МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт «Цифровых технологий управления и информационной безопасности»

Кафедра «Информационных технологий и системы»

**дисциплина «Операционные системы»**

**Контрольная работа № 2**

Студент: Бидаев Альфред Александрович

(подпись, инициалы, фамилия)

Группа: ПИЭ-21-1

Специальность: Прикладная Информатика

Преподаватель: Стариков Евгений Николаевич

(подпись инициалы, фамилия)

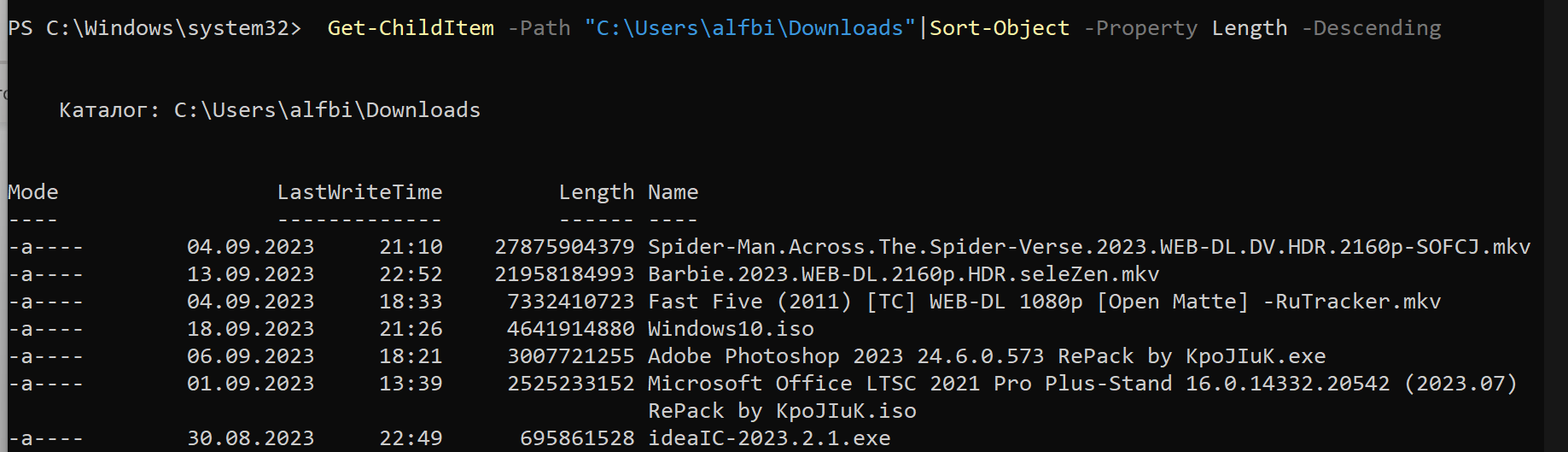
Екатеринбург 2023

1. Определение конвейера (запомнить). Привести пример применения конвейера.

Конвейер (или pipeline) в PowerShell представляет собой мощный механизм для передачи данных между командами. Он позволяет выполнять цепочку команд, где вывод одной команды становится входом для следующей команды. Конвейер в PowerShell обычно обозначается символом вертикальной черты |.

Пример применения конвейера:

Допустим, вы хотите вывести список всех файлов в определенной директории, отсортированных по размеру в убывающем порядке. Для этого вы можете использовать следующую команду:



В этой команде:

* Get-ChildItem используется для получения списка файлов и директорий в указанной директории.
* | передает вывод Get-ChildItem следующей команде через конвейер.
* Sort-Object используется для сортировки объектов по свойству "Length" (размер файла) в убывающем порядке ("-Descending").

Таким образом, вы получите список файлов в указанной директории, отсортированных по размеру в убывающем порядке.

1. Получение информации о активных процессах с помощью PowerShell:

Для получения начальной информации об активных процессах в PowerShell, вы можете использовать команду Get-Process. Эта команда предоставляет информацию о запущенных процессах, включая такие параметры, как Handles, NPM(K), WS(K), и VM(M), которые могут быть важными для мониторинга процессов.

