МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

ОТЧЕТ

по лабораторной работе №2

«Командный язык ОС Windows и командная оболочка Windows PowerShell»

по дисциплине

«Принципы и методы организации системных

программных средств»

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_ Викулова Е.Н.

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Тихонов В. А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Игнаков К. М.

(подпись) (фамилия, и.,о.)

\_\_\_\_\_\_19-В-2\_\_\_\_\_\_\_\_

(шифр группы)

Работа защищена «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2022

# Цель работы

* Ознакомиться с эволюцией, достоинствами, недостатками, особенностями командных оболочек от Microsoft
* Изучить состав команд командного процессора cmd.exe
* Изучить состав и функциональные возможности MS PowerShell

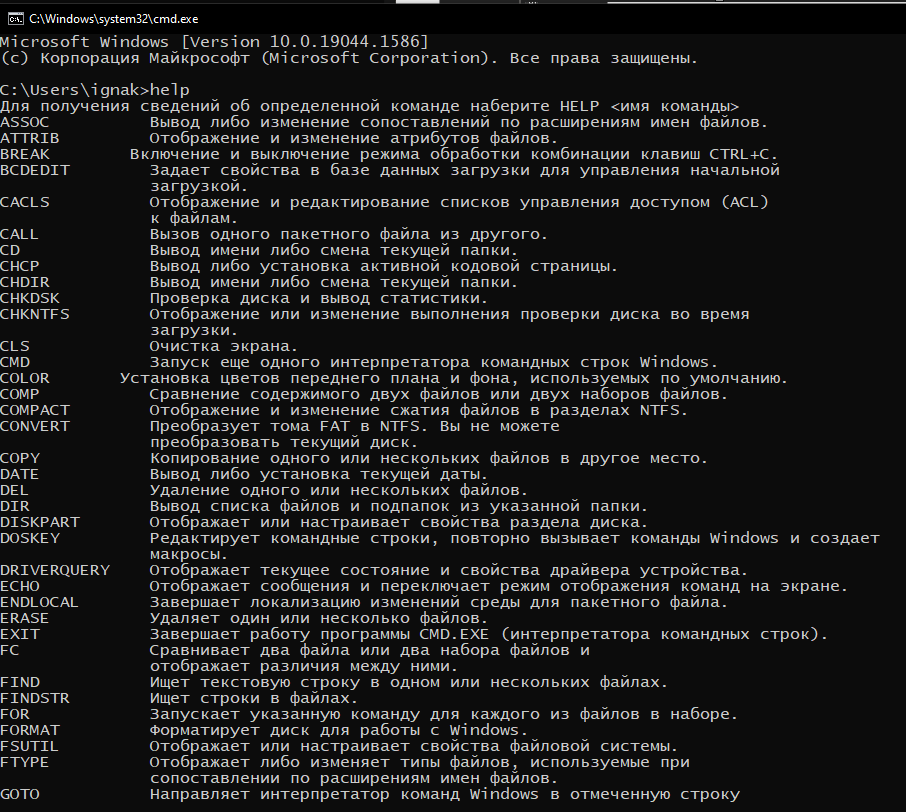
# Краткие теоретические сведения

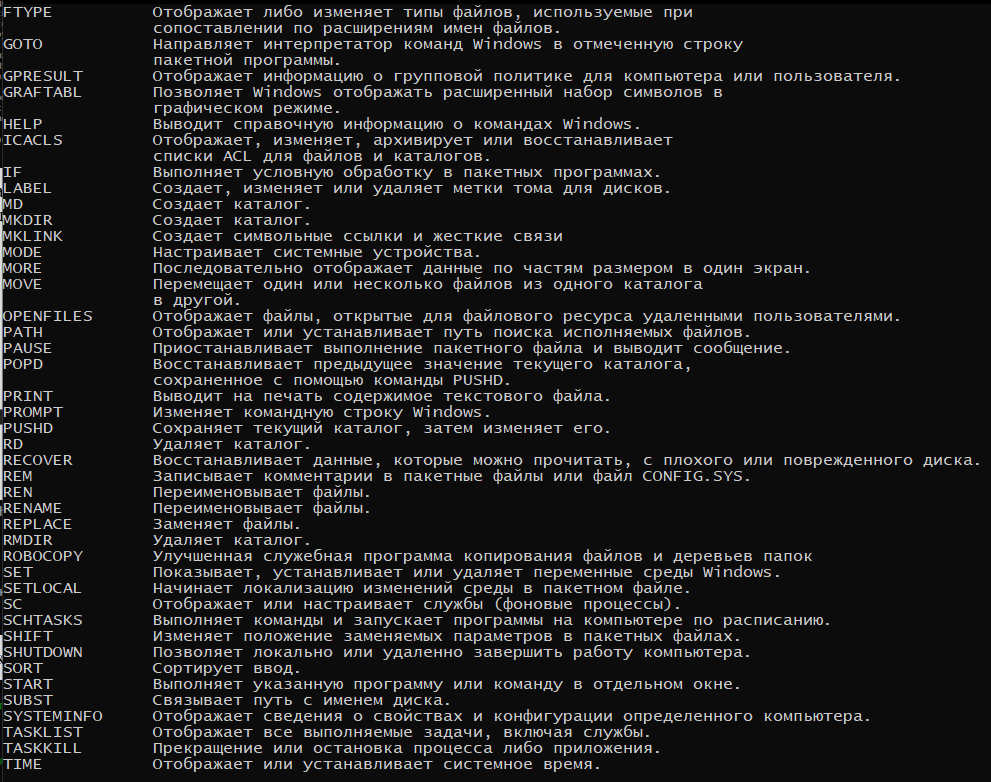
