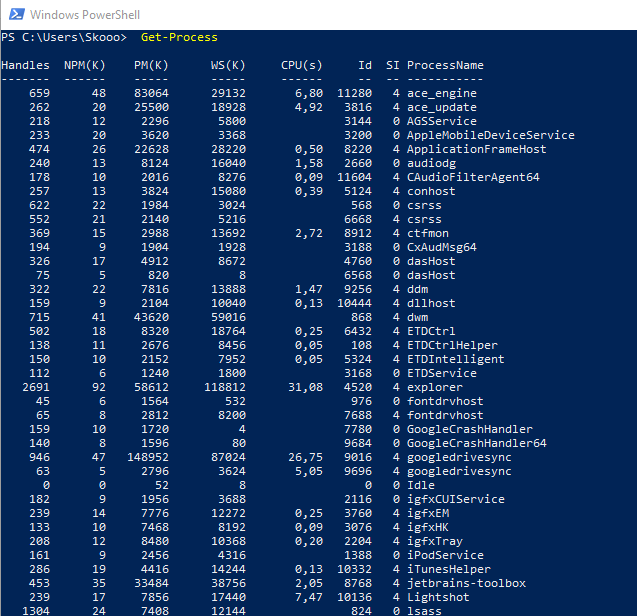
**Лабораторная работа 17**

**Конвейеризация и управление выводом команд Windows PowerShell**

**Конвейеризация объектов в PowerShell**

PS C:\> Get-Process

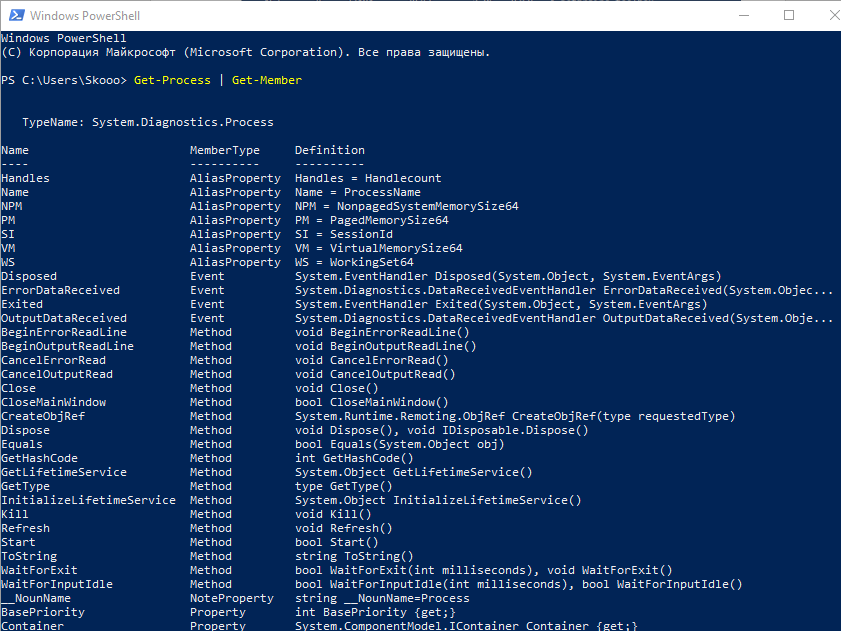


Фактически на экране мы видим только сводную информацию (результат форматирования полученных данных), а не полное представление выходного объекта. Из этой информации не понятно, сколько точно свойств имеется у объектов, генерируемых командой Get-Process, и какие имена имеют эти свойства.