Обозначения параметров:

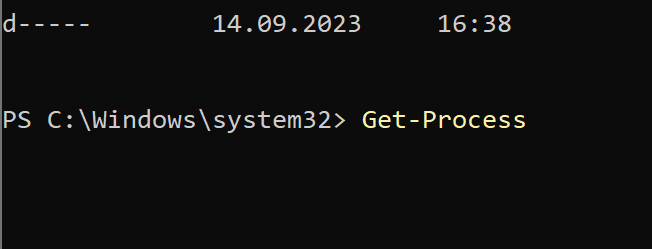
Handles: Количество дескрипторов, управляемых процессом.

NPM(K): Количество памяти в мегабайтах (МБ), используемое процессом.

WS(K): Количество рабочей памяти в мегабайтах (МБ), используемое процессом.

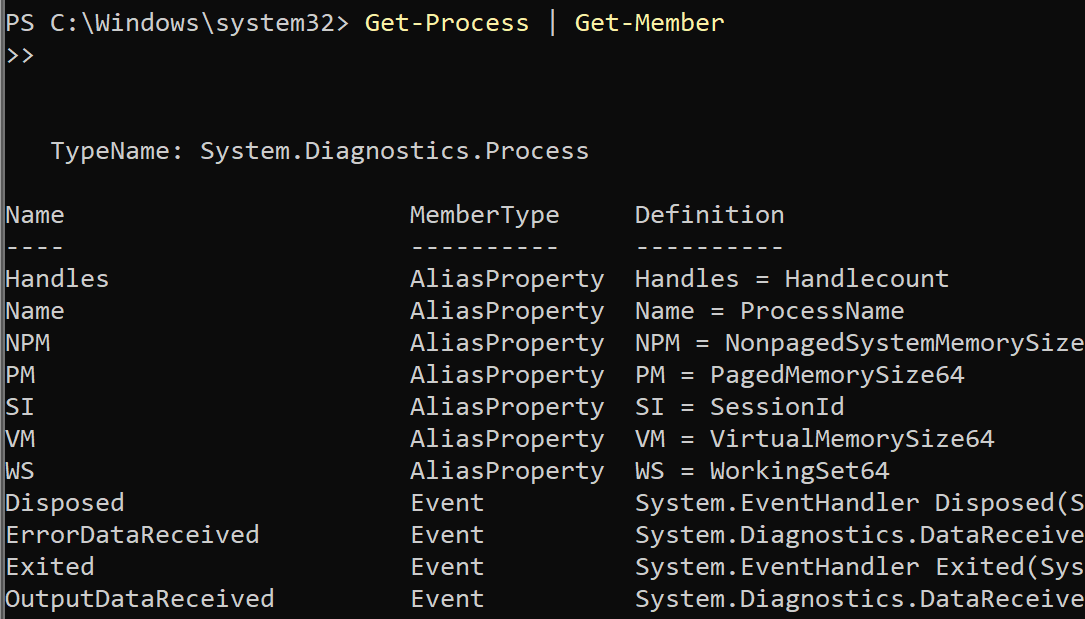
VM(M): Количество виртуальной памяти в мегабайтах (МБ), используемое процессом.

Чтобы получить информацию о всех активных процессах, выполните следующую команду:

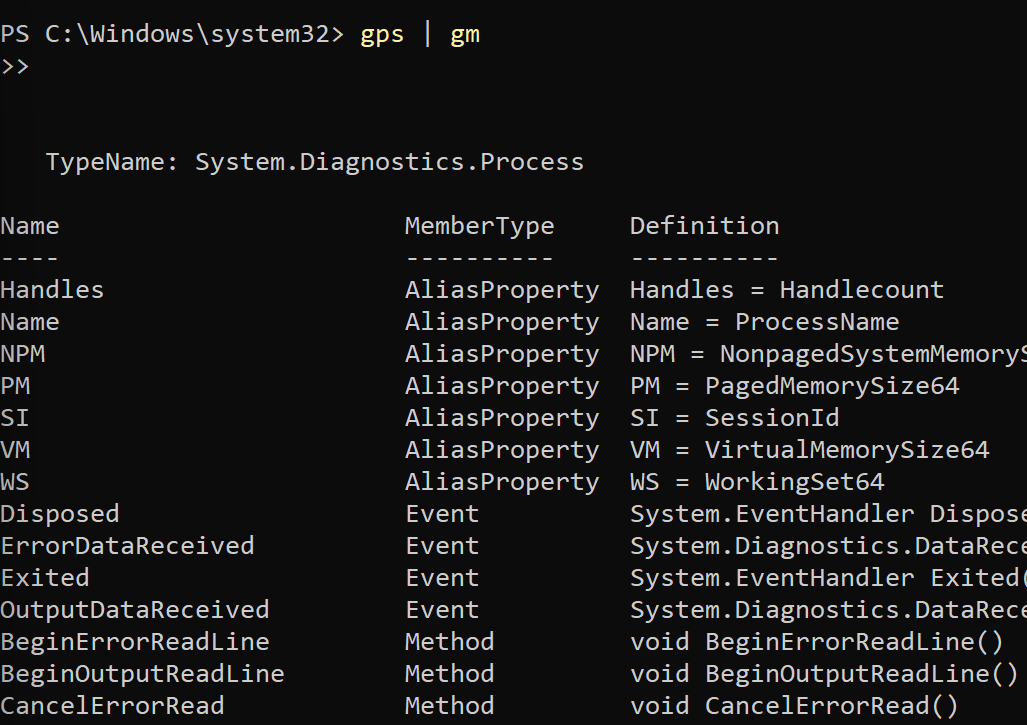


1. Пример использования конвейера из двух командлетов "Get-Process | Get-Member" в полном и сокращенном вариантах:

Полный вариант:



Сокращенный вариант:



В обоих вариантах команды "Get-Process" используется для получения списка процессов, а затем этот список передается командлету "Get-Member" через конвейер для анализа его членов (свойств и методов).

Расшифровка столбцов в выводе "Get-Member":

1. MemberType: Этот столбец указывает тип члена объекта, который анализируется. Это может быть одним из следующих значений:

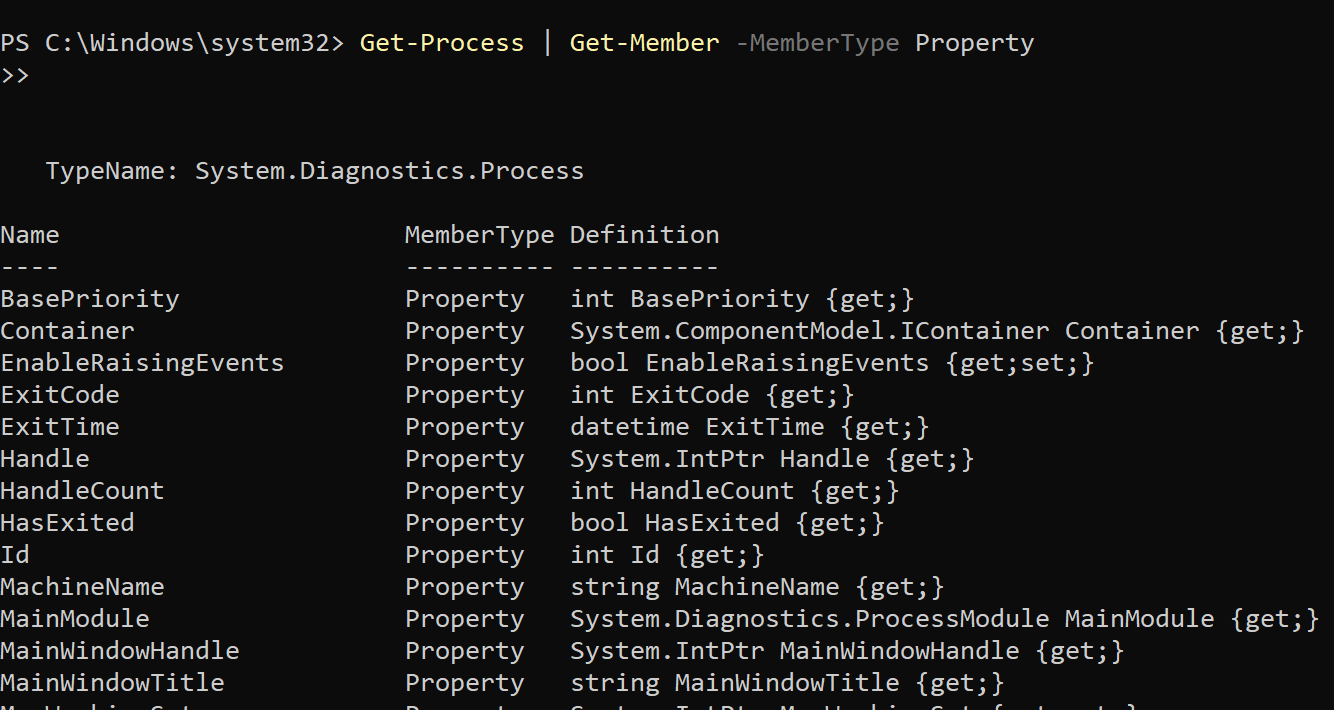
* AliasProperty: Псевдоним для свойства.
* CodeMethod: Метод, определенный как скрипт.
* Property: Свойство объекта.
* ScriptMethod: Метод объекта, определенный как скрипт.
* Method: Метод объекта.
* Event: Событие объекта.