CMD — это интерпретатор командной строки. Командная строка позволяет пользователю ПК, путем ввода различных команд - осуществлять базовые действия в ОС. Например, скопировать/удалить файл, создать папку, посмотреть или поменять текущую дату и многое, многое другое.

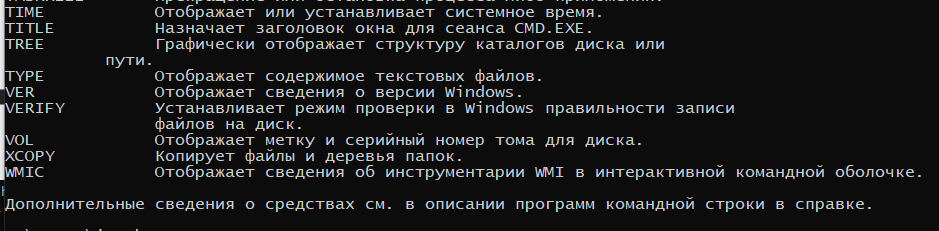
Windows PowerShell — это и командная оболочка, и язык сценариев, обеспечивающий более глубокую интеграцию с ОС Windows. Он основан на платформе Microsoft .Net, что обеспечивает доступ к различным уже существующим инструментам и функциям. Он использует текстовые команды, известные как командлеты и язык программирования C #, помогая пользователю более эффективно управлять инфраструктурой Windows.

В итоге, PowerShell – это улучшенная версия cmd, которая сочетает в себе функциональность командной строки с мощной средой сценариев для упрощения администрирования системы.

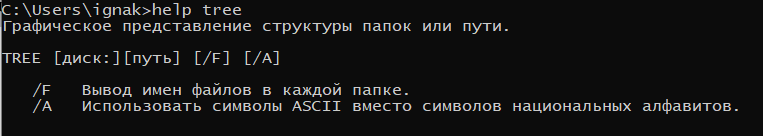
# Набор команд cmd.exe

Для получения полного списка команд, доступных нам в cmd воспользуемся командой **HELP**

