



# Файл как единица хранения информации на компьютере

Информатика для СПО

# Файлы

Файл – это определенное количество информации (программа или данные), имеющие имя и хранящиеся во внешней памяти компьютера.

# Файлы

Файлы предназначены для хранения информации на внешних носителях, то есть принято, что информация, записанная, например, на диске, должна находиться внутри файла. Обычно под файлом понимают именованную часть пространства на носителе информации.

# Файловая система

**Файловая система** - это часть операционной системы, назначение которой состоит в том, чтобы организовать эффективную работу с данными, хранящимися во внешней памяти, и обеспечить пользователю удобный интерфейс при работе с такими данными.

# Файловая система

Непосредственное взаимодействие с диском - прерогатива компонента системы ввода-вывода ОС, называемого драйвером диска.

Для того чтобы избавить пользователя компьютера от сложностей взаимодействия с аппаратурой, была придумана ясная абстрактная модель файловой системы. Операции записи или чтения файла концептуально проще, чем низкоуровневые операции работы с устройствами.

# Файлы

Главная задача файловой системы (file system) – скрыть особенности ввода-вывода и дать программисту простую абстрактную модель файлов, независимых от устройств. Пользователям хорошо знакомы такие связанные с организацией файловой системы понятия, как каталог, текущий каталог, корневой каталог, путь. Для манипулирования этими объектами в операционной системе имеются системные вызовы.

# Файловая система

Файловая система позволяет при помощи системы справочников (каталогов, директорий) связать уникальное имя файла с блоками вторичной памяти, содержащими данные файла.

# Файловая система

Иерархическая структура каталогов, используемая для управления файлами, может служить другим примером индексной структуры. В этом случае каталоги или папки играют роль индексов, каждый из которых содержит ссылки на свои подкаталоги. С этой точки зрения вся файловая система компьютера представляет собой большой индексированный файл.

# Файловая система

Помимо собственно файлов и структур данных, используемых для управления файлами (каталоги, дескрипторы файлов, различные таблицы распределения внешней памяти), понятие "файловая система" включает программные средства, реализующие различные операции над файлами.

# Файловая система

Перечислим **основные функции** файловой системы.

- Идентификация файлов. Связывание имени файла с выделенным ему пространством внешней памяти.
- Распределение внешней памяти между файлами. Для работы с конкретным файлом пользователю не требуется иметь информацию о местоположении этого файла на внешнем носителе информации..
- Обеспечение надежности и отказоустойчивости. Стоимость информации может во много раз превышать стоимость компьютера.
- Обеспечение защиты от несанкционированного доступа.
- Обеспечение совместного доступа к файлам, так чтобы пользователю не приходилось прилагать специальных усилий по обеспечению синхронизации доступа.
- Обеспечение высокой производительности.

# Файлы с точки зрения пользователя

Иногда говорят, что файл - это поименованный набор связанной информации, записанной во вторичную память.

Для большинства пользователей файловая система - наиболее видимая часть ОС. Она предоставляет механизм для онлайн-ового хранения и доступа как к данным, так и к программам для всех пользователей системы.

С точки зрения пользователя, файл - единица внешней памяти, то есть данные, записанные на диск, должны быть в составе какого-нибудь файла.

# Общие сведения о файлах. Имена файлов

Файлы представляют собой абстрактные объекты. Их задача - хранить информацию, скрывая от пользователя детали работы с устройствами. Когда процесс создает файл, он дает ему имя. После завершения процесса файл продолжает существовать и через свое имя может быть доступен другим процессам.

# Общие сведения о файлах. Имена файлов

Правила именования файлов зависят от ОС.

Многие ОС поддерживают имена из двух частей (имя+расширение), например `progr.c` (файл, содержащий текст программы на языке Си) или `autoexec.bat` (файл, содержащий команды интерпретатора командного языка).

# Общие сведения о файлах. Имена файлов

Тип расширения файла позволяет ОС организовать работу с ним различных прикладных программ в соответствии с заранее оговоренными соглашениями.

Обычно ОС накладывают некоторые ограничения, как на используемые в имени символы, так и на длину имени файла. В соответствии со стандартом POSIX, популярные ОС оперируют удобными для пользователя длинными именами (до 255 символов).