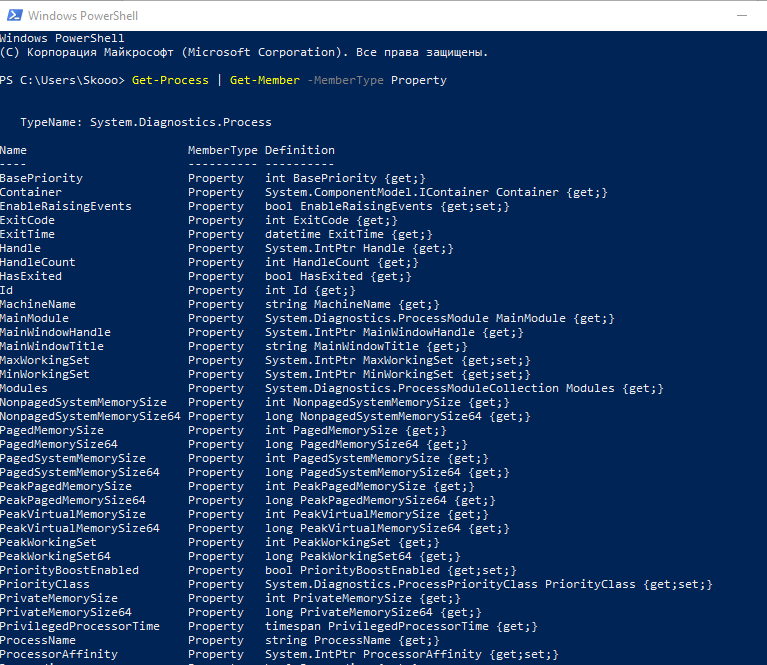
**Просмотр структуры объектов**

Для анализа структуры объекта, возвращаемого определенной командой, проще всего направить этот объект по конвейеру на командлет Get-Member (псевдоним gm ), например:

PS C:\> Get-Process | Get-Member



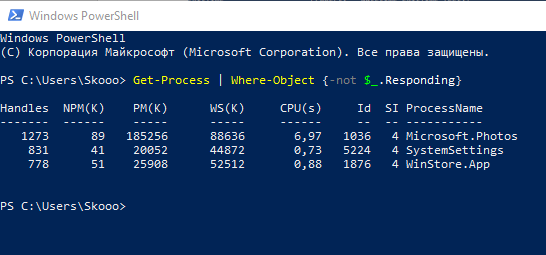
PS C:\> Get-Process | Get-Member -MemberType Property



**Фильтрация объектов в конвейере**

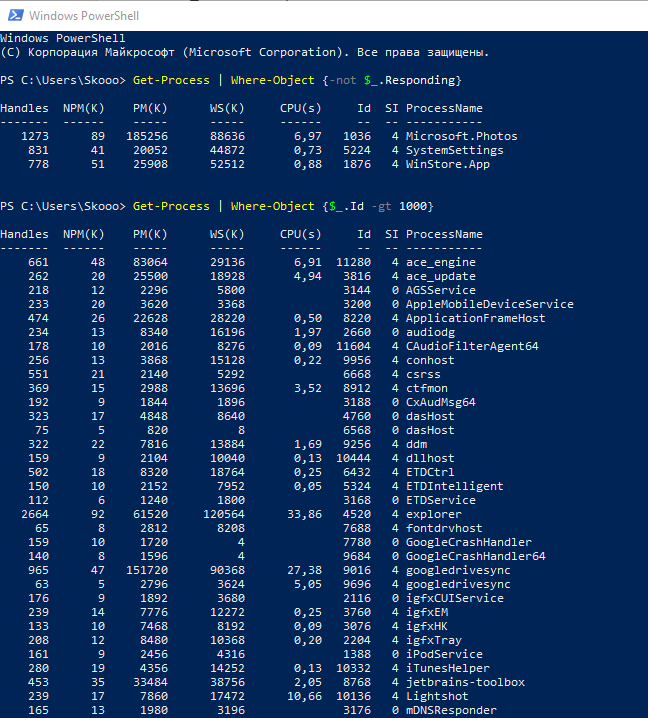
Для вывода информации о "зависших" процессах (объекты, возвращаемые командлетом Get-Process, у которых свойство Responding равно False ) можно использовать следующий конвейер:

Get-Process | Where-Object {-not $\_.Responding}



Другой пример – оставим в конвейере только те процессы, у которых значение идентификатора (свойство Id ) больше 1000:

Get-Process | Where-Object {$\_.Id -gt 1000}



|  |  |  |
| --- | --- | --- |
| Таблица 17.1. Операторы сравнения в PowerShell | | |
| **Оператор** | **Значение** | **Пример (возвращается значение True)** |
| -eq | равно | 10 -eq 10 |
| -ne | не равно | 9 -ne 10 |
| -lt | меньше | 3 -lt 4 |
| -le | меньше или равно | 3 –le 4 |
| -gt | больше | 4 -gt 3 |
| -ge | больше или равно | 4 -ge 3 |
| -like | сравнение на совпадение с учетом подстановочного знака в тексте | "file.doc" –like "f\*.doc" |
| -notlike | сравнение на несовпадение с учетом подстановочного знака в тексте | "file.doc" –notlike "f\*.rtf" |
| -contains | содержит | 1,2,3 –contains 1 |
| -notcontains | не содержит | 1,2,3 –notcontains 4 |

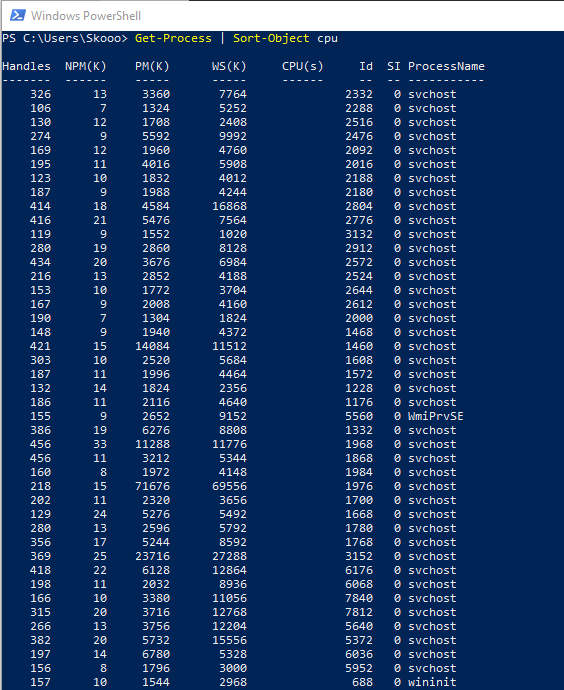
Операторы сравнения можно соединять друг с другом с помощью логических операторов (см. табл. 17.2).

|  |  |  |
| --- | --- | --- |
| Таблица 17.2. Логически операторы в PowerShell | | |
| **Оператор** | **Значение** | **Пример (возвращается значение True)** |
| -and | логическое И | (10 -eq 10) –and (1 –eq 1) |
| -or | логическое ИЛИ | (9 -ne 10) –or (3 –eq 4) |
| -not | логическое НЕ | -not (3 –gt 4) |
| ! | логическое НЕ | !(3 -gt 4) |

**Сортировка объектов**

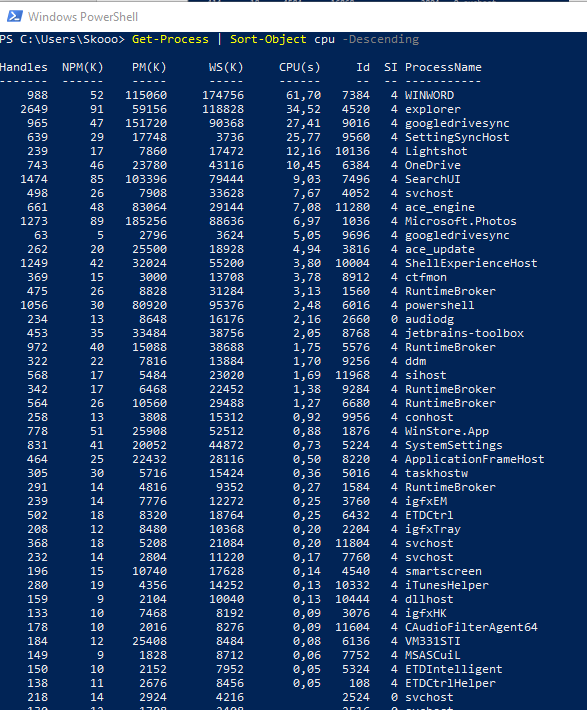
Например, для вывода списка запущенных в системе процессов, упорядоченного по затраченному процессорному времени (свойство cpu ), можно воспользоваться следующим конвейером:

PS C:\> Get-Process | Sort-Object cpu



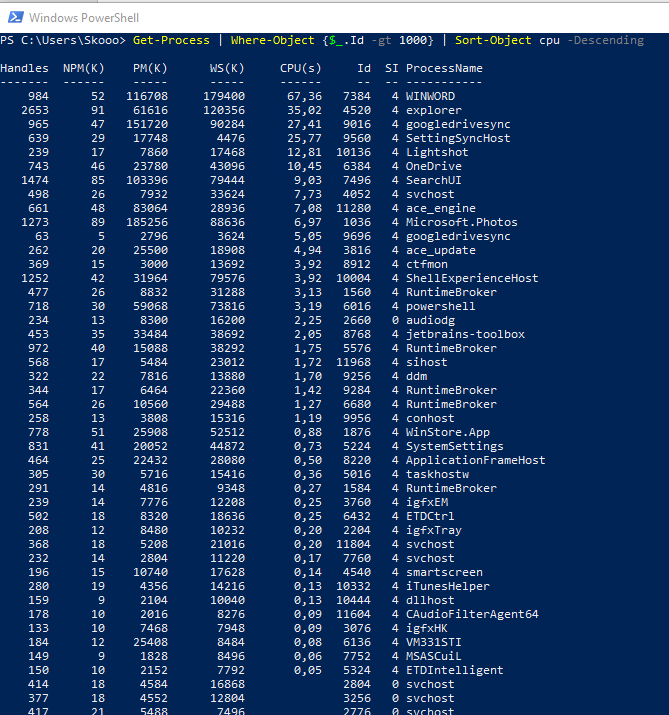
Для сортировки в обратном порядке используется параметр Descending:

PS C:\> Get-Process | Sort-Object cpu -Descending



В рассмотренных нами примерах конвейеры состояли из двух командлетов. Это не обязательное условие, конвейер может объединять и большее количество команд, например:

Get-Process | Where-Object {$\_.Id -gt 1000} | Sort-Object cpu -Descending

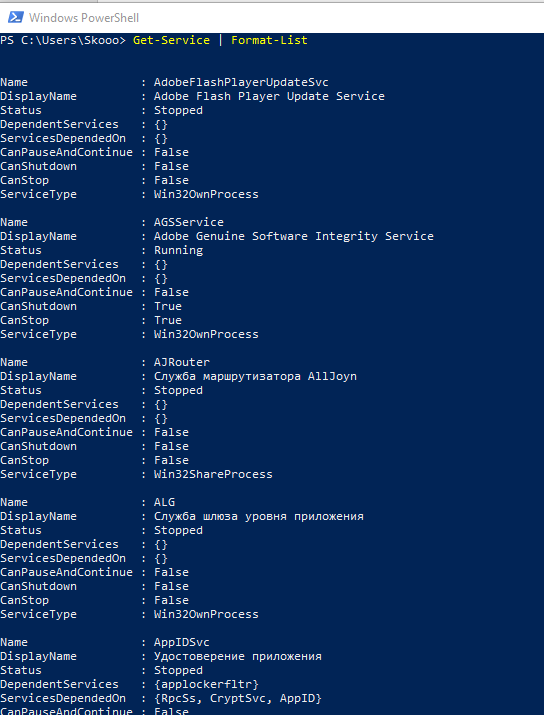


**Форматирование выводимой информации**

|  |  |
| --- | --- |
| Таблица 17.3. Командлеты PowerShell для форматирования вывода | |
| **Командлет** | **Описание** |
| Format-Table | Форматирует вывод команды в виде таблицы, столбцы которой содержат свойства объекта (также могут быть добавлены вычисляемые столбцы). Поддерживается возможность группировки выводимых данных |
| Format-List | Вывод форматируется как список свойств, в котором каждое свойство отображается на новой строке. Поддерживается возможность группировки выводимых данных |
| Format-Custom | Для форматирования вывода используется пользовательское представление (view) |
| Format-Wide | Форматирует объекты в виде широкой таблицы, в которой отображается только одно свойство каждого объекта |