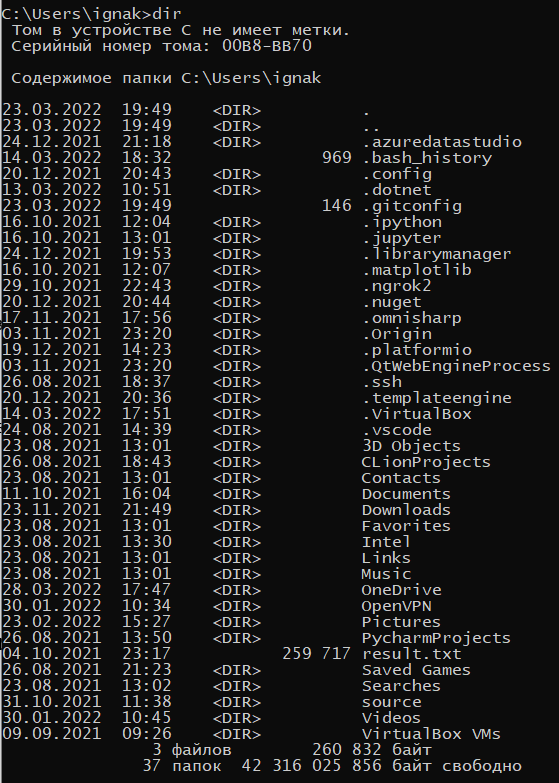




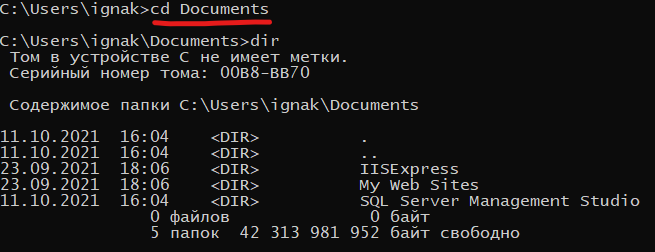
Если же нужно получить справку по конкретной команде, то можно использовать **HELP** [<команда>]



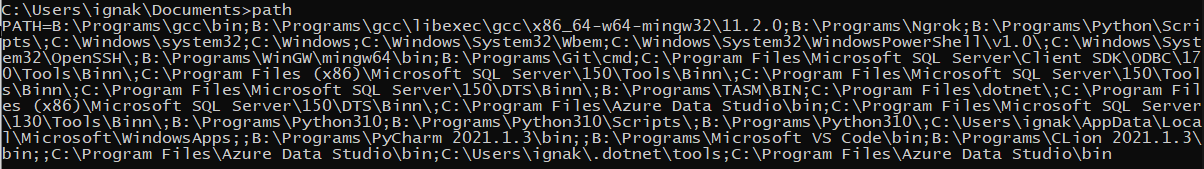
Чтобы получить список всех файлов и папок в текущей директории можно воспользоваться командой **DIR**.



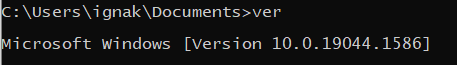
Что бы перейти в папку, достаточно вызвать команду **CD** <имя папки>.



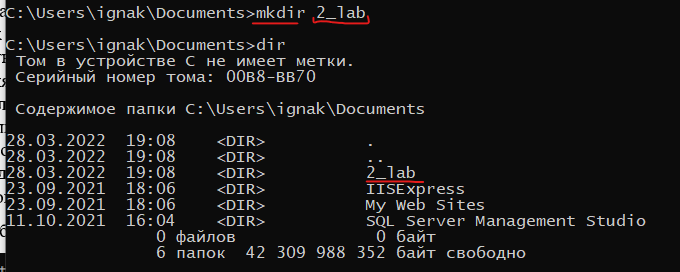
Для того, чтобы получить информацию о всех путях, которые есть в переменной среде PATH, можно узнать вызвав **PATH**.



Чтобы узнать версию Windows – **VER**.



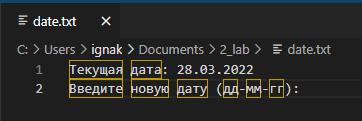
Чтобы создать новую директорию нужно использовать команду **MKDIR <имя\_каталога>**.



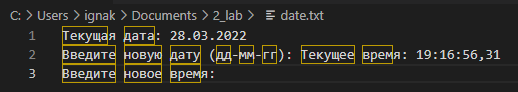
Чтобы перенаправить вывод из консоли, например, в txt файл можно воспользоваться стандартной сменой УВВ: **[command] > [file\_name]**.



