

# Microsoft® PowerShell





# Функции

- Блок кода на языке PowerShell, имеющий название и находящийся в памяти до завершения текущего сеанса командной оболочки
- Анализ синтаксиса функции производится только один раз, при ее объявлении (при повторном запуске функции подобный анализ не производится)



Функция — набор команд, имеющее название и находящееся в памяти до завершения текущего сеанса.

```
        function
        имя_функции

        {
        тело_функции

        }
```



### Вызов функции:

имя\_функции



### Пример:

Написать функцию, которая приветсвует пользователя.

```
function Hello{
"Добрый день!"
}
```

Вызов функции:

Hello



### Оператор ввода информации:

```
$переменная = Read-Host "сообщение"

На консоль выводится сообщение, а введенное значение помещается в $переменная.
```

### Пример:

```
$a_1 = Read-Host "Введите строку" $a_1
```



#### **ЗАДАНИЕ**

Изменить функцию **Hello**, чтобы она запрашивала у пользователя его имя и вводила:

Добрый день, имя\_пользователя !!!



Вывод списка функции:

dir Function:

Вывод содержимого функции:

type Function:имя\_функции

Пример:

type Function:Hello



#### **ЗАДАНИЕ**

#### Написать функцию:

- 1) Узнает имя пользователя и приветствует его.
- 2) Ввести числовое значения двух переменных.
- 3) Вывести на консоль результат деления первого числа на второе.



#### **ЗАДАНИЕ**

Написать функцию:

- 1) Узнает имя пользователя и приветствует его.
- 2) Ввести числовое значения двух переменных.
- 3) Вывести в файл строку:

```
число_1 / число_2 = результат_деления
```



#### **ЗАДАНИЕ**

#### Написать функцию:

- 1) Узнает имя пользователя и приветствует его.
- 2) Ввести числовое значения двух переменных.
- 3) Вывести в файл шесть строк показывающие все операции сравнения с этими числами:

 $число_1 on_cp число_2 = pезультат_cpавнения$ 

#### Например:

$$15 - eq 25 = True$$



В процессе работы в среде PowerShell можно вместо значений подставлять «шаблон». Предполагается, что «шаблон» заменяет все значения удовлетворяющие «шаблону».

Шаблон – это символьная строка, в которой присутствуют специальные символы:

- \* любое количество (может быть нулевым) произвольных символов;
  - ? один произвольный символ.



Имеется операция «сравнения с шаблоном»:

-like

#### Пример:

```
$w1 = '*x*'

'zxz'-like $w1 истина

'zzz'-like $w1 ложно
```



- 1) Занести в переменную шаблон '?а\*'.
- 2) Сравнить с шаблоном следующие строки:
- а, пустая строка, ba, ab, abc, babc, lavatar



- 1) Создать текстовый файл на диске PS с расширением **TXT**. В этот файл вывести содержимое корневого каталога диска **C**:
- 2) Сохранить в переменной **\$ps\_drive** список дисков.
- 3) Записать в конец созданного в п.1 файла содержимое из переменной **\$ps\_drive**
- 4) Вывести на экран содержимое созданного файла.



- 1) Создайте каталог с именем PowerShell.
- 2) В каталоге **PowerShell** создайте несколько текстовых файлов и поместите в них несколько строк.
- 3) Сохраните содержимое каталога PowerShell в переменной **\$ps\_dir**, а список дисков в переменной **\$ps drive**.
- 4) Выведите на экран содержимое переменных **sps dir** и **sps drive**.



- 1) Запустить программу Word: найти каталог, где расположен файл WinWord.exe и выполнить его.
- 2) Вывести на экран список выполняемых процессов (Get-Process) по маске имени, в которое входит слово word (\* word \*).
- 3) Остановить процесс WinWord (Stop-Process) по его идентификатору (id).



- 1) Вывести на экран список сервисов (Get-Sevice).
- 2) Посмотреть структуру объектов, описывающих сервисы (Get-Member).
- 3) Вывести информацию о 15 сервисах, упорядочив её по названию сервисов (Sort-Object). В выводимой информации отразить имя сервиса, статус, DisplayName (Select-Object).
- 4) Повторить п.3 с следующим ограничением: вывести на экран список сервисов, имена которых начинается на букву 'w' (маска w\*).



- 1) Вывести список всех выполняемых процессов. (Get-Process)
- 2) Вывести список всех выполняемых процессов, упорядочив его по убыванию процессорного времени. (Sort-Object)



- 1) Вывести список 10 процессов, которые максимально используют время центрального процессора. (Get-Process)
- 2) Занести этот список в переменную.(=)
- 3) Создать свою директорию.(New-Item) В ней создать файл в который записать содержимое из переменной п.2.(>)
- 4) Вывести на экран содержимое созданного файла.(type)



### Логические операции:

Операция	Описание	Пример	Результат
-and	Логическое И	True -and True	True
-or	Логическое ИЛИ	False -or False	False
-not	Логическое НЕ	-not False	True
!	Логическое НЕ	! True	False



#### **ЗАДАНИЕ**

Написать функцию:

- 1) Ввести числовое значения трех переменных.
- 2) Вывести в файл три строки показывающие следующие операции с этими числами:

```
uucno_1 < uucno_2  и uucno_2 < uucno_3 = peзультат
uucno_1 < uucno_2  или uucno_2 < uucno_3 = peзультат
uucno_1 < uucno_3 = peзультат
```

#### Например:

$$15 < 25$$
 и  $25 < -5$  = False

# Выбор свойств объекта



Результат работы командлеты – последовательность объектов.

Командлета Where-Object — выполняет перебор объектов из входного потока и проверку на заданное условие. Результатом будет последовательность объектов входного потока, которые удовлетворяют заданному условию.

В условии указывается очередной объект (\$\_) и через точку, свойство объекта, которое будет проверяться.

# Выбор свойств объекта



Фильтрация объектов по условия:

```
Where-Object { логическое условие }
Пример:
dir | Where-Object { $ .PSIsContainer }
dir | Where-Object { -not $ .PSIsContainer }
где $ – указывает на очередной объект входного
```

PSIsContainer – свойство объекта.

потока;



- 1) Вывести список всех сервисов (Get-Service).
- 2) Используя Where-Object вывести сервисы, которые остановлены.



- 1) Создать свой каталог с именем PowerShell.
- 2) Найти каталог, в котором имеются и подкаталоги и файлы.
- 3) Вывести в файл в каталоге PowerShell список всех подкаталогов из п.2.(dir, >, where)
- 4) В конец этого файла добавить список всех файлов из п.2. (dir, >, where)
- 5) Содержимое созданного файла вывести на экран. (type)





# Format-Table Работа с историей



- -Match сравнение по регулярному выражению
- -Notmatch не совпадает с регулярным выражением

# Вывод результатов в файл чиверсите НТМL

• Convertto-HTML например

Dir | Convertto-HTML | Out-File C:\Konkov\Examp\_1.html

Dir | Convertto-HTML > D:\Konkov\Examp\_1. html