Практическое занятие № 6

Тема: Атрибуты файла и его объем.

Цель занятия: осуществлять учет объемов файлов при их хранении, передаче.

Задание: Ознакомиться с теоретическими положениями по данной теме, выполнить задания практического занятия, сформулировать вывод.

Содержание отчета по результатам выполнения практического занятия

Отчет должен содержать:

1. Название работы
2. Цель работы
3. Результаты выполнения задания 3, 4, 5
4. Вывод по работе (необходимо указать виды выполняемых работ, достигнутые цели, какие умения и навыки приобретены в ходе ее выполнения)

1.Краткие теоретические сведения

Файл - это определенное количество информации, имеющие имя, хранящиеся в долговременной памяти компьютера.

Имя файла разделено на две части точкой: имя файла (префикс) и расширение (суффикс), определяющее его тип (программа, данные и т.д.).

Имя файлу дает пользователь, а его тип обычно задается программой автоматически.

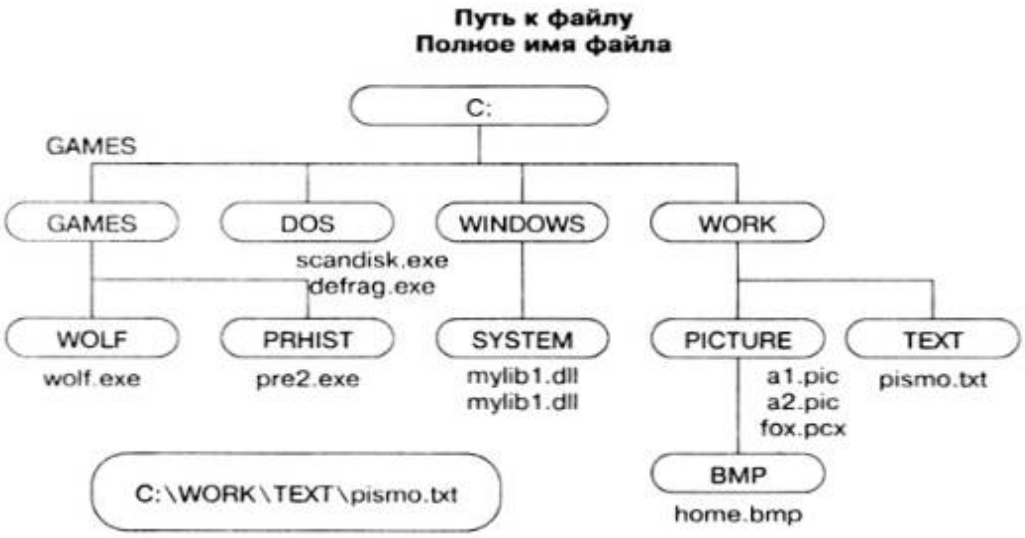
Файловая систем*а* - это функциональная часть операционной системы, обеспечивающая выполнение операций над файлами. Файловая система позволяет работать с файлами и директориями (каталогами) независимо от их содержимого, размера, типа и т. д.

Файловая система определяет общую структуру именования, хранения и организации файлов в операционной системе.

Таблица. Расширения в именах файлов

Тип файла	*Расширения*
Исполнимые файлы	.exe, .com, .bat
Текстовые файлы	.txt, .doc, .rtf
Графические файлы	.gif, .bmp, .jpg, .jpeg, .tif
Звуковые файлы	.wav, .midi, .mp3, .wma
Видеофайлы	.avi, .mpeg
Web-страницы	.htm, .html
Программы на языках программирования	.pas, .bas
Файлы данных	.dat, .dbf
Архиваторы данных	.arj, .rar, .zip

Иерархическая файловая система:



Над файлами могут производиться различные операции:

- Копирование (копия файла помещается из одного каталога в другой)
- Перемещение (сам файл перемещается в другой каталог)
- Удаление (запись о файле удаляется из каталога)
- Переименование (изменяется имя файла) и т.д.

