитать содержимое файла, запускается цикл по количеству экстентов и считывается количество байт равное размеру одного кластера. При каждой итерации выполняется метод SetFilePointerEx, который перемещает курсор считывания на i \* размер кластера.

# **Теоретические вопросы**

1. **Понятие файловой системы**

Совокупность каталогов и системных структур данных, отслеживающих размещение файлов на диске и свободное дисковое пространство, называется файловой системой. Основной структурной единицей любой файловой системы является файл и каталог.

Файл – минимальная структурированная именованная последовательность данных. Каждый файл имеет набор атрибутов, состав которого зависит от используемой файловой системы.

Каталог (папка) является своеобразной объединяющей структурой для расположенных на диске файлов. Каталог может содержать в себе файлы и другие (вложенные) каталоги. Каталоги и файлы образуют на диске древовидную иерархическую структуру – дерево каталогов. Единственный каталог не входящий ни в одну из директорий называется корневым каталогом.

Файловая система включает в себя систему каталогов и системы размещения файлов на диске, простейшей из которых считается FAT в MS-DOS. Эти системы определяют возможности и эффективность манипулирования файлами – создания, записи, чтения, поиска, модификации, удаления, восстановления удаленных файлов, – а также средства восстановления файловой системы после сбоев, вызванных неисправностями и некорректными действиями пользователей или программ. В многопользовательских системах появляется еще одна задача: защита файлов одного пользователя от несанкционированного доступа другого пользователя, а также обеспечение совместной работы с файлами, к примеру, при открытии файла одним из пользователей, для других этот же файл временно будет доступен в режиме «только чтение».

1. **Структура файловой системы FAT**

FAT (File Allocation Table – таблица размещения файлов) – этот термин относится к одному из способов организации файловой системы на диске. Эта таблица хранит информацию о файлах на жестком диске в виде последовательности чисел, определяющих, где находится каждая часть каждого файла. С ее помощью операционная система выясняет, какие кластеры занимает нужный файл. Кластер – минимальный размер места на диске, которое может быть выделено для хранения одного файла.

Файловая система FAT представляет собой таблицу размещения файлов, в которой указываются:

* непосредственно адреса участков логического диска, предназначенные для размещения файлов;
* свободные области дискового пространства
* дефектные области диска.

В этой таблице каждому блоку, предназначенному для хранения данных, соответствует 16-битовое значение. Если блок свободен, то значение будет нулевым. Если же блок принадлежит файлу, то значение равно адресу следующего блока этого файла. Если это последний блок в файле, то значение – OxFFF. Существует также специальный код для обозначения плохого блока, не читаемого из-за дефекта физического носителя. В каталоге хранится номер первого блока и длина файла, измеряемая в байтах. Емкость диска при использовании 12-битовой FAT ограничена 4096 блоками (2 Мбайт), что приемлемо для дискет, но совершенно не годится для жестких дисков и других устройств большой емкости. На таких устройствах DOS использует FAT с 16-битовыми элементами. На еще больших (более 32 Мбайт) дисках DOS выделяет пространство не блоками, а кластерами из нескольких блоков.

В файловой системе FAT дисковое пространство логического диска делится на системную область и область данных. Системная область создается и инициализируется при форматировании диска, а в последующем обновляется при работе. Область данных содержит файлы и каталоги, которые образуют на диске иерархическое дерево каталогов с единственной директорией, не входящей в другие, – корневым каталогом. Область данных доступна через пользовательский интерфейс операционной системы. Системная область содержит загрузочную запись, зарезервированные сектора, таблицу размещения файлов и корневой каталог. Для каждого файла и каталога в файловой системе хранится справочная информация. Каждый элемент такого справочника занимает 32 байта и содержит: имя файла или каталога, расширение имени файла, атрибуты файла – системный, архивный, только для чтения и др., дату и время создания и последнего изменения файла, номер начального кластера, размер файла.

1. **Разновидности FAT**

Наибольшее количество кластеров, представимых таблицей, зависит от разрядности ее ячеек. На практике встречаются 12-, 16- и 32-разрядные версии FAT – соответственно FAT12, FAT16 и FAT32 (в FAT32 реально используются лишь 28-разрядные значения, но хранятся они в 4-байтовых ячейках). Малое число кластеров на разделах большого объема приводит к слишком большому размеру каждого кластера и, как следствие, непроизводительному расходу дискового пространства. Для сравнения: у файловых систем, свободных от ограничений FAT, характерный размер распределяемых блоков составляет несколько килобайт. Для обеспечения надежности на диске присутствует обычно две копии FAT. Функции ОС, работающие с файловой системой, модифицируют обе копии согласованно, а также распознают повреждение таблиц, но автоматическое восстановление не предусмотрено. В своих программах при непосредственных обращениях к FAT также необходимо поддерживать когерентность её копий.

1. **Загрузочная запись**

**MBR (Главная загрузочная запись) — это специальный загрузочный сектор**, расположенный в начале диска. Этот сектор содержит загрузчик для установленной операционной системы, а также информацию о логических разделах диска. Загрузчик — это небольшой кусок кода, который обычно используется для загрузки большого загрузчика с другого раздела или диска. Если на компьютере установлен Windows, здесь будут находиться начальные элементы загрузчика Windows. Именно поэтому приходится восстанавливать MBR, если он был перезаписан и Windows не загружается. Если установлен Linux, MBR чаще всего будет содержать загрузчик GRUB.

**MBR работает с дисками объёмом до 2 Тб**., но он может справиться и с дисками большего размера. Кроме этого MBR поддерживает не более 4 основных разделов. Если нужно больше, придётся сделать один из основных разделов «расширенным разделом» и разместить в нём логические разделы.

1. **Организация каталога**

Понятие каталога позволяет систематизировать все объекты, размещённые на носителе данных (например, на диске). В большинстве современных файловых систем используется иерархическая модель организации данных: существует один каталог, объединяющий все данные в файловой системе — это «корень» всей файловой системы, **корневой каталог**. Корневой каталог может содержать любые объекты файловой системы, и в частности, подкаталоги (каталоги первого **уровня вложенности**). Те, в свою очередь, также могут содержать любые объекты файловой системы и подкаталоги (второго уровня вложенности) и т. д. Таким образом. всё*,* что записано на диске — файлы, каталоги и специальные файлы — обязательно «принадлежит» корневому каталогу: либо непосредственно (содержится в нём), либо на некотором уровне вложенности.

Иерархию вложенных друг в друга каталогов можно соотнести с иерархией данных в системе: объединить тематически связанные файлы в каталог, тематически связанные каталоги — в один общий каталог и т. д. Если строго следовать иерархическому принципу, то чем глубже будет **уровень вложенности** каталога, тем более частным признаком должны быть объединены содержащиеся в нём данные. Если этому принципу неследовать, то вскоре окажется гораздо проще складывать все файлы в один каталог и искать нужный среди них, чем проделывать такой поиск по всем подкаталогам системы. Однако в этом случае о какой бы то ни было систематизации файлов говорить не приходится.

Структуру файловой системы можно представить наглядно в виде дерева, «корнем» которого является корневой каталог, а в вершинах расположены все остальные каталоги.