# Типы файлов

Важный аспект организации файловой системы и ОС - следует ли поддерживать и распознавать типы файлов. Если да, то это может помочь правильному функционированию ОС, например не допустить вывода на принтер бинарного файла.

# Типы файлов

Основные типы файлов: регулярные (обычные) файлы и директории (справочники, каталоги). Обычные файлы содержат пользовательскую информацию. Директории - системные файлы, поддерживающие структуру файловой системы. В каталоге содержится перечень входящих в него файлов и устанавливается соответствие между файлами и их характеристиками (атрибутами).

# Типы файлов

Далее речь пойдет главным образом об **обычных файлах**.

Обычные (или регулярные) файлы реально представляют собой набор блоков (возможно, пустой) на устройстве внешней памяти, на котором поддерживается файловая система. Такие файлы могут содержать как текстовую информацию (обычно в формате ASCII), так и произвольную двоичную (бинарную) информацию.

# Типы файлов

Текстовые файлы содержат символьные строки, которые можно распечатать, увидеть на экране или редактировать обычным текстовым редактором.

# Типы файлов

Другой тип файлов - нетекстовые, или бинарные, файлы.

Обычно они имеют некоторую внутреннюю структуру.

Например, исполняемый файл в ОС Unix имеет пять секций: заголовок, текст, данные, биты релокации и символную таблицу. ОС выполняет файл, только если он имеет нужный формат.

Другим примером бинарного файла может быть архивный файл. Типизация файлов не слишком строгая.

# Типы файлов

Обычно прикладные программы, работающие с файлами, распознают тип файла по его имени в соответствии с общепринятыми соглашениями.

Например, файлы с расширениями `.c`, `.pas`, `.txt` - ASCII-файлы, файлы с расширениями `.exe` - выполнимые, файлы с расширениями `.obj`, `.zip` - бинарные и т. д.

# Атрибуты файлов

Кроме имени ОС часто связывают с каждым файлом и другую информацию, например дату модификации, размер и т. д.

Эти другие характеристики файлов называются атрибутами.

Список атрибутов в разных ОС может варьироваться.

# Атрибуты файлов

Список атрибутов обычно хранится в структуре директорий или других структурах, обеспечивающих доступ к данным файла.

Обычно он содержит следующие элементы: основную информацию (имя, тип файла), адресную информацию (устройство, начальный адрес, размер), информацию об управлении доступом (владелец, допустимые операции) и информацию об использовании (даты создания, последнего чтения, модификации и др.).

# Операции над файлами

Операционная система должна предоставить в распоряжение пользователя набор операций для работы с файлами, реализованных через системные вызовы.

Чаще всего при работе с файлом пользователь выполняет не одну, а несколько операций.

Во-первых, нужно найти данные файла и его атрибуты по символьному имени, во-вторых, считать необходимые атрибуты файла в отведенную область оперативной памяти и проанализировать права пользователя на выполнение требуемой операции.

Затем следует выполнить операцию, после чего освободить занимаемую данными файла область памяти.

# Основные файловые операции ОС Unix

- Создание файла, не содержащего данных. Смысл данного вызова - объявить, что файл существует, и присвоить ему ряд атрибутов. При этом выделяется место для файла на диске и вносится запись в каталог.
- Удаление файла и освобождение занимаемого им дискового пространства.
- Открытие файла. Перед использованием файла процесс должен его открыть. Цель данного системного вызова - разрешить системе проанализировать атрибуты файла и проверить права доступа к нему, а также считать в оперативную память список адресов блоков файла для быстрого доступа к его данным. Открытие файла является процедурой создания **дескриптора** или управляющего блока файла.

# Основные файловые операции ОС Unix

Дескриптор (описатель) файла хранит всю информацию о нем. Иногда, в соответствии с парадигмой, принятой в языках программирования, под дескриптором понимается альтернативное имя файла или указатель на описание файла в таблице открытых файлов, используемый при последующей работе с файлом .

Например, на языке Си операция открытия файла `fd=open(pathname,flags,modes);` возвращает дескриптор `fd`, который может быть задействован при выполнении операций чтения (`read(fd,buffer,count);` ) или записи..