Содержимое файла date.txt:



Чтобы переписать содержимое файла, можно выполнить эту команду снова, а что бы дописать в конец файла еще данные, можно воспользоваться такой структурой: **[command] >> [exist\_file\_name]**:



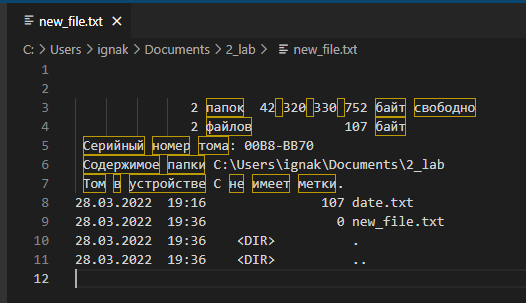
Второй вариант перенаправления – конвейеры. Они нужны для того, чтобы перенаправить вывод одной команды на вход другой. Конвейеры удобно использовать вместе с фильтрами.

Фильтры – это программа, принимающая на вход информационный поток, преобразующая его по определенному алгоритму и выводящая результат работы.

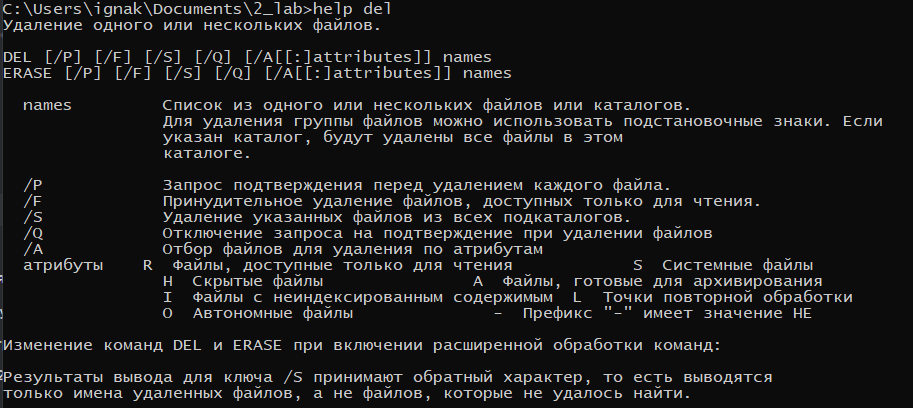
В командной строке существуют следующие фильтры:

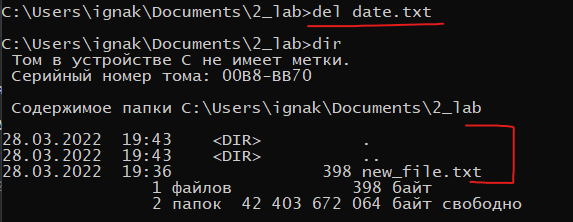
1. **SORT** – сортирует файлы или выводы команд по алфавиту.
2. **MORE** – отображает содержимое файла или вывода команды в одном окне командной строки за раз.
3. **FIND** – поиск указанных символов в файлах и выходе команды.

Пример использования конвейера с перенаправлением и фильтром в одной команде: **dir | sort > new\_file.txt**



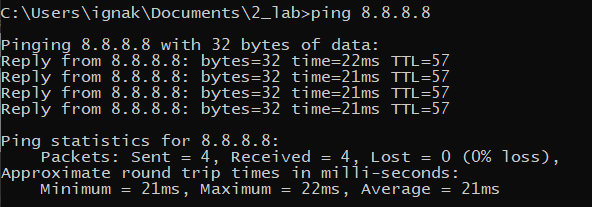
Чтобы удалить файл, можно воспользоваться командой **DEL**.



