ЛАБОРАТОРНАЯ РАБОТА №1 ИНТЕРПРЕТАТОР КОМАНДНОЙ СТРОКИ ОС MS WINDOWS

Часть 1. Внешние и внутренние команды

Цель работы – знакомство с возможностями интерпретатора командной строки и командами MS Windows

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 Эволюция инструментов для автоматизации работы в ОС Microsoft Windows

В настоящее время графический интерфейс Windows стал настолько привычным, что многие пользователи и начинающие администраторы даже не задумываются об альтернативных способах управления данной ОС, связанных с командной строкой (command line) и различными сценариями (scripts), о тех преимуществах, которые дают эти инструменты с точки зрения автоматизации работы. Подобная ситуация обусловлена тем, что исторически командная строка всегда была слабым местом операционной системы Windows (по сравнению с Unix-системами).

При этом, однако, неправильно было бы думать, что командная строка или сценарии нужны только администраторам. Ведь ежедневные рутинные задачи пользователей (связанные, например, с копированием или архивированием файлов, подключением или отключением сетевых ресурсов и т.п.), которые обычно выполняются с помощью графического интерфейса проводника Windows, можно полностью самостоятельно автоматизировать, написав нехитрый командный файл, состоящий всего из нескольких строчек! Более того, для человека, не знающего основные команды Windows и такие базовые возможности ОС, как перенаправление ввода/вывода и конвейеризация команд, некоторые простейшие задачи могут показаться нетривиальными. Попробуйте,например, пользуясь только графическими средствами, сформировать файл, содержащий имена файлов из всех подкаталогов какого-либо каталога! А ведь для этого достаточно выполнить единственную команду DIR (с определенными ключами) и перенаправить вывод этой команды в нужный текстовый файл.

Каким же нам хотелось бы видеть инструмент для автоматизации работы в ОС? Желательно, чтобы было реализовано следующее:

- работа в разных версиях ОС без установки какого-либо дополнительного программного обеспечения;
- интеграция с командной строкой (непосредственное выполнение вводимых с клавиатуры команд);
- согласованный и непротиворечивый синтаксис команд и утилит;

- наличие подробной встроенной справки по командам с примерами использования.

В ОС Windows дело обстоит сложнее. На сегодняшний день одного "идеального" средства автоматизации, удовлетворяющего сразу всем перечисленным выше требованиям, в Windows нет; в последних версиях ОС поддерживаются несколько стандартных инструментов автоматизации, сильно отличающихся друг от друга: оболочка командной строки cmd.exe, среда выполнения сценариев Windows Script Host и оболочка Microsoft PowerShell. Поэтому администратору или пользователю Windows приходится выбирать, каким именно подходом воспользоваться для решения определенной задачи, а для этого желательно иметь четкое представление о сильных и слабых сторонах данных средств автоматизации. Рассмотрим достоинства и недостатки каждого из них.

1.1.1 Оболочка (интерпретатор) командной строки command.com/cmd.exe

Во всех версиях ОС Windows поддерживается интерактивная оболочка командной строки (command shell) и определенный набор утилит командной строки (количество и состав этих утилит зависит от версии ОС).

Механизм работы оболочек командной строки в разных системах одинаков: в ответ на приглашение ("подсказку", prompt), выдаваемое находящейся в ожидании оболочкой, пользователь вводит некоторую команду (функциональность этой команды может быть реализована либо самой оболочкой, либо определенной внешней утилитой), оболочка выполняет ее, при необходимости выводя на экран какую-либо информацию, после чего снова выводит приглашение и ожидает ввода следующей команды.

Оболочка представляет собой построчный интерпретатор простого языка сентенциального (директивного) программирования, в качестве операторов которого могут использоваться исполняемые программы.

Наряду с интерактивным режимом работы оболочки, как правило, поддерживают и пакетный режим, в котором система последовательно выполняет команды, записанные в текстовом файле-сценарии. Оболочка Windows не является исключением, с точки зрения программирования язык командных файлов Windows может быть охарактеризован следующим образом:

- реализация сентенциальной (директивной) парадигмы программирования;
- выполнение в режиме построчной интерпретации;
- наличие управляющих конструкций;
- поддержка нескольких видов циклов (в том числе специальных циклов для обработки текстовых файлов);

- наличие оператора присваивания (установки значения переменной);
- возможность использования внешних программ (команд) операционной системы в качестве операторов и обработки их кодов возврата;
- наличие нетипизированных переменных, которые декларируются первым упоминанием (значения переменных могут интерпретироваться как числа и использоваться в выражениях целочисленной арифметики).

Начиная с версии Windows NT, оболочка командной строки представляется интерпретатором Cmd.exe.

Итак, учитывая сказанное выше, можно сделать вывод: оболочка командной строки cmd.exe и командные файлы – наиболее универсальные и простые в изучении средства автоматизации работы в Windows, доступные во всех версиях операционной системы.

1.1.2 Поддержка языков сценариев. Сервер сценариев Windows Script Host

Следующим шагом в развитии средств и технологий автоматизации в ОС Windows стало появление сервера сценариев Windows Script Host (WSH). Этот инструмент разработан для всех версий Windows и позволяет непосредственно в ОС выполнять сценарии на полноценных языках сценариев (по умолчанию, VBScript и JScript), которые до этого были доступны только внутри HTML-страниц и работали в контексте безопасности веб-браузера (в силу этого подобные сценарии, например, могли не иметь доступа к файловой системе локального компьютера).

По сравнению с командными файлами интерпретатора cmd.exe сценарии WSH имеют несколько преимуществ.

Во-первых, VBScript и JScript – это полноценные алгоритмические языки, имеющие встроенные функции и методы для обработки символьных строк, выполнения математических операций, обработки исключительных ситуаций и т.д.; кроме того, для написания сценариев WSH может использоваться любой другой язык сценариев (например, широко распространенный в Unix-системах Perl), для которого установлен соответствующий модуль поддержки.