# Основные файловые операции ОС Unix

- **Закрытие файла.** Если работа с файлом завершена, его атрибуты и адреса блоков на диске больше не нужны. В этом случае файл нужно закрыть, чтобы освободить место во внутренних таблицах файловой системы.
- **Позиционирование.** Дает возможность специфицировать место внутри файла, откуда будет производиться считывание (или запись) данных, то есть задать **текущую** позицию.

# Основные файловые операции ОС Unix

- Чтение данных из файла. Обычно это делается с текущей позиции. Пользователь должен задать объем считываемых данных и предоставить для них буфер в оперативной памяти.
- Запись данных в файл с текущей позиции. Если текущая позиция находится в конце файла, его размер увеличивается, в противном случае запись осуществляется на место имеющихся данных, которые, таким образом, теряются.

Есть и другие операции, например переименование файла, получение атрибутов файла и т. д.

# Директории. Логическая структура файлового архива

Количество файлов на компьютере может быть большим.

Отдельные системы хранят тысячи файлов, занимающие сотни гигабайт дискового пространства.

Эффективное управление этими данными подразумевает наличие в них четкой логической структуры.

Все современные файловые системы поддерживают многоуровневое именование файлов за счет наличия во внешней памяти дополнительных файлов со специальной структурой - каталогов (или директорий).

# Директории. Логическая структура файлового архива

Каждый каталог содержит список каталогов и/или файлов, содержащихся в данном каталоге. Каталоги имеют один и тот же внутренний формат, где каждому файлу соответствует одна запись в файле директории

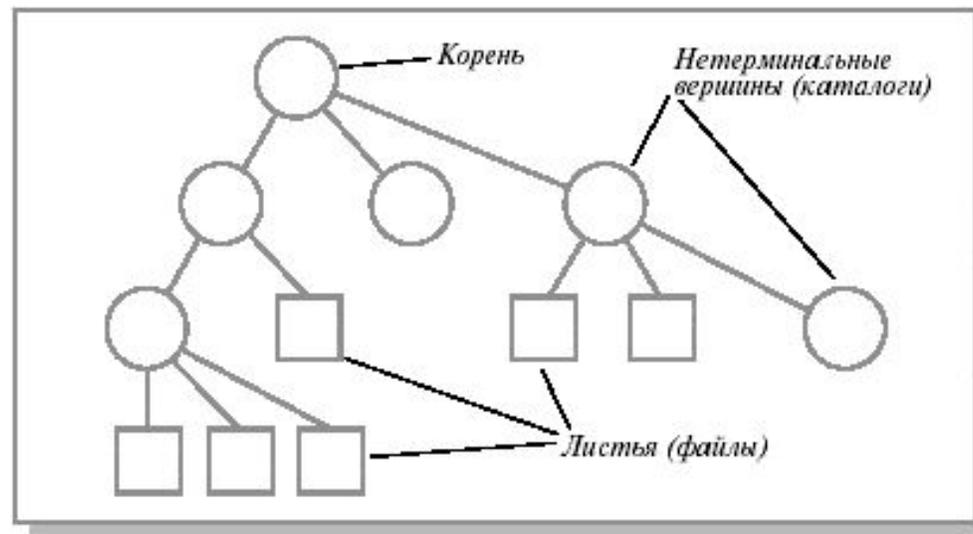
Имя файла (каталога)	Тип файла (обычный или каталог)	
Anti	К	атрибуты
Games	К	атрибуты
Autoexec.bat	О	атрибуты
mouse.com	О	атрибуты

# Директории. Логическая структура файлового архива

Число директорий зависит от системы. В ранних ОС имелась только одна корневая директория, затем появились директории для пользователей (по одной директории на пользователя). В современных ОС используется произвольная структура дерева директорий.

# Директории. Логическая структура файлового архива

Таким образом, файлы на диске образуют иерархическую древовидную структуру



# Директории. Логическая структура файлового архива

Внутри одного каталога имена листовых файлов уникальны. Имена файлов, находящихся в разных каталогах, могут совпадать. Для того чтобы однозначно определить файл по его имени, принято именовать файл так называемым **абсолютным или полным именем (pathname)**, состоящим из списка имен вложенных каталогов, по которому можно найти путь от корня к файлу плюс имя файла в каталоге, непосредственно содержащем данный файл. То есть полное имя включает цепочку имен - путь к файлу, например `/usr/games/doom`. Такие имена уникальны.