Понятие «файл» включает не только хранимые им данные и имя, но и атрибуты. **Атрибуты** - это информация, описывающая свойства файла. Примеры возможных атрибутов файла:

- Тип файла (обычный файл, каталог, специальный файл и т. п.) > Владелец файла ' - Создатель файла;
- Пароль для доступа к файлу;
- Информация о разрешенных операциях доступа к файлу;
- Время создания, последнего доступа и последнего изменения;
- Текущий размер файла;
- Максимальный размер файла;
- Признак «только «для чтения»;
- Признак «скрытый файл»;
- Признак «системный файл»;
- Признак «архивный файл»;
- Признак «двоичный / символьный»;
- Признак «временный файл» (удалить после завершения процесса);
- Признак блокировки;
- Длина записи в файле;
- Указатель на ключевое поле в записи;
- Длина ключа.

Набор атрибутов файла определяется спецификой файловой системы; в файловых системах разного типа для характеристики файлов могут использоваться разные наборы атрибутов. Например, в файловых системах, поддерживающих неструктурированные файлы, нет необходимости использовать три последних атрибута в приведенном списке, связанных со структуризацией файла. В однопользовательской ОС в наборе атрибутов будут отсутствовать характеристики, имеющие отношение к пользователям и защите, такие как владелец файла, создатель файла, пароль для доступа к файлу, информация о разрешенном доступе к файлу.

Пользователь может получать доступ к атрибутам, используя средства, предоставленные для этих целей файловой системой. Обычно разрешается читать значения всех атрибутов, а изменять — только некоторые. Например, пользователь может изменить права доступа к файлу (при условии, что он обладает необходимыми для этого полномочиями), но изменять дату создания или текущий размер файла ему не разрешается;

Значения атрибутов файлов могут непосредственно содержаться в каталогах, как это сделано в файловой системе MS - DOS. Другим вариантом является размещение атрибутов в специальных таблицах, когда в каталогах содержаться только ссылки на эту таблицы. Такой подход реализован, например, в файловой системе ufs ОС UNIX. В этой файловой системе структура каталога очень простая. Запись о каждом файле содержит короткое символьное имя файла и указатель на индексный дескриптор файла, (так называется в ufs, таблица, в которой сосредоточены значения атрибутов файла).

В том и в другом вариантах каталоги обеспечивают связь между именами файлов и собственно файлами. Однако подход, когда имя файла отделено от его атрибутов, делает систему более гибкой. Например, файл может быть легко включен сразу в несколько каталогов. Записи об этом файле в разных каталогах могут содержать разные простые имена, но в поле ссылки будет указан один и тот же номер индексного дескриптора.

Логическая организация файлов

В общем случае данные, содержащиеся в файле, имеют некоторую логическую структуру. Эта структура является базой при разработке программы, предназначенной для обработки этих данных. Например, чтобы текст мог быть правильно выведен на экран, программа должна иметь возможность выделить отдельные слова, строки, абзацы и т. д. Признаками, отделяющими один структурный элемент от другого, могут служить определённые кодовые последовательности или просто известные программе значения смещений этих структурных элементов относительно начала файла. Поддержание структуры данных может быть либо целиком возложено на приложение, либо в той или иной степени эту работу может взять на себя файловая система.

В первом случае, когда все действия, связанные со структуризацией и интерпретацией содержимого файла, целиком относятся к ведению приложения, файл представляется ФС неструктурированной последовательностью данных. Приложение формулирует запросы к файловой системе на ввод - вывод, используя общие для всех приложений системные средства, например, указывая смещение от начала файла и количество байт которые необходимо считать или записать. Поступивший к приложению поток байт интерпретируется в соответствии с заложенной в программе логикой. Например, компилятор генерирует, а редактор связей воспринимает вполне определённый формат объектного модуля программы. При этом формат файла, в котором хранится объектный модуль, известен только этим программам. Подчеркнём, что интерпретация данных никак не связана с действительным способом их хранения в файловой системе.

Модель файла, в соответствии с которой содержимое файла представляется неструктурированной последовательностью (потоком), байт, стала популярной вместе с ОС UNIX, а теперь она широко используется в большинстве современных ОС, в том числе в MS -DOS, Windows NT / 2000, Net Ware. Неструктурированная модель файла позволяет легко организовать разделение файла между несколькими приложениями: разные приложения могут по-своему структурировать и интерпретировать данные, содержащиеся в файле.