1. Definition: Этот столбец предоставляет определение члена. Для свойств это может быть тип данных свойства, а для методов - код метода.

Таким образом, командлет "Get-Member" позволяет изучать структуру объектов, возвращаемых командлетом "Get-Process", и узнавать о доступных свойствах и методах для каждого из них. Это полезно при работе с объектами в PowerShell, так как вы можете узнать, какие операции можно выполнять с полученными данными.

4.Получить перечень свойств процессов для заданной категории свойств. Категория свойств процессов «-MemberType Property».

Для получения перечня свойств процессов, принадлежащих категории свойств "Property", вы можете использовать команду "Get-Process" с конвейером, чтобы отфильтровать только свойства, соответствующие указанной категории. Вот как это можно сделать:



Эта команда выполнит следующие шаги:

Get-Process - получает список процессов.

«|» - передает список процессов в следующую команду через конвейер.

Get-Member -MemberType Property - анализирует члены списка процессов и выводит только те, которые принадлежат категории "Property" (свойства процессов).

Результатом выполнения команды будет список свойств процессов, принадлежащих к указанной категории. Вы увидите названия свойств и их типы данных.

5. С помощью оболочки «PowerShell» найти несколько конфигурационных файлов, заканчивающихся на «\*format.pslxml». Выделить файл «DotNetTypes. format.pslxml

Для поиска конфигурационных файлов, заканчивающихся на "\*format.pslxml" и выделения файла "DotNetTypes.format.pslxml" с помощью оболочки PowerShell, вы можете использовать команду Get-ChildItem с использованием маски в имени файла и затем фильтровать результаты. Вот как это можно сделать:

# Найти все файлы, заканчивающиеся на \*format.pslxml

$files = Get-ChildItem -Path C:\путь\_к\_директории -Filter "\*format.pslxml"

# Вывести список найденных файлов

$files

# Выделить файл "DotNetTypes.format.pslxml"

$dotNetTypesFile = $files | Where-Object { $\_.Name -eq "DotNetTypes.format.pslxml" }

# Вывести найденный файл "DotNetTypes.format.pslxml"

$dotNetTypesFile

6. Пояснить «Методы дихотомии». Определение?

Примеры демонстрируют, что объем выводимых данных может оказаться огромным, и для удобства необходимы инструменты, способные отсеивать лишнюю информацию и выделять нужную. Обычно для этой задачи применяют методы сортировки и фильтрации. Создание рабочих массивов данных, упорядоченных по определенному параметру, позволяет использовать методы дихотомии в соответствии с зависимостью

р = log2 N,

где р – количество шагов, необходимых для нахождения данных в отсортированном массиве; N – число элементов в изначальном анализируемом массиве.

Дихотомия означает разделение на две части, последовательное разбиение на две непересекающиеся части. Этот метод логического разделения класса на подклассы заключается в том, что исходное понятие разделяется на два взаимодополняющих понятия. Дихотомическое разделение в математике используется для формирования подразделов одного понятия или термина и служит для классификации элементов. Существует теорема: "Если непрерывная функция имеет разные знаки на концах некоторого интервала, то внутри этого интервала у нее есть корень (как минимум один, но может быть и несколько)". Эта теорема является основой численного нахождения приближенного значения корня функции. В общем смысле, этот метод называется дихотомией, что означает деление интервала на две части.