# Директории. Логическая структура файлового архива

Компоненты пути разделяют различными символами: "/" (слэш) в Unix или обратными слэшем в MS-DOS (в Multics - ">"). Таким образом, использование древовидных каталогов минимизирует сложность назначения уникальных имен.

# Директории. Логическая структура файлового архива

Указывать полное имя не всегда удобно, поэтому применяют другой способ задания имени - **относительный** путь к файлу. Он использует концепцию рабочей или текущей директории, которая обычно входит в состав атрибутов процесса, работающего с данным файлом. Тогда на файлы в такой директории можно ссылаться только по имени, при этом поиск файла будет осуществляться в рабочем каталоге. Это удобнее, но, по существу, то же самое, что и абсолютная форма.

# Директории. Логическая структура файлового архива

Для получения доступа к файлу и локализации его блоков система должна выполнить навигацию по каталогам. Рассмотрим для примера путь `/usr/linux/progr.c`. Алгоритм одинаков для всех иерархических систем. Сначала в фиксированном месте на диске находится корневая директория. Затем находится компонент пути `usr`, т. е. в корневой директории ищется файл `/usr`. Исследуя этот файл, система понимает, что данный файл является каталогом, и блоки его данных рассматривает как список файлов и ищет следующий компонент `linux` в нем. Из строки для `linux` находится файл, соответствующий компоненту `usr/linux/`. Затем находится компонент `progr.c`, который открывается, заносится в таблицу открытых файлов и сохраняется в ней до закрытия файла.

# Директории. Логическая структура файлового архива

Отклонение от типовой обработки компонентов `pathname` может возникнуть в том случае, когда этот компонент является не обычным каталогом с соответствующим ему индексным узлом и списком файлов, а служит точкой связывания (принято говорить "точкой монтирования") двух файловых архивов.

# Директории. Логическая структура файлового архива

Многие прикладные программы работают с файлами, находящимися в текущей директории, не указывая явным образом ее имени. Это дает пользователю возможность произвольным образом именовать каталоги, содержащие различные программные пакеты. Для реализации этой возможности в большинстве ОС, поддерживающих иерархическую структуру директорий, используется обозначение "." - для текущей директории и ".." - для родительской.

# Разделы диска. Организация доступа к архиву файлов

Задание пути к файлу в файловых системах некоторых ОС отличается тем, с чего начинается эта цепочка имен. В современных ОС принято разбивать диски на логические диски (это низкоуровневая операция), иногда называемые разделами (partitions). Бывает, что, наоборот, объединяют несколько физических дисков в один логический диск (например, это можно сделать в ОС Windows NT). Будем считать, что каждый раздел представляет собой отдельный (виртуальный) диск.

# Разделы диска. Организация доступа к архиву файлов

Диск содержит иерархическую древовидную структуру, состоящую из набора файлов, каждый из которых является хранилищем данных пользователя, и каталогов или директорий (то есть файлов, которые содержат перечень других файлов, входящих в состав каталога), необходимых для хранения информации о файлах системы.

# Разделы диска. Организация доступа к архиву файлов

В некоторых системах управления файлами требуется, чтобы каждый архив файлов целиком располагался на одном диске (разделе диска). В этом случае полное имя файла начинается с имени дискового устройства, на котором установлен соответствующий диск (буквы диска). Например, `c:\util\nu\ndd.exe`.

# Разделы диска. Организация доступа к архиву файлов

В ОС Unix предполагается наличие нескольких архивов файлов, каждый на своем разделе, один из которых считается корневым. После запуска системы можно "смонтировать" корневую файловую систему и ряд изолированных файловых систем в одну общую файловую систему.

# Операции над директориями

Как и в случае с файлами, система обязана обеспечить пользователя набором операций, необходимых для работы с директориями, реализованных через системные вызовы. Несмотря на то что директории - это файлы, логика работы с ними отличается от логики работы с обычными файлами и определяется природой этих объектов, предназначенных для поддержки структуры файлового архива. Совокупность системных вызовов для управления директориями зависит от особенностей конкретной ОС. Напомним, что операции над каталогами являются прерогативой ОС, то есть пользователь не может, например, выполнить запись в каталог начиная с текущей позиции.