Во-вторых, WSH поддерживает несколько собственных объектов, свойства и методы которых позволяют решать некоторые часто возникающие повседневные задачи администратора операционной системы: работа с сетевыми ресурсами, переменными среды, системным реестром, ярлыками и специальными папками Windows, запуск и управление работой других приложений.

В-третьих, из сценариев WSH можно обращаться к службам любых приложений-серверов автоматизации (например, программ из пакета MS Office), которые регистрируют в ОС свои объекты.

Наконец, сценарии WSH позволяют работать с объектами информационной модели Windows Management Instrumentation (WMI), обеспечивающей программный интерфейс управления всеми компонентами операционной модели, а также с объектами службы каталогов Active Directory Service Interface.

Следует также отметить, что технология WSH поддерживается в Windows уже довольно давно, в Интернете (в том числе на сайте Microsoft) можно найти множество готовых сценариев.

1.1.3 Командная оболочка Microsoft PowerShell

С одной стороны функциональности и гибкости языка оболочки cmd.exe явно недостаточно, а с другой стороны сценарии WSH, работающие с объектными моделями ADSI и WMI, слишком сложны для пользователей среднего уровня и начинающих администраторов.

Перед разработчиками новой оболочки, получившей название Windows PowerShell, стояли следующие основные цели:

применение командной строки в качестве основного интерфейса администрирования;

реализация модели ObjectFlow (элементом обмена информации является объект);

переработка существующих команд, утилит и оболочки;

интеграция командной строки, объектов COM, WMI и .NET;

работа с произвольными источниками данных в командной строке по принципу файловой системы.

Самая важная идея, заложенная в PowerShell, состоит в том, что в командной строке вывод результатов команды представляет собой не текст (в смысле последовательности символов), а объект (данные вместе со свойственными им методами). В силу этого работать в PowerShell становится проще, чем в традиционных оболочках, так как не нужно выполнять никаких манипуляций по выделению нужной информации из символьного потока.

Отметим, что PowerShell одновременно является и оболочкой командной строки (пользователь работает в интерактивном режиме) и средой выполнения сценариев, которые пишутся на специальном языке PowerShell.

В целом, оболочка PowerShell намного удобнее и мощнее своих предшественников (cmd.exe и WSH), а основным недостатком, сдерживающим распространение нового инструмента, является тот факт, что PowerShell работает не во всех версиях ОС Windows. Оболочкой можно пользоваться только на версиях не ниже Windows XP Service Pack 2 с установленным пакетом .NET Framework 2.0.

1.2 Оболочка командной строки Windows. Интерпретатор Cmd.exe

Рассматриваются внутренние команды, поддерживаемые интерпретатором Cmd.exe, и наиболее часто используемые внешние команды (утилиты командной строки). Описываются механизмы перенаправления ввода/вывода, конвейеризации и условного выполнения команд.

В ОС Windows, как и в других ОС, интерактивные (набираемые с клавиатуры и сразу же выполняемые) команды выполняются с помощью так называемого командного интерпретатора, иначе называемого командным процессором или оболочкой командной строки (command shell). Начиная с версии Windows NT, в операционной системе реализован интерпретатор команд Cmd.exe, обладающий гораздо более широкими возможностями.

1.2.1 Запуск оболочки

В Windows файл Cmd.exe, как и другие исполняемые файлы, соответствующие внешним командам ОС, находятся в каталоге %SystemRoot%\SYSTEM32 (значением переменной среды %SystemRoot% является системный каталог Windows, обычно C:\Windows или C:\WinNT). Для запуска командного интерпретатора (открытия нового сеанса командной строки) можно выбрать пункт Выполнить… (Run) в меню Пуск (Start), ввести имя файла Cmd.exe и нажать кнопку OK. В результате откроется новое окно (см. рис. 1), в котором можно запускать команды и видеть результат их работы.

Рис. 1 - Командное окно интерпретатора Cmd.exe в Windows 7

1.2.2 Настройка параметров командного окна интерпретатора

У утилиты командной строки, которая поставляется в виде стандартного приложения ОС Windows, имеется свой набор опций и параметров настройки. Один из способов просмотра этих опций – использование пункта Свойства управляющего меню окна (нажать правой кнопкой мыши на заголовок окна). В окне свойств (см. рис. 2) будут доступны четыре вкладки с опциями: общие, шрифт, расположение и цвета.