7. Вывести список процессов, упорядоченных по процессорному

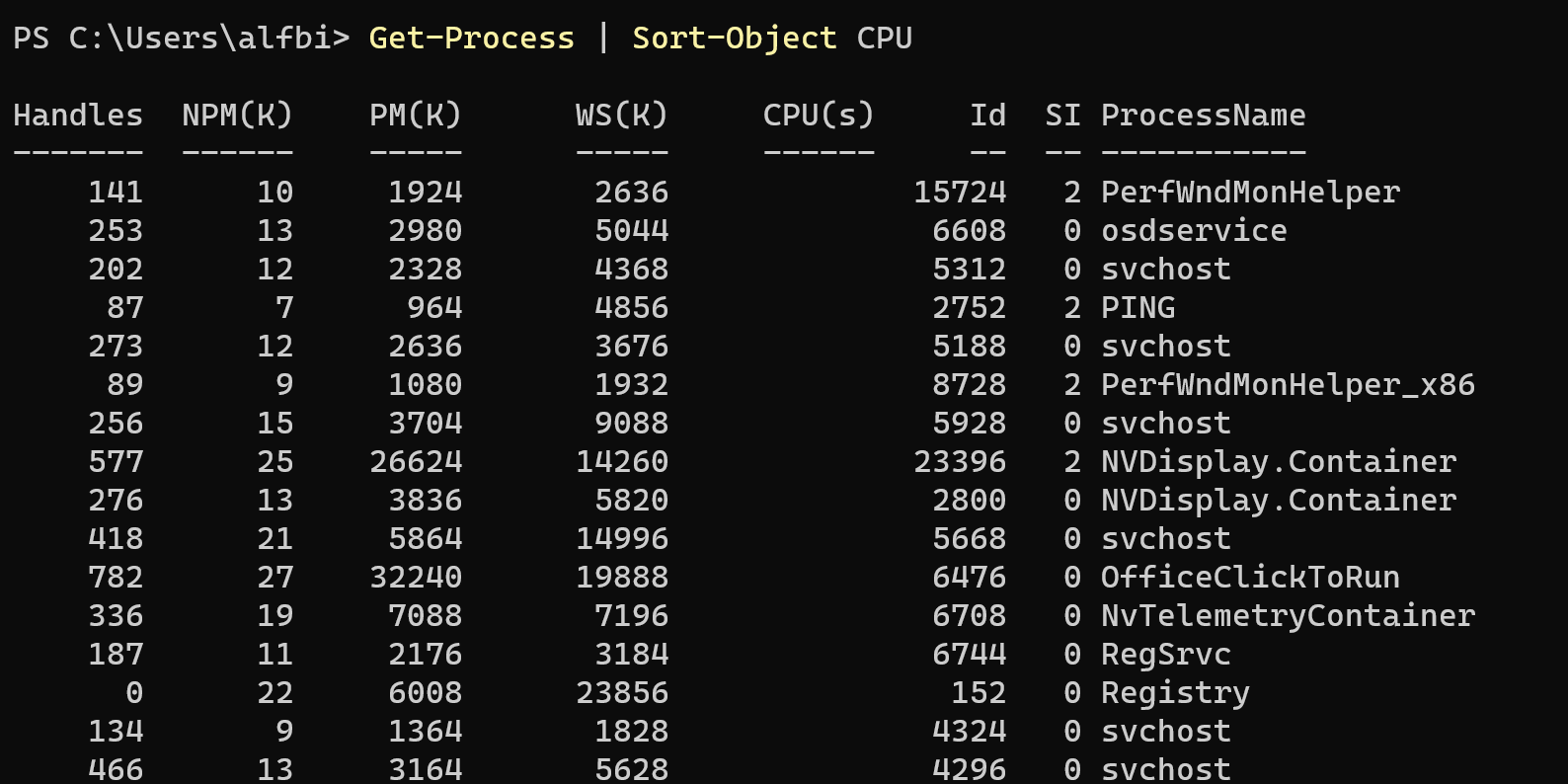
времени. Вывести список процессов, упорядоченных в порядке

убывания процессорного времени.