Другая модель файла, которая применялась в ОС OS/360, DBS RSX, VMS, а в настоящее время используется достаточно редко, - это структурированный файл. В этом случае поддержание структуры файла поручается файловой системе. Файловая система видит файл как

упорядоченную последовательность логических записей. Приложение может обращаться к ФС с запросами на ввод - вывод на уровне записей, например, «считать запись 25 из файла File. Doc». ФС должна обладать информацией о структуре файла, достаточной для того, чтобы выделить любую запись. ФС предоставляет приложению доступ к записи, а вся дальнейшая обработка данных, содержащихся в этой записи, выполняется приложением. Развитием этого подхода стали системы управления базами данных (СУБД), которые поддерживают не только сложную структуру данных, но и взаимосвязи между ними.

Логическая запись является наименьшим элементом данных, которым может оперировать программист при организации обмена с внешним устройством. Даже, если физический обмен с устройством осуществляется большими единицами, операционная система должна обеспечивать программисту доступ к отдельной логической записи.

Файловая система может использовать два способа доступа к логическим записям: читать или записывать логические записи последовательно (последовательный доступ) или позиционировать файл на запись с указанным номером (прямой доступ).

Очевидно, что ОС не может поддерживать все возможные способы структурирования данных в файле, поэтому в тех ОС, в которых вообще существует поддержка логической структуризации файлов, она существует для небольшого числа широко распространенных схем логической организации файла.

К числу таких способов структуризации относится представление данных в виде записей, длина которых фиксирована в пределах файла. В таком случае доступ к некоторой записи A осуществляется либо путём последовательного чтения (A - 1) предшествующих записей, либо прямо по адресу, вычисленному по её порядковому номеру. Например, если R - длина записи, то начальный адрес записи A = R* A. Заметим, что при такой логической организации размер записи фиксирован в пределах файла, а записи в разных файлах, принадлежащих одной и той же -файловой системе, могут иметь разный размер.

Другой способ структуризации состоит в представлении данных в виде последовательности записей, размер которых изменяется в пределах одного файла. При такой организации данных для поиска нужной записи система должна последовательно считать все предшествующие записи. Вычислить адрес нужной записи по её номеру при такой логической организации невозможно, а, следовательно, не может быть применён более эффективный метод прямого доступа.

Файлы, доступ к записям которых осуществляется последовательно, по номерам позиций, называются неиндексированными, или последовательными.

Индексная логическая организация

Другим типом файлов являются индексированные файлы, они допускают более быстрый прямой доступ к отдельной логической записи. В

индексированном файле записи имеют одно или более ключевых (индексных) полей и могут адресоваться путём указаний значений этих полей. Для быстрого поиска данных в индексированном файле предусматривается специальная индексная таблица, в которой значениям ключевых полей ставится в соответствие адрес внешней памяти. Этот адрес может указывать либо непосредственно на искомую запись, либо на некоторую область внешней памяти, занимаемую несколькими записями, в число которых входит искомая запись. В последнем случае говорят, что файл имеет индексно — последовательную организацию, так как поиск включает в себя два этапа: прямой доступ по индексу к указанной области диска, а затем последовательный просмотр записей в указанной области. Ведение индексных таблиц берёт на себя файловая система. Понятно, что записи в индексированных файлах могут иметь произвольную длину.

Всё вышесказанное в большей степени относится к обычным файлам, которые могут быть как структурированными, так и неструктурированными. Что же касается других типов файлов, то они обладают определённой структурой, известной файловой системе. Например, файловая система должна понимать структуру данных, хранящихся в файле - каталоге или файле типа «символьная связь».

Физическая организация файловой системы

Представление пользователя о файловой системе как об иерархически организованном множестве информационных объектов имеет мало общего с порядком хранения файлов на диске. Файл, имеющий образ цельного, непрерывающегося набора байт, на самом деле очень часто разбросан «кусочками» по всему диску, причем это разбиение никак не связано с логической структурой файла, например, его отдельная логическая запись может быть расположена в несмежных секторах диска. Логически объединённые файлы из одного каталога совсем не обязаны соседствовать на диске. Принципы размещения файлов, каталогов и системной информации на реальном устройстве описываются физической организацией файловой системы. Очевидно, что разные файловые системы имеют разную физическую организацию.