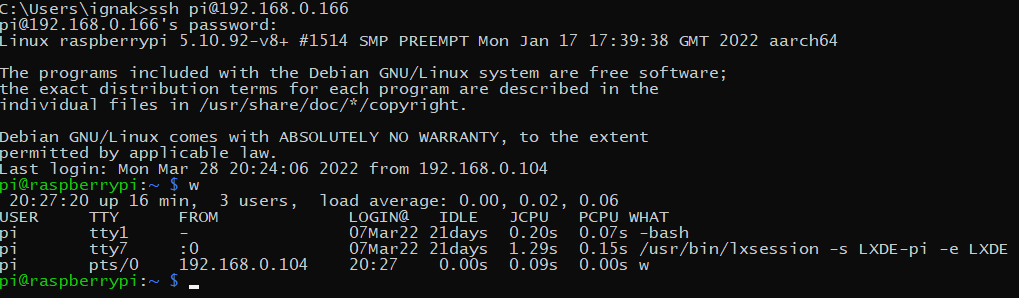


# Пример использования командного языка для решения практической задачи

Для проверки соединения с интернетом используется команда **PING** и аргументом передается ip адрес какого-то сервера или url адрес. В данном примере 8.8.8.8 это DNS сервер от компании Alphabet.



Для подключения к компьютеру, который находится в локальной сети по протоколу ssh используется команда **SSH**:



MS PowerShell

Windows PowerShell — это и командная оболочка, и язык сценариев, обеспечивающий более глубокую интеграцию с ОС Windows. Он основан на платформе Microsoft .Net, что обеспечивает доступ к различным уже существующим инструментам и функциям. Он использует текстовые команды, известные как командлеты и язык программирования C #, помогая пользователю более эффективно управлять инфраструктурой Windows.

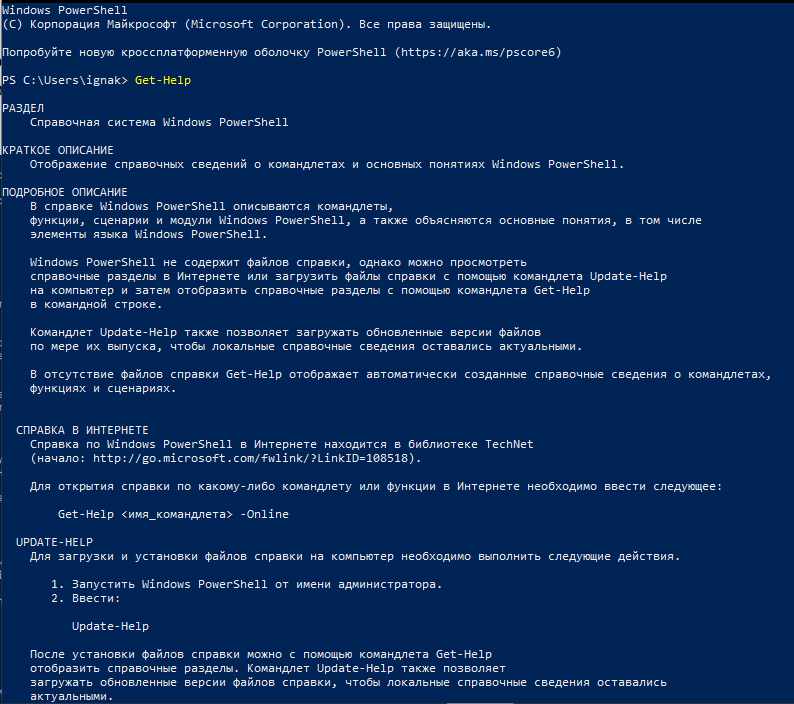
В итоге, PowerShell – это улучшенная версия cmd, которая сочетает в себе функциональность командной строки с мощной средой сценариев для упрощения администрирования системы.