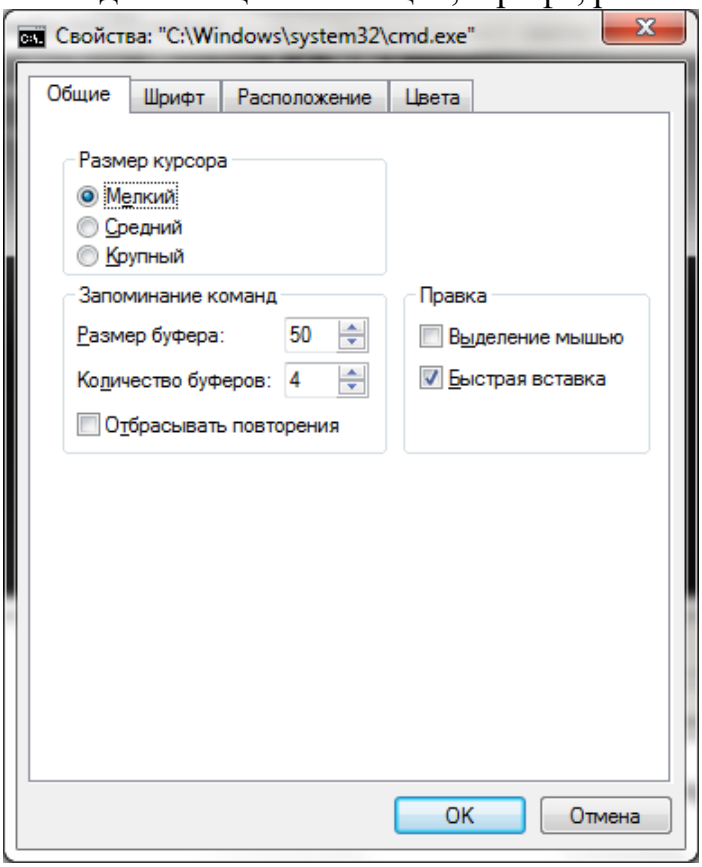

Рис. 2 – окно настройки свойств интерпретатора

1.2.3 Внутренние и внешние команды. Структура команд

Некоторые команды распознаются и выполняются непосредственно самим командным интерпретатором — такие команды называются **внутренними** (например, `COPY` или `DIR`). Другие команды ОС представляют собой отдельные программы, расположенные по умолчанию в том же каталоге, что и Cmd.exe, которые Windows загружает и выполняет аналогично другим программам. Такие команды называются внешними (например, `MORE` или `XCOPY`).

Рассмотрим структуру самой командной строки и принцип работы с ней. Для того, чтобы выполнить команду, после приглашения командной строки (например, `C:\>`) следует ввести имя этой команды (регистр не важен), ее параметры и ключи (если они необходимы) и нажать клавишу <Enter>. Например:

C:\>COPY C:\myfile.txt A:\ /V

Имя команды здесь — COPY, параметры — C:\myfile.txt и A:\, а ключом является /V. Отметим, что в некоторых командах ключи могут начинаться не с символа /, а с символа – (минус), например, -V.

Многие команды Windows имеют большое количество дополнительных параметров и ключей, запомнить которые зачастую бывает трудно. Большинство команд снабжено встроенной справкой, в которой кратко описываются назначение и синтаксис данной команды. Получить доступ к такой справке можно путем ввода команды с ключом /?. Например, если выполнить команду ATTRIB /?, то в окне MS-DOS мы увидим следующий текст: Отображение и изменение атрибутов файлов.

ATTRIB [+R|-R] [+A|-A] [+S|-S] [+H|-H] [[диск:][путь]имя_файла] [/S]

\+ Установка атрибута.

\- Снятие атрибута.

R Атрибут "Только чтение".

A Атрибут "Архивный".

S Атрибут "Системный".

H Атрибут "Скрытый".

/S Обработка файлов во всех вложенных папках указанного пути.

Для некоторых команд текст встроенной справки может быть довольно большим и не умещаться на одном экране. В этом случае помощь можно выводить последовательно по одному экрану с помощью команды MORE и символа конвейеризации |, например:

XCOPY /? | MORE

В этом случае после заполнения очередного экрана вывод помощи будет прерываться до нажатия любой клавиши. Кроме того, используя символы перенаправления вывода > и >>, можно текст, выводимый на экран, направить в текстовый файл для дальнейшего просмотра. Например, для вывода текста справки к команде XCOPY в текстовый файл xcopy.txt, используется следующая команда:

XCOPY /? > XCOPY.TXT

Замечание. Вместо имени файла можно указывать обозначения устройств компьютера. В Windows поддерживаются следующие **имена устройств**: PRN (принтер), CON (терминал: при вводе это клавиатура, при выводе - монитор), NUL (пустое устройство, все операции ввода/вывода для него игнорируются).

1.2.4 Перенаправление ввода/вывода и конвейеризация (композиция) команд

С помощью переназначения устройств ввода/вывода одна программа может направить свой вывод на вход другой или перехватить вывод

другой программы, используя его в качестве своих входных данных. Таким образом, имеется возможность передавать информацию от процесса к процессу при минимальных программных издержках. Практически это означает, что для программ, которые используют стандартные входные и выходные устройства, ОС позволяет:

- выводить сообщения программ не на экран (стандартный выходной поток), а в файл или на принтер (перенаправление вывода);
- читать входные данные не с клавиатуры (стандартный входной поток), а из заранее подготовленного файла (перенаправление ввода);
- передавать сообщения, выводимые одной программой, в качестве входных данных для другой программы (конвейеризация или композиция команд).

Из командной строки эти возможности реализуются следующим образом. Для того, чтобы перенаправить текстовые сообщения, выводимые какой-либо командой, в текстовый файл, нужно использовать конструкцию
команда > имя_файла

Если при этом заданный для вывода файл уже существовал, то он перезаписывается, если не существовал — создается. Можно также не создавать файл заново, а *дописывать* информацию, выводимую командой, в конец существующего файла. Для этого команда перенаправления вывода должна быть задана так:
команда >> имя_файла

С помощью символа < можно прочитать входные данные для заданной команды не с клавиатуры, а из определенного (заранее подготовленного) файла:
команда < имя_файла

Приведем несколько примеров перенаправления ввода/вывода:

Вывод встроенной справки для команды COPY в файл copy.txt:
COPY /? > copy.txt

Добавление текста справки для команды XCOPY в файл copy.txt:
XCOPY /? >> copy.txt

Вывод текущей даты в файл date.txt (DATE /T — это команда для просмотра и изменения системной даты, T ключ для получения только даты без запроса нового значения):
DATE /T > date.txt

Если при выполнении определенной команды возникает ошибка, то сообщение об этом по умолчанию выводится на экран. В случае необходимости сообщения об ошибках (стандартный поток ошибок) можно перенаправить в текстовый файл с помощью конструкции
команда 2> имя_файла

В этом случае стандартный вывод будет производиться на экран.