Диски, разделы, секторы, кластеры

Основным типом устройства, которое используется в современных вычислительных системах для хранения файлов, являются дисковые накопители. Эти устройства предназначены для считывания и записи данных на жесткие и гибкие магнитные диски. Жёсткий диск состоит из одной или нескольких стеклянных или металлических пластин, каждая из которых покрыта с одной или двух сторон магнитным материалом; Таким образом, диск в общем случае состоит из пакета пластин.

На каждой стороне каждой пластины размечены тонкие концентрические кольца - дорожки (tracks) на которых хранятся данные. Количество дорожек зависит от типа диска. Нумерация дорожек начинается с 0 от внешнего края к центру диска. Когда диск вращается, элемент,

называемый головкой, считывает двоичные данные с магнитной дорожки или записывает их на магнитную дорожку.

Головка может позиционироваться над заданной дорожкой. Головки перемещаются над поверхностью диска дискретными шагами, каждый шаг соответствует сдвигу на одну дорожку. Запись на диск осуществляется благодаря способности головки изменять магнитные свойства дорожки. В некоторых дисках вдоль каждой поверхности перемещается одна головка, а в других - имеется по головке на каждую дорожку. В первом случае для поиска информации головка должна перемещаться по радиусу диска. Обычно все головки закреплены на едином перемещающем механизме и двигаются синхронно. Поэтому, когда головка фиксируется на заданной дорожке одной поверхности, все остальные головки останавливаются над дорожками с такими же номерами. В тех же случаях, когда на каждой дорожке имеется отдельная головка, никакого перемещения головок с одной дорожки на другую не требуется, за счёт этого экономится время, затрачиваемое на поиск данных.

Совокупность дорожек одного радиуса на всех поверхностях всех пластин пакета называется цилиндром (cylinder). Каждая дорожка разбивается на фрагменты, называемые секторами (sectors), или блоками ' (blocks), так что все дорожки имеют равное число секторов, в которые можно максимально записать одно и то же число байт. Сектор имеет фиксированный для конкретной системы размер, выражающийся степенью двойки. Чаще всего размер сектора составляет 512 байт. Учитывая, что дорожки разного радиуса имеют одинаковое количество секторов, плотность записи становится тем выше, чем ближе дорожка к центру.

Сектор наименьшая адресуемая единица обмена данными дискового устройства с оперативной памятью. Для того чтобы контроллер мог найти на диске нужный сектор, необходимо задать ему все составляющие адреса сектора: номер цилиндра, номер поверхности и номер сектора. Так как прикладной программе в общем случае нужен не сектор, а некоторое количество байт, не обязательно кратное размеру сектора, то типичный запрос включает чтение нескольких секторов, содержащих требуемую информацию, и одного из двух секторов, содержащих наряду с требуемыми и избыточные данные.

Операционная система при работе с диском использует, как правило, собственную единицу дискового пространства, называемую кластерам (cluster). При создании файла на диске место ему выделяется кластерами. Например, если файл имеет размер 2560 байт, а размер кластера в файловой системе определён в 1024 байта, то файлу будет выделено на диске три кластера.

Дорожки и секторы создаются в результате выполнения процедуры физического, или низкоуровневого, форматирования диска, предшествующей использованию диска. Для определения границ блоков на диск записывается идентификационная информация. Низкоуровневый формат диска не зависит от типа операционной системы, которая этот диск будет использовать.

Разметку диска под конкретный тип файловой системы выполняют процедуры высокоуровневого, или логического, форматирования. При высокоуровневом форматировании определяется размер кластера и на диск записывается информация, необходимая для работы файловой системы, в том числе информация о доступном и неиспользуемом пространстве, о границах областей, отведённых под файлы и каталоги, информация о повреждённых областях. Кроме того, на диск записывается загрузчик операционной системы - небольшая программа, которая начинает процесс инициализации операционной системы после включения питания или рестарта компьютера.