# Операции над директориями

Рассмотрим в качестве примера некоторые системные вызовы, необходимые для работы с каталогами

- Создание директории. Вновь созданная директория включает записи с именами '.' и '..', однако считается пустой.
- Удаление директории. Удалена может быть только пустая директория.
- Открытие директории для последующего чтения. Например, чтобы перечислить файлы, входящие в директорию, процесс должен открыть директорию и считать имена всех файлов, которые она включает.
- Закрытие директории после ее чтения для освобождения места во внутренних системных таблицах.

# Операции над директориями

- Поиск. Данный системный вызов возвращает содержимое текущей записи в открытой директории. Вообще говоря, для этих целей может использоваться системный вызов `Read`, но в этом случае от программиста потребуются знание внутренней структуры директории.
- Получение списка файлов в каталоге.

# Операции над директориями

- Переименование. Имена директорий можно менять, как и имена файлов.
- Создание файла. При создании нового файла необходимо добавить в каталог соответствующий элемент.
- Удаление файла. Удаление из каталога соответствующего элемента. Если удаляемый файл присутствует только в одной директории, то он вообще удаляется из файловой системы, в противном случае система ограничивается только удалением специфицируемой записи.

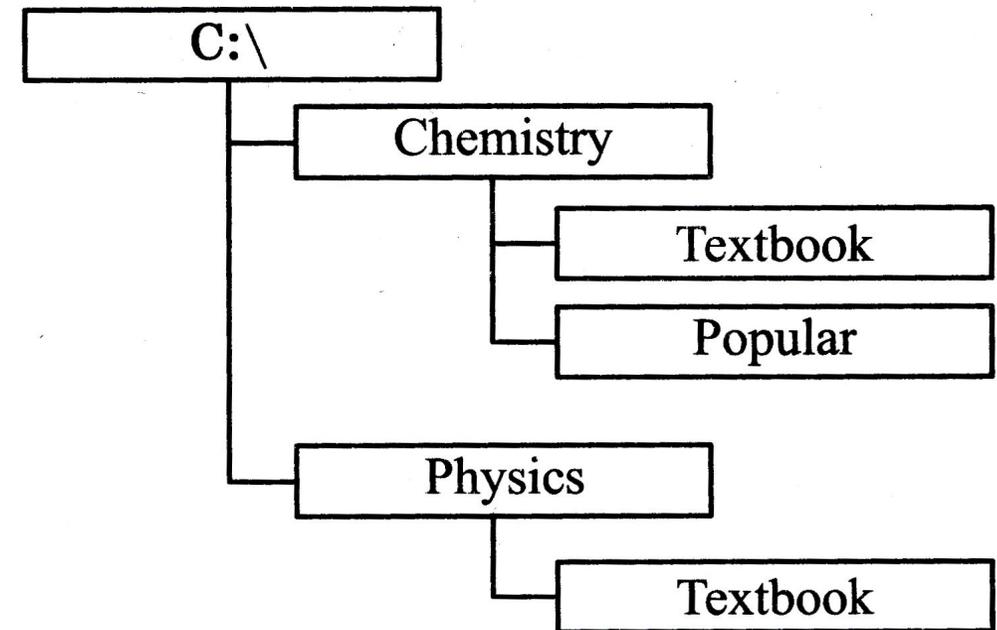
# Примеры решения типовых задач

Перемещаясь из одного каталога в другой, пользователь последовательно посетил каталоги **Textbook**, **Chemistry**, **Popular**, **Chemistry**, **C:\**, **Physics**, **Textbook**. При каждом перемещении пользователь либо спускался в каталог на уровень ниже, либо поднимался на уровень выше. Каково полное имя каталога, из которого начал перемещение пользователь?