Также имеется возможность информационные сообщения и сообщения об ошибках выводить в один и тот же файл. Делается это следующим образом:
команда > имя_файла 2>&1

Например, в приведенной ниже команде стандартный выходной поток и стандартный поток ошибок перенаправляются в файл copy.txt:
XCOPY A:\1.txt C: > copy.txt 2>&1
Наконец, с помощью конструкции
команда1 | команда2 можно использовать сообщения, выводимые первой командой, в качестве входных данных для второй команды (конвейер команд).

Используя механизмы перенаправления ввода/вывода и конвейеризации, можно из командной строки посылать информацию на различные устройства и автоматизировать ответы на запросы, выдаваемые командами или программами, использующими стандартный ввод. Для решения таких задач служит команда
ECHO [сообщение], которая выводит сообщение на экран. Пример использования этой команды.

Удаление всех файлов в текущем каталоге без предупреждения (автоматический положительный ответ на запрос об удалении):
ECHO y | DEL *.*

1.2.5 Команды MORE и SORT

Одной из наиболее часто использующихся команд, для работы с которой применяется перенаправление ввода/вывода и конвейеризация, является `MORE`. Эта команда считывает стандартный ввод из конвейера или перенаправленного файла и выводит информацию частями, размер каждой из которых не больше размера экрана. Используется `MORE` обычно для просмотра длинных файлов. Возможны три варианта синтаксиса этой команды:
MORE [диск:][путь]имя_файла
MORE < [диск:][путь]имя_файла
имя_команды | MORE

Параметр `[диск:][путь]имя_файла` определяет расположение и имя файла с просматриваемыми на экране данными. Параметр `имя_команды` задает команду, вывод которой отображается на экране (например, `DIR` или команда `TYPE`, использующаяся для вывода содержимого текстового файла на экран). Приведем два примера.

Для поэкранного просмотра текстового файла news.txt возможны следующие варианты команд:
MORE news.txt
MORE < news.txt
TYPE news.txt | MORE

Другой распространенной командой, использующей перенаправление ввода/вывода и конвейеризацию, является SORT. Эта команда работает как фильтр: она считывает символы в заданном столбце, упорядочивает их в возрастающем или убывающем порядке и выводит отсортированную информацию в файл, на экран или другое устройство. Возможны два варианта синтаксиса этой команды:

SORT [/R] [/+n] [[диск1:][путь1]файл1] [> [диск2:][путь2]файл2]

или

[команда |] SORT [/R] [/+n] [> [диск2:][путь2]файл2]

В первом случае параметр [диск1:][путь1]файл1 определяет имя файла, который нужно отсортировать. Во втором случае будут отсортированы выходные данные указанной команды. Если параметры файл1 или команда не заданы, то SORT будет считывать данные с устройства стандартного ввода.

Параметр [диск2:][путь2]файл2 задает файл, в который будет направляется сортированный вывод; если этот параметр не задан, то вывод будет направлен на устройство стандартного вывода.

По умолчанию сортировка выполняется в порядке возрастания. Ключ /R позволяет изменить порядок сортировки на обратный (от Z к A и затем от 9 до 0). Например, для поэкранного просмотра отсортированного в обратном порядке файла price.txt, нужно задать следующую команду:

SORT /R < price.txt |MORE

Ключ /+n задает сортировку в файле по символам n-го столбца. Например, /+10 означает, что сортировка должна осуществляться, начиная с 10-й позиции в каждой строке. По умолчанию файл сортируется по первому столбцу.

1.2.6 Условное выполнение и группировка команд

В командной строке Windows можно использовать специальные символы, которые позволяют вводить несколько команд одновременно и управлять работой команд в зависимости от результатов их выполнения. С помощью таких символов условной обработки можно содержание небольшого пакетного файла записать в одной строке и выполнить полученную составную команду.

Используя символ амперсанда &, можно разделить несколько утилит в одной командной строке, при этом они будут выполняться друг за другом. Например, если набрать команду DIR & PAUSE & COPY /? и нажать клавишу <Enter>, то вначале на экран будет выведено содержимое текущего каталога, а после нажатия любой клавиши — встроенная справка команды COPY.

Условная обработка команд в Windows осуществляется с помощью символов && и || следующим образом. Двойной амперсанд && запускает

команду, стоящую за ним в командной строке, только в том случае, если команда, стоящая перед амперсандами была выполнена успешно. Например, если в корневом каталоге диска C: есть файл plan.txt, то выполнение строки `TYPE C:\plan.txt && DIR` приведет к выводу на экран этого файла и содержимого текущего каталога. Если же файл C:\plan.txt не существует, то команда `DIR` выполняться не будет.

Два символа `||` осуществляют в командной строке обратное действие, т.е. запускают команду, стоящую за этими символами, только в том случае, если команда, идущая перед ними, не была успешно выполнена. Таким образом, если в предыдущем примере файл C:\plan.txt будет отсутствовать, то в результате выполнения строки `TYPE C:\plan.txt || DIR` на экран выведется содержимое текущего каталога.

Отметим, что условная обработка действует только на ближайшую команду, то есть в строке

TYPE C:\plan.txt && DIR & COPY /?

команда `COPY /?` запустится в любом случае, независимо от результата выполнения команды `TYPE C:\plan.txt`.

Несколько утилит можно сгруппировать в командной строке с помощью *круглых скобок*. **Рассмотрим, например, две строки:**

TYPE C:\plan.txt && DIR & COPY /?

TYPE C:\plan.txt && (DIR & COPY /?)

В первой из них символ условной обработки `&&` действует только на команду `DIR`, во второй — одновременно на две команды: `DIR` и `COPY`.

1.3 Команды для работы с файловой системой

Рассмотрим некоторые наиболее часто используемые команды для работы с файловой системой. Отметим сначала несколько особенностей определения путей к файлам в Windows.