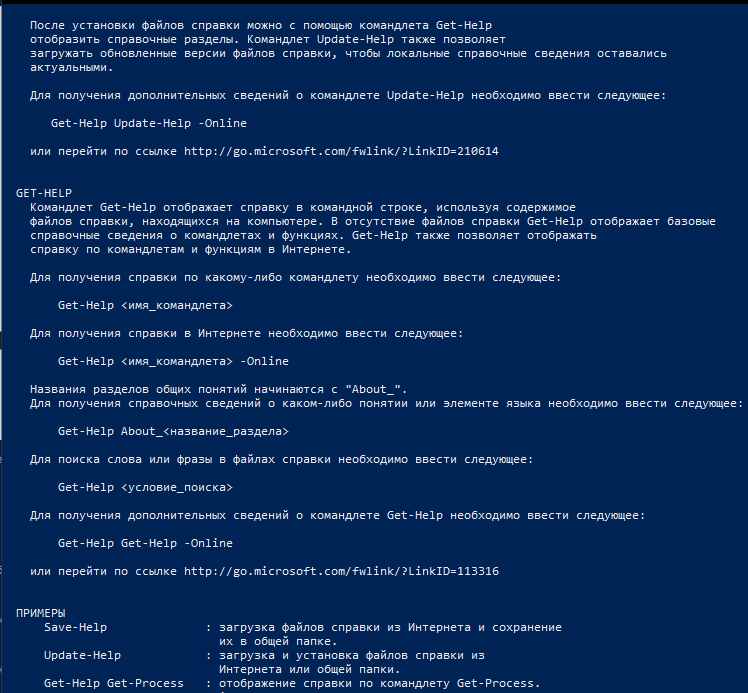
Командлет — это команда с одним компонентом, производящая действия с объектами в PowerShell.

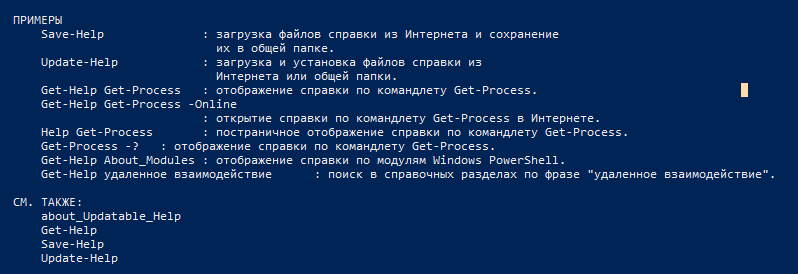
Командлеты можно распознать по формату имени, которое представляет собой глагол и существительное, разделенные дефисом (-), например, Get-Help, Get-Process, и Start-Service. Командлеты должны быть простыми и предназначены для использования в сочетании с другими командлетами. Например, командлеты Get только получают данные, командлеты Set только устанавливают или изменяют данные, командлеты Format только форматируют данные, а командлеты Out только перенаправляют выходные данные в указанное место назначения.

* **Get-Help** – показывает справку по командлету, функции и общую справку по Windows PowerShell. Справка бывает нескольких типов: краткая, детальная, полная и вывод только примеров;
* **Update-Help** — загружает и устанавливает новые файлы справки, т.е. обновление справки;
* **Get-Command** – командлет поиска нужной команды, можно искать как по глаголу, так и по существительному, также возможно использование маски, если Вы не знаете точное наименование глагола или существительного;
* **Get-Alias** – показывает псевдонимы, все или конкретной команды;
* **Get-PSDrive** – показывает подключенные диски;
* **Get-Member** – выводит свойства и методы, которые есть у объекта;
* **Get-History** — возвращает список команд, введенных в ходе текущей сессии.

В качестве примера рассмотрим командлет **Get-Help**, который отображает информацию о концепциях и командах PowerShell, включая командлеты, функции, команды общей информационной модели (CIM), рабочие процессы, поставщиков, псевдонимы и сценарии.





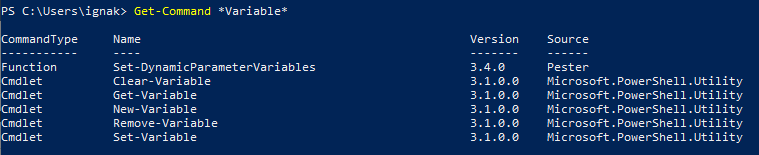


В справке появилась информация полная информация по данному командлету.