- 1) C:\Physics\Textbook;
- 2) Chemistry\Textbook;
- 3) C:\Chemistry
- 4) C:\Chemistry\Textbook

# Примеры решения типовых задач

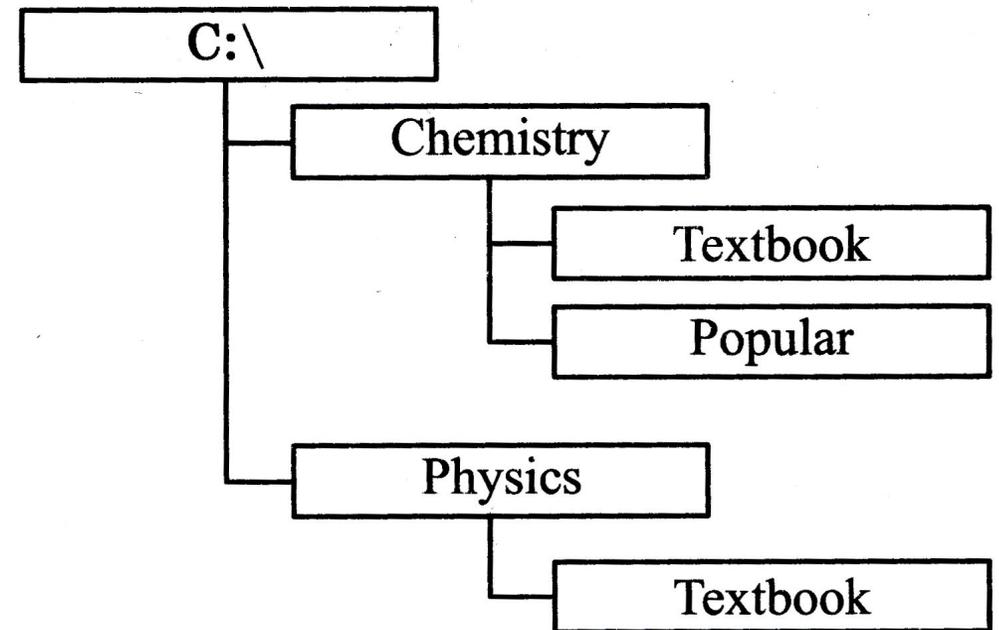
**Указания к решению.** Указание на то, что перемещение по каталогам осуществляется только на один уровень вверх или вниз, позволяет определить, что в корневом каталоге размещается каталог Chemistry, в котором находятся каталоги Textbook и Popular на одном уровне иерархии. Аналогично в корневом каталоге размещается каталог Physics, в котором находится каталог Textbook, что соответствует следующей структуре:



# Примеры решения типовых задач

Полное имя папки (каталога) включает указание имени устройства, пути к каталогу и имени каталога. Таким образом, правильным является имя C:\Chemistry\Textbook.

**Ответ:** 4 (4-й из предложенных).



# Примеры решения типовых задач

В некотором каталоге хранится файл Text.txt. После того как в этом каталоге создали подкаталог и переместили в созданный подкаталог файл Text.txt, полное имя файла стало A:\Part1\Tema1\Example\Text.txt. Каково полное имя каталога, в котором хранился файл до перемещения?

- 1) A:\Part1\Tema1\Example;
- 2) A:\Part1\Tema1;
- 3) \Part1\Tema1;
- 4) \Part1

# Примеры решения ТИПОВЫХ задач

**Решению.** По условию, файл должен находиться во вновь созданном каталоге. Если полное имя файла `A:\Part1\Tema1\Example\Text.txt`, то файл непосредственно находится в каталоге `Example`, который и был создан. Каталог `Example` находится в каталоге с полным именем `A:\Part1\Tema1`.

Ответ: 2.

# Примеры решения типовых задач

Определите, какое из указанных имён файлов удовлетворяет маске `?or???.*`

1) `norto.exe`; 2) `norto`; 3) `orton.exe`; 4) `norton`

# Примеры решения ТИПОВЫХ задач

## Решение

Маске удовлетворяют файлы, содержащие обязательно один символ перед символами «or», три символа после «or», любое количество символов, в том числе и их отсутствие после точки. Перечисленным требованиям удовлетворяет norton.

Ответ: 4.

# Задачи для самостоятельного решения

1. В некотором каталоге хранился файл Text. После того как в этом каталоге создали подкаталог и переместили в созданный подкаталог файл Text, полное имя файла стало A:\Pascal\Program\Exercise\Text. Каково было полное имя этого файла до перемещения?