1.3.1 Пути к объектам файловой системы

Файловая система логически имеет древовидную структуру и имена файлов задаются в формате `[диск:][путь\]имя_файла`, то есть обязательным параметром является только имя файла. При этом, если путь начинается с символа `"\"`, то маршрут вычисляется от корневого каталога, иначе — от текущего каталога. Например, имя C:123.txt задает файл 123.txt в текущем каталоге на диске C:, имя C:\123.txt — файл 123.txt в корневом каталоге на диске C:, имя ABC\123.txt — файл 123.txt в подкаталоге ABC текущего каталога.

Существуют особые обозначения для текущего каталога и родительского каталогов. Текущий каталог обозначается символом .

(точка), его родительский каталог — символами .. (две точки). Например, если текущим каталогом является C:\WINDOWS, то путь к файлу autoexec.bat в корневом каталоге диска C: может быть записан в виде ..\autoexec.bat.

В именах файлов (но не дисков или каталогов) можно применять так называемые **групповые символы** или шаблоны: ? (вопросительный знак) и * (звездочка). Символ * в имени файла означает произвольное количество любых допустимых символов, символ ? — один произвольный символ или его отсутствие. Скажем, под шаблон text??1.txt подходят, например, имена text121.txt и text11.txt, под шаблон text*.txt — имена text.txt, textab12.txt, а под шаблон text.* — все файлы с именем text и произвольным расширением.

Для того, чтобы использовать длинные имена файлов при работе с командной строкой, их нужно заключать в двойные кавычки. Например, чтобы запустить файл с именем 'Мое приложение.exe' из каталога 'Мои документы', нужно в командной строке набрать `"C:\Мои документы\Мое приложение.exe"` и нажать клавишу <Enter>.

1.3.2 Команда CD

Текущий каталог можно изменить с помощью команды CD [диск:][путь\]. Путь к требуемому каталогу указывается с учетом приведенных выше замечаний. Например, команда `CD \` выполняет переход в корневой каталог текущего диска. Если запустить CD без параметров, то на экран будут выведены имена текущего диска и каталога.

1.3.3 Команда COPY

Одной из наиболее часто повторяющихся задач при работе на компьютере является копирование и перемещение файлов из одного места в другое. Для копирования одного или нескольких файлов используется команда COPY.

Синтаксис этой команды:

COPY [/A|/B] источник [/A|/B] [+ источник [/A|/B] [+ ...]]
[результат [/A|/B]] [/V][/Y|/–Y]

Краткое описание параметров и ключей команды `COPY` приведено в (табл. 1).

Таблица 1.

Параметры и ключи команды COPY

Параметр	Описание
источник	Имя копируемого файла или файлов
/A	Файл является текстовым файлом ASCII, то есть конец файла обозначается символом с кодом ASCII 26 (<Ctrl>+<Z>)

Параметр	Описание
источник	Имя копируемого файла или файлов
/B	Файл является двоичным. Этот ключ указывает на то, что интерпретатор команд должен при копировании считывать из источника число байт, заданное размером в каталоге копируемого файла
результат	Каталог для размещения результата копирования и/или имя создаваемого файла
/V	Проверка правильности копирования путем сравнения файлов после копирования
/Y	Отключение режима запроса подтверждения на замену файлов
/-Y	Включение режима запроса подтверждения на замену файлов

Примеры использования команды `COPY`.

Копирование файла abc.txt из текущего каталога в каталог D:\PROGRAM под тем же именем:

```
COPY abc.txt D:\PROGRAM
```

Копирование файла abc.txt из текущего каталога в каталог D:\PROGRAM под новым именем def.txt:

```
COPY abc.txt D:\PROGRAM\def.txt
```

Копирование всех файлов с расширением txt с диска A: в каталог 'Мои документы' на диске C:

```
COPY A:\*.txt "C:\Мои документы"
```

Если не задать в команде целевой файл, то команда COPY создаст копию файла-источника с тем же именем, датой и временем создания, что и исходный файл, и поместит новую копию в текущий каталог на текущем диске. Например, для того, чтобы скопировать все файлы из корневого каталога диска A: в текущий каталог, достаточно выполнить такую команду:

```
COPY A:\*.*
```

Пример 1. Создания нового текстового файла и записи в него информации без использования текстового редактора.

Для решения задачи необходимо ввести команду `COPY CON my.txt`, которая будет копировать то, что набирается на клавиатуре в файл my.txt (если этот файл существовал, то он перезапишется, иначе — создастся). Для завершения ввода необходимо ввести символ конца файла, то есть нажать клавиши <Ctrl>+<Z>.

Команда `COPY` может также объединять (склеивать) несколько файлов в один. Для этого необходимо указать единственный результирующий файл и несколько исходных. Это достигается путем использования групповых знаков (? и *) или формата файл1 + файл2 +

файл3. Например, для объединения файлов 1.txt и 2.txt в файл 3.txt можно задать следующую команду:

COPY 1.txt+2.txt 3.txt

Объединение всех файлов с расширением dat из текущего каталога в один файл all.dat может быть произведено так:

COPY /B *.dat all.dat

Ключ `/B` здесь используется для предотвращения усечения соединяемых файлов, так как при комбинировании файлов команда `COPY` по умолчанию считает файлами текстовыми.

Если имя целевого файла совпадает с именем одного из копируемых файлов (кроме первого), то исходное содержимое целевого файла теряется. Если имя целевого файла опущено, то в его качестве используется первый файл из списка. Например, команда `COPY 1.txt+2.txt` добавит к содержимому файла 1.txt содержимое файла 2.txt. Командой `COPY` можно воспользоваться и для присвоения какому-либо файлу **текущей даты и времени** без модификации его содержимого. Для этого нужно ввести команду

COPY /B 1.txt +,,

Здесь запятые указывают на пропуск параметра приемника, что и приводит к требуемому результату.

Команда `COPY` имеет и свои недостатки. Например, с ее помощью нельзя копировать скрытые и системные файлы, файлы нулевой длины, файлы из подкаталогов. Кроме того, если при копировании группы файлов COPY встретит файл, который в данный момент нельзя скопировать (например, он занят другим приложением), то процесс копирования полностью прервется, и остальные файлы не будут скопированы.