Переменные в PowerShell

В Windows PowerShell есть как встроенные переменные, так и переменные, которые может создавать пользователь. Особенностью переменных в PowerShell является то, что они могут менять свой тип в зависимости от значения этой переменной, но это только случае, если при объявлении переменной не был указан её тип.

Переменные живут в рамках текущего сеанса. Если вы работаете через консоль - после ее закрытия они удаляться, если это работающий скрипт - до закрытия программы. Исключения только с системными переменными и импортируемыми модулями. Одна из таких переменных, которая создается каждый сеанс $PSVersionTable, хранящая версию Powershell и ее не получится изменить или создать заново.



**Get-Variable** - позволяет получить значение переменной со всеми дополнительными параметрами. Если не указать название переменной, командлет выведет список всех текущих переменных *Powershell*.

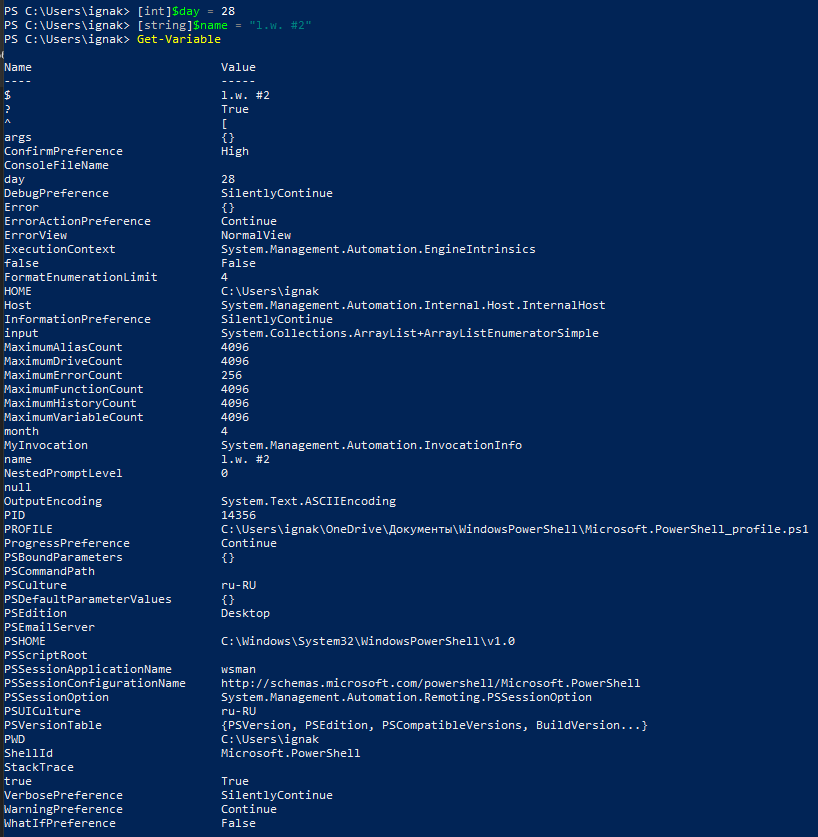
**New-Variable** - создает новую переменную с возможностью задать дополнительные параметры. Можно создать переменную только для чтения, скрытую переменную или переменную с пробелом в имени.

**Set-Variable** - изменение значений в переменной. В случае отсутствия переменной с указанным именем, она будет создана.

**Clear-Variable** - удаляет все значения в переменной, но не саму переменную. Тип переменой остается тот же что был до очистки значений.

**Remove-Variable** - удаляет переменную со всеми ее значениями.

Для примера создадим несколько своих переменных и выведем список всех переменных:



Форматирование в PowerShell

В Windows PowerShell существует набор командлетов, которые предназначены для форматирования вывода результата работы командлета. Они позволяют пользователю отобразить результат в том виде, в котором ему удобно просматривать данный результат.

**Format-List** – вывод результата команды в формате списка свойств, где на каждой новой строке отдельное свойство;

**Format-Table** — вывод результата команды в виде таблицы;

**Format-Wide** — вывод результата команды в виде широкой таблицы, в которой отображается только одно свойство каждого объекта;

**Format-Custom** – в данном случае форматирование вывода происходит с использованием пользовательского представления.

Процессы в PowerShell

Командлеты Process в Windows PowerShell позволяют управлять локальными и удаленными процессами в Windows PowerShell.