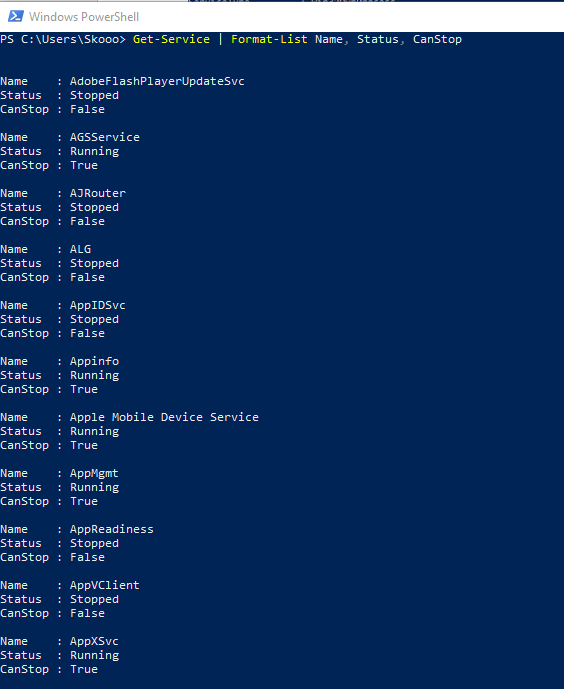
Для изменения формата выводимых данных нужно направить их по конвейеру соответствующему командлету Format. Например, следующая команда выведет список служб с помощью командлета Format-List:

PS C:\> Get-Service | Format-List

. . .

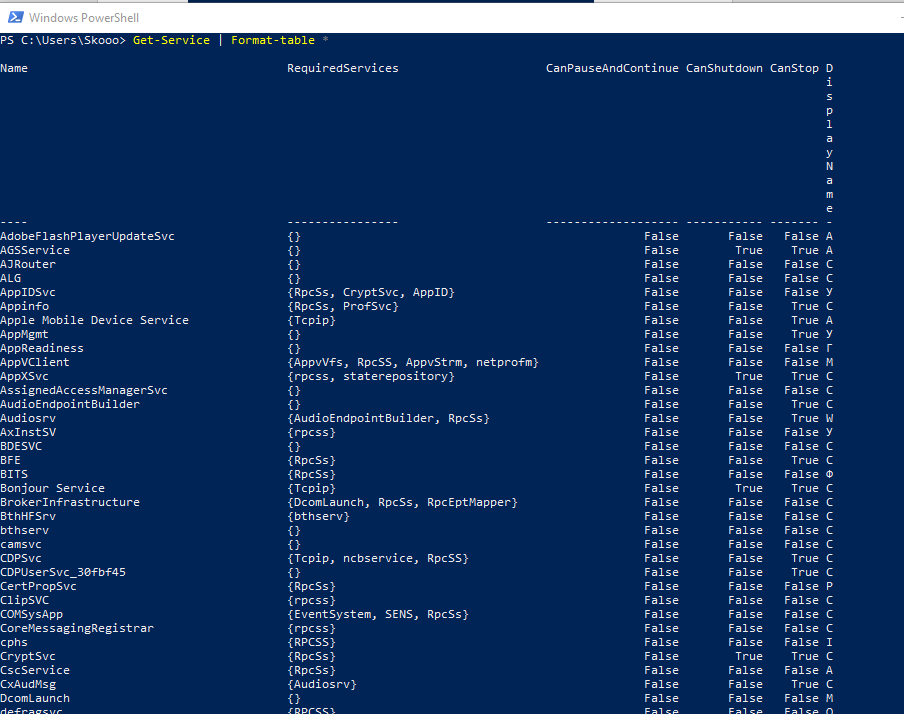
При форматировании вывода с помощью командлетов Format-List и Format-Table можно указывать имена свойства объекта, которые должны быть отображены (напомним, что просмотреть список свойств, имеющихся у объекта, позволяет рассмотренный ранее командлет Get-Member ). Например:

PS C:\> Get-Service | Format-List Name, Status, CanStop



Вывести все имеющиеся у объектов свойства можно с помощью параметра \*, например:

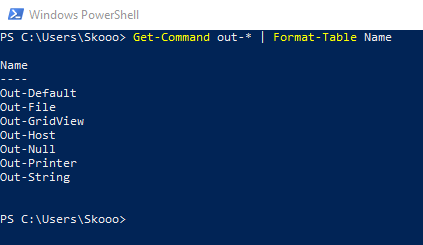
PS C:\> Get-Service | Format-table \*



**Перенаправление выводимой информации**

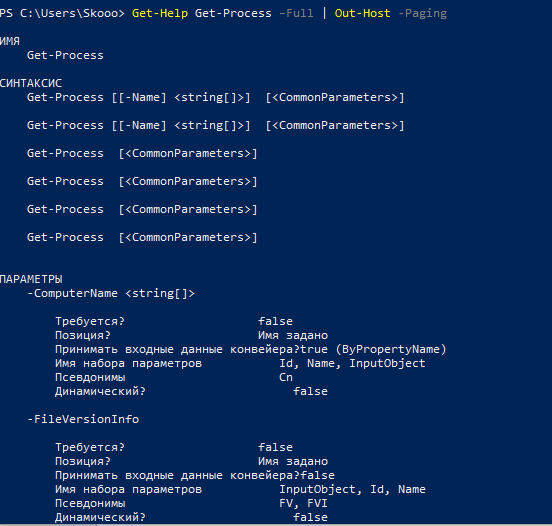
В оболочке PowerShell имеются несколько командлетов, с помощью которых можно управлять выводом данных. Эти командлеты начинаются со слова Out, их список можно увидеть следующим образом:

PS C:\> Get-Command out-\* | Format-Table Name



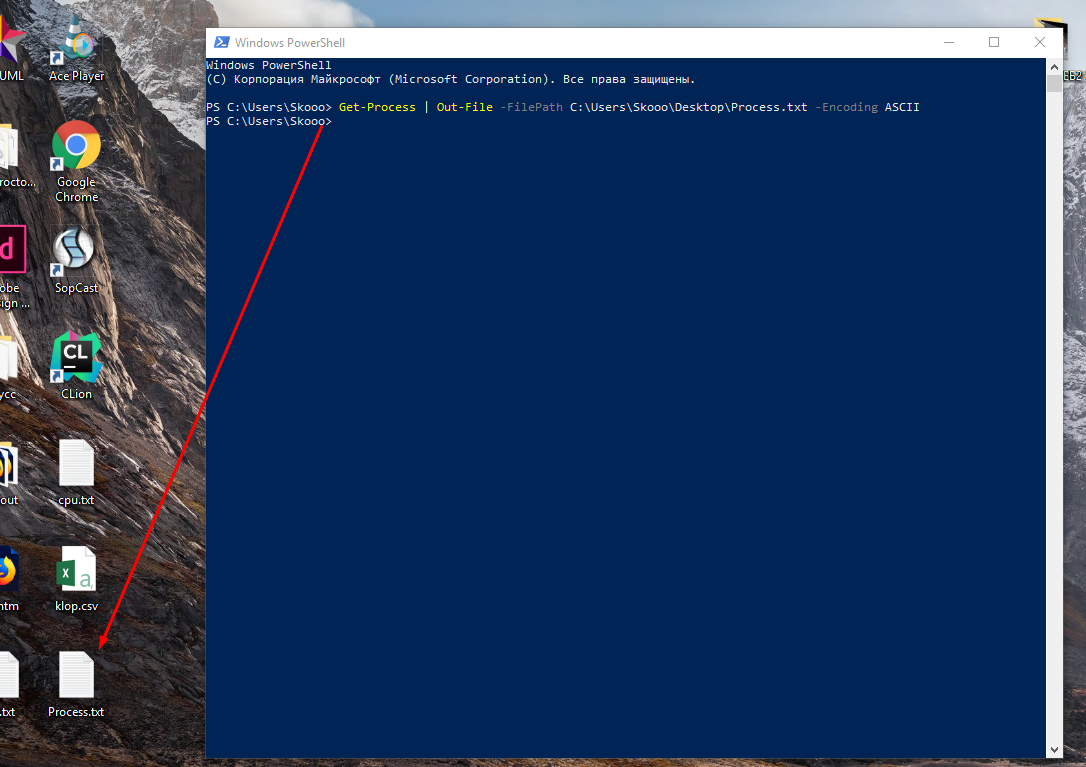
Параметр Paging командлета Out-Host, подобно команде more интерпретатора Cmd.exe, позволяет организовать постраничный вывод информации, например:

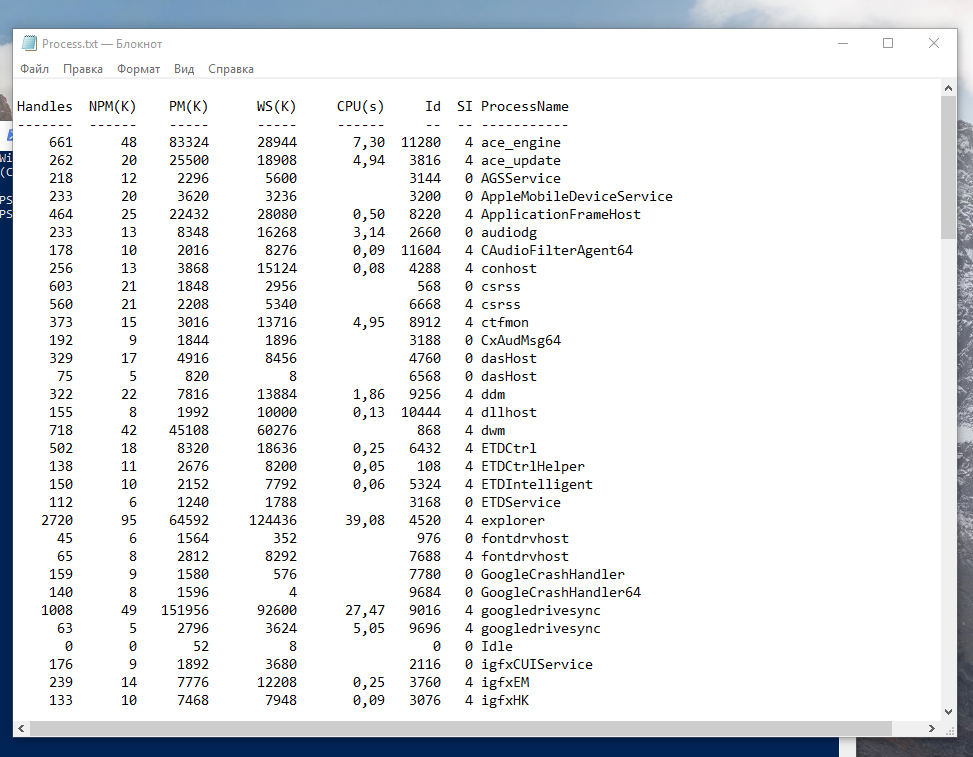
Get-Help Get-Process –Full | Out-Host –Paging



**Сохранение данных в файл**

Get-Process | Out-File -FilePath C:\Users\Skooo\Desktop\Process.txt -Encoding ASCII





**Печать данных**

Данные можно вывести непосредственно на принтер с помощью командлета Out-Printer. При этом печать может производиться как на принтере по умолчанию (никаких специальных параметров для этого указывать не нужно), так и на произвольном принтере (в этом случае отображаемое имя принтера должно быть указано в качестве значения параметра Name ). Например:

PS C:\script> Get-Process | Out-Printer -Name "Xerox Phaser 3500 PCL 6"

(необходимо изменить название принтера)

**Подавление вывода**

Командлет Out-Null служит для отбрасывания любых своих входных данных. Это может пригодиться для подавления вывода на экран ненужных сведений, полученных в качестве побочного эффекта выполнения какой-либо команды. Например, при создании каталога командой mkdir на экран выводится его содержимое:

PS C:\> mkdir klop

