**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное учреждение Московской области «Люберецкий техникум имени Героя Советского Союза, летчика-космонавта Ю.А.Гагарина»**

**КУРСОВАЯ РАБОТА**

**Ефремова Егора Михайловича**

**МДК 01.01 Операционные системы  
По теме: разработка модели интерпретатора командной строки ОС Windows**

Выполнил/а студент Ефремов Егор Михайлович

(подпись) (ФИО полностью)

Руководитель Тарджиманян Лия Николаевна

(подпись) (ФИО полностью)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дзержинский 2023

**ВВЕДЕНИЕ**……..……………………………….…...………………...................3

**ГЛАВА 1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ**.......................................................5

1.1 Оболочка (интерпретатор) командной строки.........................................5 1.1.2 Поддержка языков сценариев. Сервер сценариев Windows................6 1.1.3 Командная оболочка Microsoft...............................................................8

1.2 Оболочка командной строки Windows. Интерпретатор..........................9  
 1.2.1 Запуск оболочки…………………………………………………...........9  
 1.2.2 Настройка параметров командного окна интерпретатора.................10  
 1.2.3 Внутренние и внешние команды. Структура команд…….................11   
 1.2.4 Перенаправление ввода/вывода и конвейеризация команд...............13  
 1.3 Пути к объектам файловой системы………………………...................15

**ГЛАВА 2. МЕТОДИКА**......................................................................................17 **ЗАКЛЮЧЕНИЕ**………………………………………..……….…..…..............19

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**…..….…..…….............20

**ВВЕДЕНИЕ**

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 длярешения определенной задачи, а для этого желательно иметь четкое представление о сильных и слабых сторонах данных средств автоматизации. Рассмотрим достоинства и недостатки каждого

из них.

**Глава1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ  
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 возникающиеповседневные 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.

Даются примеры команд для работы с файловой системой

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 NT/2000/XP 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.

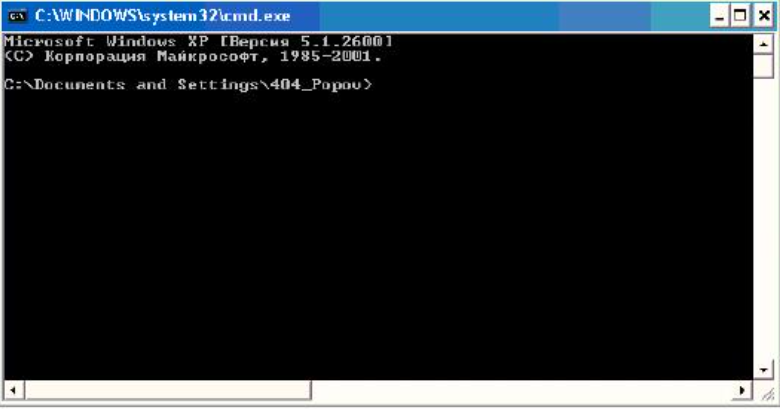


Рис. 1 - Командное окно интерпретатора Cmd.exe в Windows XP

**1.2.2 Настройка параметров командного окна интерпретатора**

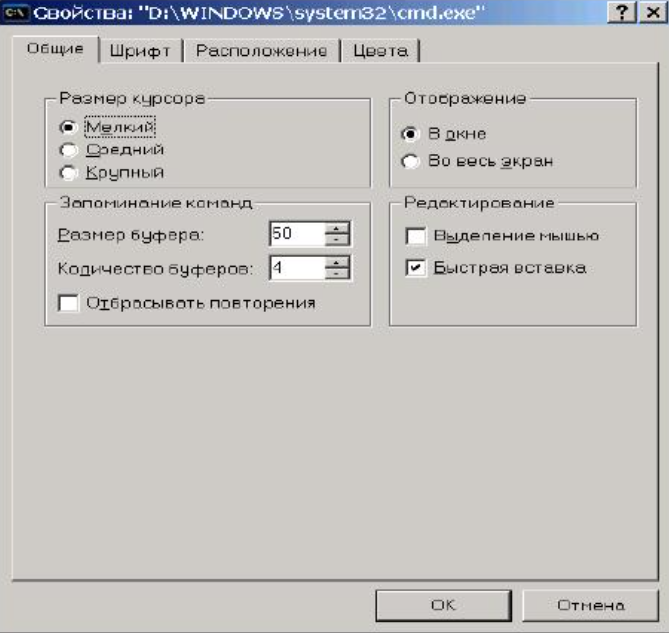
У утилиты командной строки, которая поставляется в виде стандартного приложения ОС

Windows, имеется свой набор опций и параметров настройки. Один из способов просмотра этих

опций – использование п ункта 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 . 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 донажатия 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.

1. Вывод встроенной справки для команды COPY в файл copy.txt:

COPY /? > copy.txt

2. Добавление текста справки для команды XCOPY в файл copy.txt:

XCOPY /? >> copy.txt

3. 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 конструкциикоманда1 | 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.

1. Удаление всех файлов в текущем каталоге без предупреждения (автоматическийположительный ответ на запрос об удалении):

ECHO y | DEL \*.\*

**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

**Глава2. 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

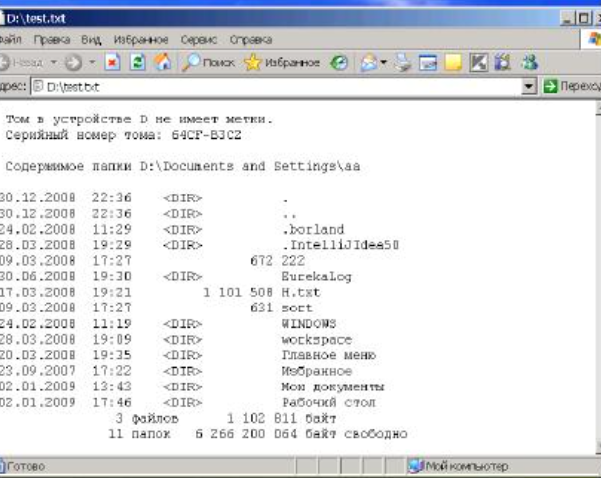
синий фон и белый шрифт).

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 переназначать 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).

1. Для просмотра содержимого буфера используйте клавиши клавиатуры СТРЕЛКА ВВЕРХ
2. 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. Скопировать все имеющиеся в каталоге Windows растровые графические файлы в каталог WinGrafika на диске С:. Если диск С: недоступен, использовать любой другой доступный диск.

7. Скопировать все имеющиеся в каталоге Windows исполняемые файлы в каталог WinEx на диске С:. Если диск С: недоступен, использовать любой другой доступный диск.

**ЗАКЛЮЧЕНИЕ**

С помощью информации, собранной в сети Интернет, мы ознакомились с возможностями интерпретатора командной строки (cmd) и командами операционной системы MS Windows

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ:**

# Интерпретатор командной строки ОС MS Windows XP (дата просмотра 22.04.2023) - <https://ru.readkong.com/page/interpretator-komandnoy-stroki-os-ms-windows-xp-6011776>

1. Оболочка командной строки Windows. Интерпретатор Cmd.exe (дата просмотра 20.04.2023) - <https://intuit.ru/studies/courses/1059/225/lecture/27285>

# Тяжкое наследие прошлого. Проблемы командной строки Windows (дата просмотра 29.04.2023) - <https://habr.com/ru/articles/417679/>