1.3.4 Команда XCOPY

Указанные при описании команды `COPY` проблемы можно решить с помощью команды `XCOPY`, которая предоставляет намного больше возможностей при копировании. Необходимо отметить, правда, что `XCOPY` может работать только с файлами и каталогами, но не с **устройствами**.

Синтаксис команды: XCOPY источник [результат] [ключи]

Команда `XCOPY` имеет множество ключей, далее приведены лишь некоторых из них. Ключ `/D[:[дата]]` позволяет копировать только файлы, измененные не ранее указанной даты. Если параметр дата не указан, то копирование будет производиться только если источник новее результата. Например, команда XCOPY "C:\Мои документы*.*" "D:\BACKUP\Мои документы" /D скопирует в каталог 'D:\BACKUP\Мои документы' только те файлы из каталога 'C:\Мои документы', которые были изменены со времени последнего подобного копирования или которых вообще не было в 'D:\BACKUP\Мои документы'.

Ключ /S позволяет копировать все непустые подкаталоги в каталоге-источнике. С помощью же ключа /E можно копировать вообще все подкаталоги, включая и пустые.

Если указан ключ /C, то копирование будет продолжаться даже в случае возникновения ошибок. Это бывает очень полезным при операциях копирования, производимых над группами файлов, например, при резервном копировании данных.

Ключ /I важен для случая, когда копируются несколько файлов, а файл назначения отсутствует. При задании этого ключа команда XCOPY считает, что файл назначения должен быть каталогом. Например, если задать ключ /I в команде копирования всех файлов с расширением txt из текущего каталога в несуществующий еще подкаталог TEXT, XCOPY *.txt TEXT /I то подкаталог TEXT будет создан без дополнительных запросов.

Ключи /Q, /F и /L отвечают за режим отображения при копировании. При задании ключа /Q имена файлов при копировании не отображаются, ключа /F — отображаются полные пути источника и результата. Ключ /L обозначает, что отображаются только файлы, которые должны быть скопированы (при этом само копирование не производится).

С помощью ключа /H можно копировать скрытые и системные файлы, а с помощью ключа /R — заменять файлы с атрибутом "Только для чтения". Например, для копирования всех файлов из корневого каталога диска C: (включая системные и скрытые) в каталог SYS на диске D:, нужно ввести следующую команду:

```
XCOPY C:\*.* D:\SYS /H
```

Ключ /T позволяет применять XCOPY для копирования только структуры каталогов источника, без дублирования находящихся в этих каталогах файлов, причем пустые каталоги и подкаталоги не включаются. Для того, чтобы все же включить пустые каталоги и подкаталоги, нужно использовать комбинацию ключей /T /E.

Используя XCOPY можно при копировании обновлять только уже существующие файлы (новые файлы при этом не записываются). Для этого применяется ключ /U. Например, если в каталоге C:\2 находились файлы a.txt и b.txt, а в каталоге C:\1 — файлы a.txt, b.txt, c.txt и d.txt, то после выполнения команды:

```
XCOPY C:\1 C:\2 /U
```

в каталоге C:\2 по-прежнему останутся лишь два файла a.txt и b.txt, содержимое которых будет заменено содержимым соответствующих файлов из каталога C:\1. Если с помощью XCOPY копировался файл с атрибутом "Только для чтения", то по умолчанию у файла-копии этот атрибут снимется. Для того, чтобы копировать не только данные, но и полностью атрибуты файла, необходимо использовать ключ /K.

Ключи /Y и /-Y определяют, нужно ли запрашивать подтверждение перед заменой файлов при копировании. /Y означает, что такой запрос нужен, /-Y — не нужен.

1.3.5. Команда DIR

Команда: DIR [диск:][путь][имя_файла] [ключи] используется для вывода информации о содержимом дисков и каталогов. Параметр [диск:][путь] задает диск и каталог, содержимое которого нужно вывести на экран. Параметр [имя_файла] задает файл или группу файлов, которые нужно включить в список.

Например, команда DIR C:*.bat выведет на экран все файлы с расширением bat в корневом каталоге диска C:. Если задать эту команду без параметров, то выводится метка диска и его серийный номер, имена (в коротком и длинном вариантах) файлов и подкаталогов, находящихся в текущем каталоге, а также дата и время их последней модификации. После этого выводится число файлов в каталоге, общий объем (в байтах), занимаемый файлами, и объем свободного пространства на диске. Например: Том в устройстве C имеет метку PHYS1_PART2
Серийный номер тома: 366D-6107
Содержимое папки C:\aditor

```
.            <ПАПКА>     25.01.15  17:15 .
..           <ПАПКА>     25.01.15  17:15 ..
HILITE   DAT          1 082  18.09.16              18:55 hilite.dat
TEMPLT01 DAT             48  07.08.16              1:00 templt01.dat
TTABLE   DAT            357  07.08.16              1:00 ttable.dat
ADITOR   EXE        461 312  01.12.15             23:13 aditor.exe
README   TXT          3 974  25.01.15             17:26 readme.txt
ADITOR   HLP         24 594  08.10.16             23:12 aditor.hlp
ТЕКСТО~1 TXT              0  11.03.15        9:02 Текстовый файл.txt
      11 файлов        533 647 байт
       2 папок     143 261 696 байт свободно
```

С помощью ключей команды DIR можно задать различные режимы расположения, фильтрации и сортировки. Например, при использовании ключа /W перечень файлов выводится в широком формате с максимально возможным числом имен файлов или каталогов на каждой строке. Например: Том в устройстве C имеет метку PHYS1_PART2
Серийный номер тома: 366D-6107
Содержимое папки C:\aditor

```
[.]             [..]            TEMPLT02.DAT  UNINST1.000     HILITE.DAT
TEMPLT01.DAT  UNINST0.000     TTABLE.DAT     ADITOR.EXE
      README.TXT
ADITOR.HLP      ТЕКСТО~1.TXT
```