- 1) A:\Pascal\Program\Exercise\Text; 2) A:\Pascal\Program;  
3) A:\Pascal\Program\Text; 4) C:\Pascal\Program\Text

# Задачи для самостоятельного решения

2 Перемещаясь из одного каталога в другой, пользователь последовательно посетил каталоги July, Summer, Липе, Summer, C:\, Winter, January. При каждом перемещении пользователь либо спускался в каталог на уровень ниже, либо поднимался на уровень выше. Каково полное имя каталога, из которого начал перемещение пользователь?

1) C:\Summer; 2) Summer\July; 3) C:\July 4) C:\Summer\July

# Задачи для самостоятельного решения

3 Перемещаясь из одного каталога в другой, пользователь последовательно посетил каталоги Novels, Literature, C:\, Literature, Fiction, Literature, Textbooks. При каждом перемещении пользователь либо спускался в каталог на уровень ниже, либо поднимался на уровень выше. Каково полное имя каталога, в котором оказался пользователь?

- 1) Literature\ Novels;
- 2) C:\Literature\Textbooks;
- 3) Literature\Novels;
- 4) Literature\Textbooks

# Задачи для самостоятельного решения

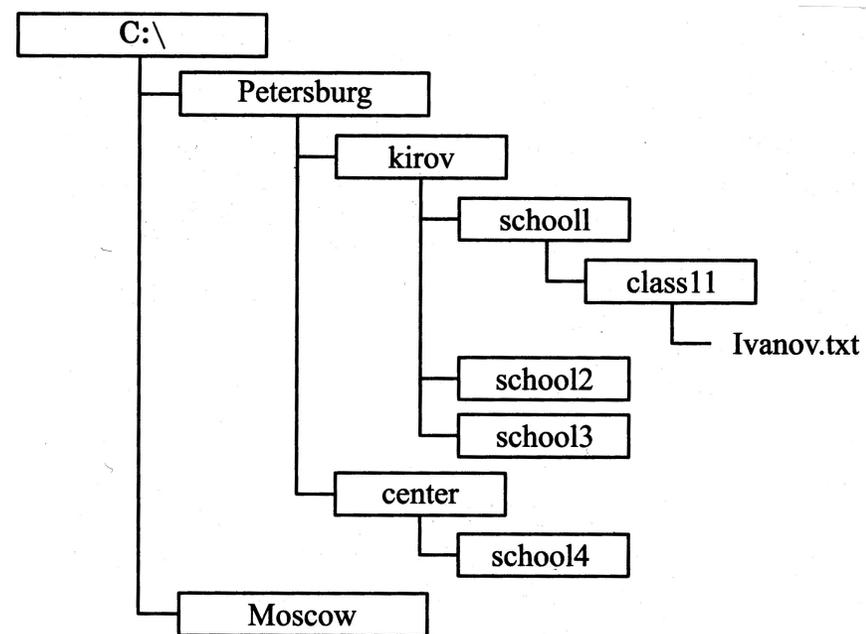
4. Перемещаясь из одного каталога в другой, пользователь последовательно посетил каталоги Writer, XXIvek, Literature, XXvek, Writer, XXvek, Literature, C:\. При каждом перемещении пользователь либо спускался в каталог на уровень ниже, либо поднимался на уровень выше. Каково полное имя каталога, из которого начал перемещение пользователь?

- 1) C:\ Literature\XXIvek\Writer; 2) C:\Literature\XXIvek;
- 3) C:\Literature\Writer; 4) C:\ Literature\XXvek

# Задачи для самостоятельного решения

5. Укажите полное имя файла Ivanov.txt.

- 1) C:\Petersburg\kirov\sch0011\class11\Ivanov.txt
- 2) Petersburg\kirov\sch0011\class11\Ivanov.txt
- 3) C:\Petersburg\kirov\sch0011\class11
- 4) Petersburg\kirov\sch0011\class11



# Задачи для самостоятельного решения

6. Определите, какое имя файла будет выбрано по маске (шаблону) \*PRO?.\*с

- 1) PROBA.doc
- 2) COPROM.doc
- 3) A1PROD.ocd
- 4) SPROS.cod