Прежде чем форматировать диск под определённую файловую систему, он может быть разбит на разделы. Раздел - непрерывная часть физического диска, которую операционная система представляет пользователю как логическое устройство (используются также названия логический диск и логический раздел). Логическое устройство функционирует так, как если бы это был отдельный физический диск. Именно с логическими устройствами работает пользователь, обращаясь к ним по символьным именам, используя, например, обозначения a, Ь, s, sys и т. п. Операционные системы разного типа используют единое для всех них представление о разделах, но создают на его основе логические устройства, специфические для каждого типа ОС. Так как файловая система, с которой работает одна ОС, в общем случае не может интерпретироваться ОС другого типа, логические устройства не могут быть использованы операционными системами разного типа. На каждом логическом устройстве может создаваться только одна файловая система.

В частном случае, когда всё дисковое пространство охватывается одним разделом, логическое устройство представляет физическое устройство в целом. Если диск разбит на несколько разделов, то для каждого из этих разделов может быть создано отдельное логическое устройство. Логическое устройство может быть создано и на базе нескольким разделов, причём эти разделы не обязательно должны принадлежать одному физическому устройству. Объединение нескольких разделов в единое логическое устройство может выполняться разными способами и преследовать разные цели, основные из которых увеличение общего объёма логического раздела, повышение производительности и отказоустойчивости. Примерами организации совместной работы нескольких дисковых разделов являются так называемые RAID - МАССИВЫ, подробнее о которых будет сказано ниже.

На разных логических устройствах одного и того же физического диска могут располагаться файловые системы разного типа.

Все разделы одного диска имеют одинаковый размер блока, определённый для данного диска в режиме низкоуровневого форматирования. Однако и в результате высокоуровневого форматирования в разных разделах одного и того же диска, предоставленных разными логическими устройствами, могут быть установлены различные файловые системы, в которых определены кластеры разных размеров.

Операционная система может поддерживать разные статусы разделов, особым образом отмечая разделы, которые могут быть использованы для загрузки модулей операционной системы, и разделы, в которых можно устанавливать только приложения и хранить файлы данных. Один из разделов диска помечается как загружаемый (или активный). Именно из этого раздела считывается загрузчик операционной системы.

Правила создания имени файла:

1. Нельзя использовать следующие символы, которые зарезервированы для специальных функций: ? . , ; : = + * / \ “ | < > [ ] ПРОБЕЛ
2. В длинных именах нельзя использовать следующие символы: ? : * / \ “ > < |

2.Задание

Задание 3. Предложите варианты имен и типов для перечисленных ниже файлов. Перенесите в тетрадь таблицу и заполните ее.

Содержание	**Имя**	**Тип**	**Полное имя файла**
Фото моей семьи			
Рецепт яблочного пирога			
Буклет «Мой колледж»			
Открытое письмо Биллу Гейтсу	BillG	doc	BillG.doc
Семейный альбом «Моя родословная»			
Репродукция картины Малевича «Черный квадрат»			
Петиция директору техникума об увеличении числа уроков информатики			
Реферат по истории			
Реклама концерта рок-группы			
Статья в журнал «Информатика и образование»			

Задание 4. Предложите варианты программ, открывающих файлы с тем или иным расширением. Перечертите таблицу в тетрадь и заполните ее.

Расширение имени файла	**Программа**
TXT	
DOC	
RTF	
BMP	
ARJ	
HTML	

«неудовлетворительно» - менее 45%.

Задание 5. Выполните задания в тетради.

a. Придумай имя текстового файла, в котором будет содержаться информация о твоем доме. Подчеркни собственное имя файла.

b. Придумай имя графического файла, в котором будет содержаться рисунок твоего дома. Подчеркни расширение файла.

c. Выпиши в один столбик правильные имена файлов, а во второй правильные имена каталогов:

Письмо.18, letter.txt, WinWord, письмо.doc, Колледж?12, Мои документы, роза.bmp, crop12.exe, 1С, red.com

3.Вопросы для самоконтроля:

1. Что такое кластер?
2. В чем суть разархивирование?
3. Как задать имя файла?
4. Что такое файловая система?
5. Что такое иерархическая файловая система?
6. Какие операции могут производиться над файлами?