11 файлов 533 647 байт
2 папок 143 257 600 байт свободно

С помощью ключа /A[[:]атрибуты] можно вывести имена только тех каталогов и файлов, которые имеют заданные атрибуты (R — "Только чтение", A — "Архивный", S — "Системный", H — "Скрытый", префикс "–" имеет значение НЕ). Если ключ /A используется более чем с одним значением атрибута, будут выведены имена только тех файлов, у которых все атрибуты совпадают с заданными. Например, для вывода имен всех файлов в корневом каталоге диска C:, которые одновременно являются скрытыми и системными, можно задать команду

DIR C:\ /A:HS

а для вывода всех файлов, кроме скрытых — команду

DIR C:\ /A:-H

Отметим здесь, что атрибуту каталога соответствует буква D, и для того, чтобы, например, вывести список всех каталогов диска C:, нужно задать команду

DIR C: /A:D

Ключ /O[[:]сортировка] задает порядок сортировки содержимого каталога при выводе его командой DIR. Если этот ключ опущен, DIR печатает имена файлов и каталогов в том порядке, в котором они содержатся в каталоге. Если ключ /O задан, а параметр сортировка не указан, то DIR выводит имена в алфавитном порядке. В параметре сортировка можно использовать следующие значения: N — по имени (алфавитная), S — по размеру (начиная с меньших), E — по расширению (алфавитная), D — по дате (начиная с более старых), A — по дате загрузки (начиная с более старых), G — начать список с каталогов. Префикс "–" означает обратный порядок. Если задается более одного значения порядка сортировки, файлы сортируются по первому критерию, затем по второму и т.д.

Ключ /S означает вывод списка файлов из заданного каталога и его подкаталогов. Ключ /B перечисляет только названия каталогов и имена файлов (в длинном формате) по одному на строку, включая расширение. При этом выводится только основная информация, без итоговой. Например:

templt02.dat
UNINST1.000
hilite.dat
templt01.dat
UNINST0.000
ttable.dat
aditor.exe
readme.txt

aditor.hlp
Текстовый файл.txt

1.3.6 Команды MKDIR и RMDIR

Для создания нового каталога и удаления уже существующего пустого каталога используются команды MKDIR [диск:]путь и RMDIR [диск:]путь [ключи] соответственно (или их короткие аналоги MD и RD). Например:

```
MKDIR "C:\Примеры"
RMDIR "C:\Примеры"
```

Команда MKDIR не может быть выполнена, если каталог или файл с заданным именем уже существует. Команда RMDIR не будет выполнена, если удаляемый каталог не пустой.

1.3.7 Команда DEL

Удалить один или несколько файлов можно с помощью команды
DEL [диск:][путь]имя_файла [ключи]

Для удаления сразу нескольких файлов используются групповые знаки ? и *. Ключ /S позволяет удалить указанные файлы из всех подкаталогов, ключ /F – принудительно удалить файлы, доступные только для чтения, ключ /A[[:]атрибуты] – отбирать файлы для удаления по атрибутам (аналогично ключу /A[[:]атрибуты] в команде DIR).

1.3.8 Команда REN

Переименовать файлы и каталоги можно с помощью команды RENAME (REN). **Синтаксис этой команды имеет следующий вид:**
REN [диск:][путь][каталог1|файл1] [каталог2|файл2]

Здесь параметр каталог1|файл1 определяет название каталога/файла, которое нужно изменить, а каталог2|файл2 задает новое название каталога/файла. В любом параметре команды REN можно использовать групповые символы ? и *. При этом представленные шаблонами символы в параметре файл2 будут идентичны соответствующим символам в параметре файл1. Например, чтобы изменить у всех файлов с расширением txt в текущей директории расширение на doc, нужно ввести такую команду:
REN *.txt *.doc

Если файл с именем файл2 уже существует, то команда REN прекратит выполнение, и произойдет вывод сообщения, что файл уже существует или занят. Кроме того, в команде REN нельзя указать другой диск или каталог для создания результирующих каталога и файла. Для этой цели нужно использовать команду MOVE, предназначенную для переименования и перемещения файлов и каталогов.

1.3.9 Команда MOVE

Синтаксис команды для перемещения одного или более файлов имеет вид:

MOVE [/Y|/–Y] [диск:][путь]имя_файла1[,...] результирующий_файл

Синтаксис команды для переименования папки имеет вид:

MOVE [/Y|/–Y] [диск:][путь]каталог1 каталог2

Здесь параметр `результирующий_файл` задает новое размещение файла и может включать имя диска, двоеточие, имя каталога, либо их сочетание. Если перемещается только один файл, допускается указать его новое имя. Это позволяет сразу переместить и переименовать файл. Например, MOVE "C:\Мои документы\список.txt" D:\list.txt.

Если указан ключ /-Y, то при создании каталогов и замене файлов будет выдаваться запрос на подтверждение. Ключ /Y отменяет выдачу такого запроса.

2 МЕТОДИКА ВЫПОЛНЕНИЯ

1. Ознакомиться с теоретическими сведениями.
2. Запустить интерпретатор командной строки
3. Увеличить размер окна интерпретатора и задать цвет фона и цвет шрифта (рекомендуется синий фон и белый шрифт).
4. Создать список фамилий студентов группы, используя пример 1. Отсортировать список в алфавитном порядке и сохранить его в новом файле.

Замечание 1. При создании текстового файла интерпретатор командной строки использует кодировку **кириллица (DOS)**. Поэтому рекомендуется переназначить вывод в файл с расширением .**txt,** а для просмотра содержимого файла использовать Internet Explorer, указав вид кодировки кириллица (DOS). Пример вывода содержимого текстового файла приведен на рис. 3.