1. **Get-Process** – позволяет получить все запущенные процессы.
2. **Stop-Process** – позволяет остановить указанный процесс.

Работа с объектами

Так как PowerShell работает с объектами, он позволяет выполнять некие действия над этими объектами, например:

**Measure-Object** – командлет позволяет рассчитывать на основе свойств объектов такие числовые агрегирующие параметры как: минимальное, максимальное, среднее значение, сумму и количество. Например, Вы хотите узнать максимальный или средний размер файла в определенном каталоге, или просто узнать количество файлов (запущенных процессов, служб и так далее);

**Select-Object** – с помощью данного командлета можно выбрать определенные объекты или свойства этих объектов, например Вы хотите выводить только название файла и его размер;

**Sort-Object** — сортирует объекты по значениям свойств;

**Where-Object**– командлет для того чтобы ставить условие для выборки объектов на основе значений их свойств;

**Group-Object** – группирует объекты, которые содержат одинаковое значение для заданных свойств;

**ForEach-Object** – перебор объектов с целью выполнения определенной операции над каждым из этих объектов.

Пример практического использования PowerShell

WSL (Windows Subsystem of Linux – Подсистема Windows для Linux)

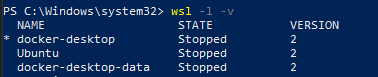
Подсистема Windows для Linux (WSL) позволяет разработчикам запускать среду GNU/Linux с большинством программ командной строки, служебных программ и приложений непосредственно в Windows без каких-либо изменений и необходимости использовать традиционную виртуальную машину или двойную загрузку.

Установка

Теперь вы можете установить все необходимые компоненты для запуска подсистемы Windows для Linux (WSL), введя эту команду в PowerShell от имени администратора или командной строке Windows, а затем перезапустив компьютер.

wsl --install

Вы можете получить список установленных распределений Linux и проверить версию WSL, задав команду **wsl -l -v** в PowerShell или в Командной строке Windows.



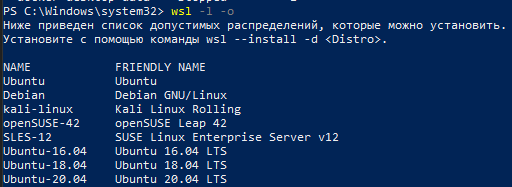
Изменение установленного дистрибутива Linux по умолчанию

По умолчанию в качестве устанавливаемого дистрибутива Linux используется Ubuntu. Для этого используйте флаг **-d**.

Чтобы изменить установленный дистрибутив, введите **wsl --install -d <Distribution Name>**. Замените <Distribution Name> именем дистрибутива, который хотите установить.

Чтобы просмотреть список доступных дистрибутивов Linux, доступных для скачивания через Интернет-магазин, введите **wsl --list --online** или **wsl -l -o**.

Чтобы установить дополнительные дистрибутивы Linux после исходной установки, можно также использовать команду **wsl --install -d <Distribution Name>**.



Вывод

В ходе выполнения данной лабораторной работы мы ознакомились с командными оболочками cmd.exe и MS PowerShell.

В cmd.exe научились выполнять команды различных типов, перенаправлять потоки вывода, использовать группировку команд и условное выполнение. В результате был реализован пример практического использования cmd.exe: проверка интернет подключения и подключение к удаленному компьютеру по ssh.

В MS PowerShell научились выполнять командлеты, работать с объектами, применять сценарии и конвейеры. В результате был реализован пример практического использования MS PowerShell: настройка wsl.

Командный процессор cmd.exe проще в изучении и позволяет вывести результаты работы команд обычным текстом. Также он удобен для выполнения небольшого количества команд и операций. MS PowerShell – более мощный инструмент, позволяющий работать с языками программирования и командлетами. Его удобно использовать при выполнении сложных операций и создании длинных сценариев.