Замечание 2. Интерпретатор хранит историю введенных команд в буфере (размером 50 строк). Для просмотра содержимого буфера используйте клавиши клавиатуры СТРЕЛКА ВВЕРХ и СТРЕЛКА ВНИЗ. Полученную команду можно отредактировать и выполнить снова.

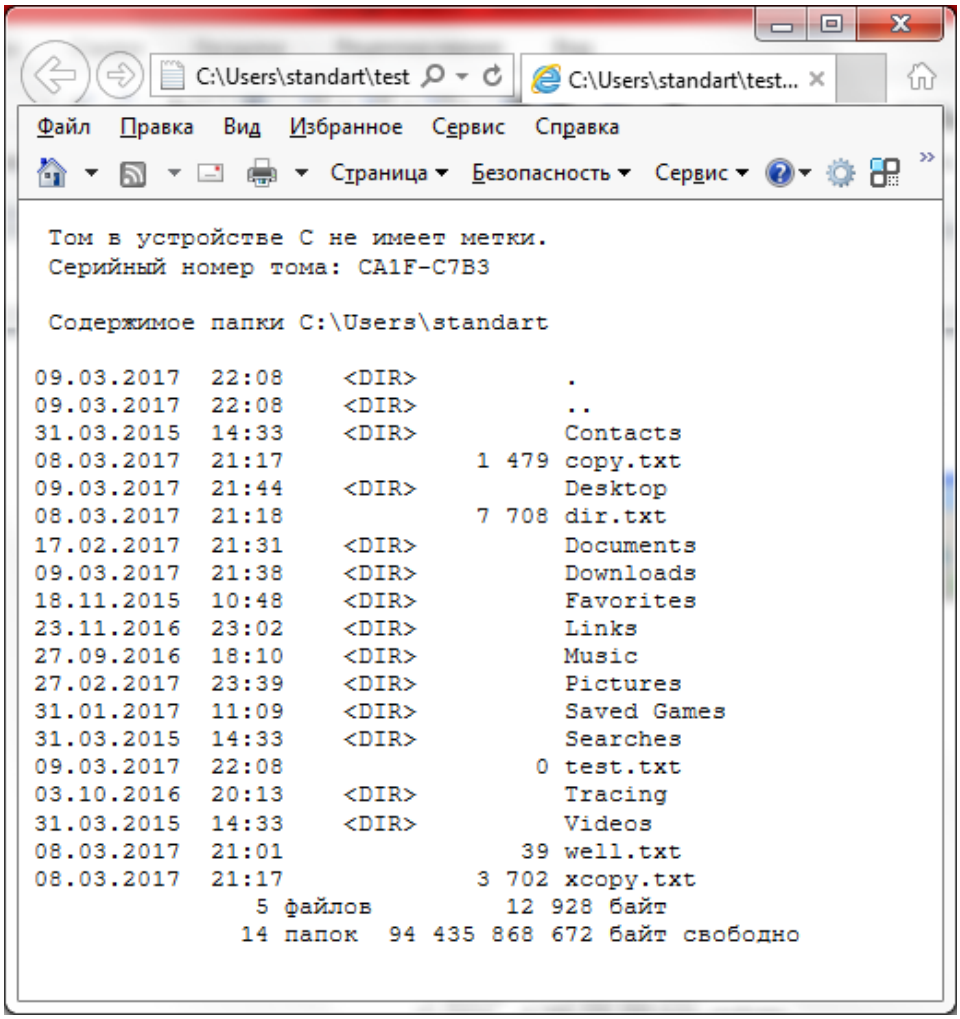

Рис.3 – вывод содержимого текстового файла, полученного с помощью команды DIR, в IE в кодировке кириллица (DOS)

5. Создать текстовый файл, содержащий справочные сведения по командам DIR, COPY и XCOPY.
6. Вывести содержимое указанного в табл.2 каталога по указанному формату на экран и в файл.

Таблица 2.

Варианты заданий для бригад

Номера бригад	Имя каталога	Что выводить	Сортировать по	Атрибуты фай-лов и каталогов
1, 6	%Windows%	Только файлы	По размеру	Системный
2, 7	%Windows%	Файлы и подкаталоги	По дате	Скрытый
3, 8	%Windows%	Только подкаталоги	Именам	Только чтение

Номера бригад	Имя каталога	Что выводить	Сортировать по	Атрибуты фай-лов и каталогов
4, 9	%Windows% и все подкаталоги	Только файлы bmp	По размеру	Только чтение
5, 10	%Windows% и все подкаталоги	Только файлы jpg	Именам	Любые

7. Скопировать все имеющиеся в каталоге Windows растровые графические файлы в каталог WinGrafika на диске C:. Если диск C: недоступен, использовать любой другой доступный диск.
8. Скопировать все имеющиеся в каталоге Windows исполняемые файлы в каталог WinEx на диске C:. Если диск C: недоступен, использовать любой другой доступный диск.

3 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Достоинства и недостатки интерфейса командной строки.
2. Инструменты командной строки для автоматизации работы в ОС Microsoft Windows.
3. Настраиваемые свойства интерпретатора.
4. Различие между внутренними и внешними командами. Примеры внешних и внутренних команд.
5. Структура команды интерпретатора.
6. Получение информации о конкретной команде.
7. Групповые символы (шаблоны) и их использование.
8. Перенаправление ввода/вывода и конвейеризация команд.
9. Условное выполнение и группировка команд.
10. Назначение символов &, &&, || и () .
11. Команды для работы с файловой системой – названия и возможности.
12. Достоинства и недостатки команд COPY и XCOPY.
13. Назначение команды ECHO и примеры ее использования.
14. Команда DIR и ее возможности.
15. В какой кодировке интерпретатор выводит информацию и как получить читаемую твердую копию?