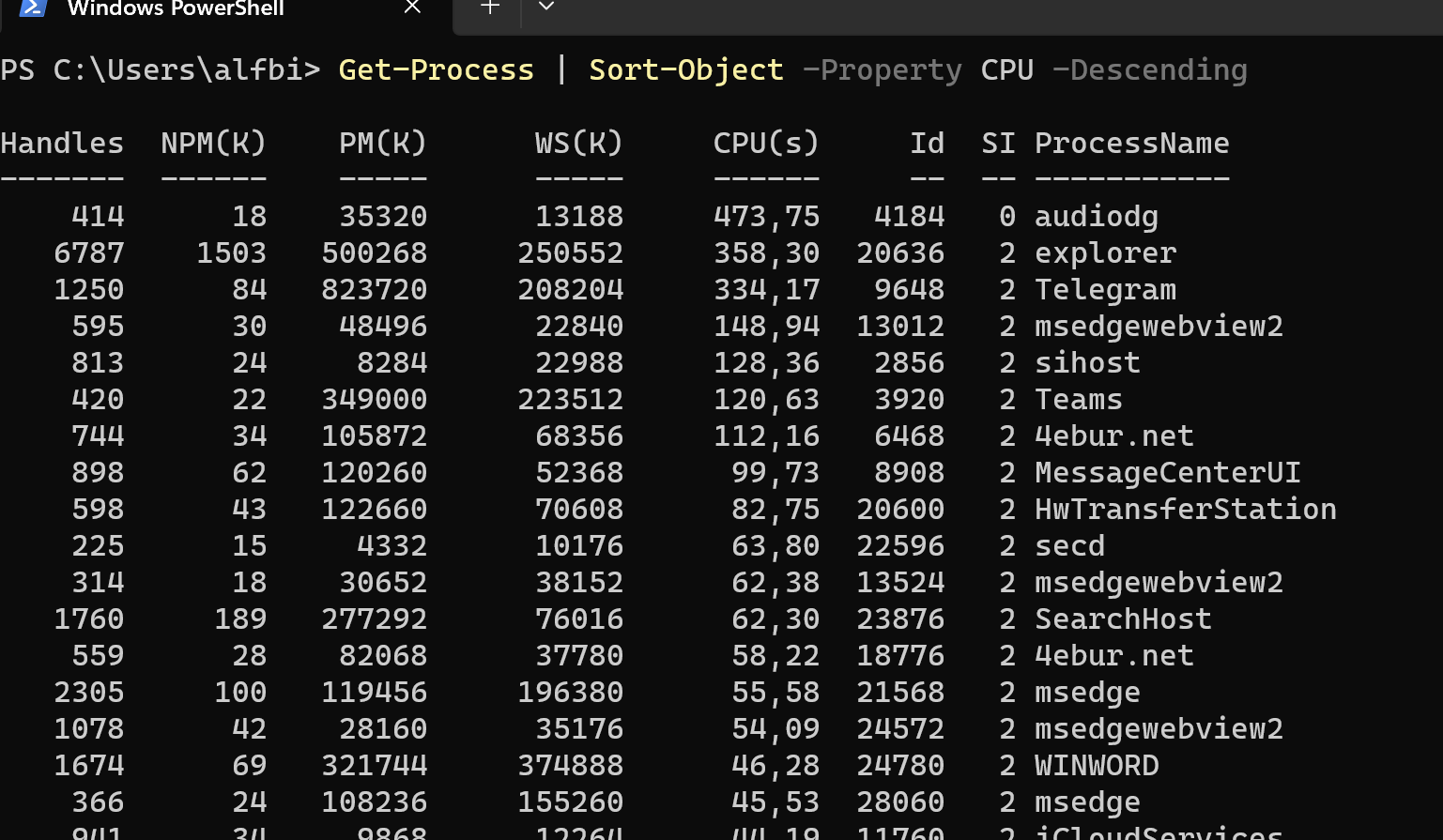
Для выполнения этой задачи в PowerShell, вы можете использовать команду Get-Process, а также метод сортировки Sort-Object. Вот как можно выполнить ваше задание:

Вывести список процессов, упорядоченных по процессорному времени (по возрастанию):

Powershell



Вывести список процессов, упорядоченных в порядке убывания процессорного времени:

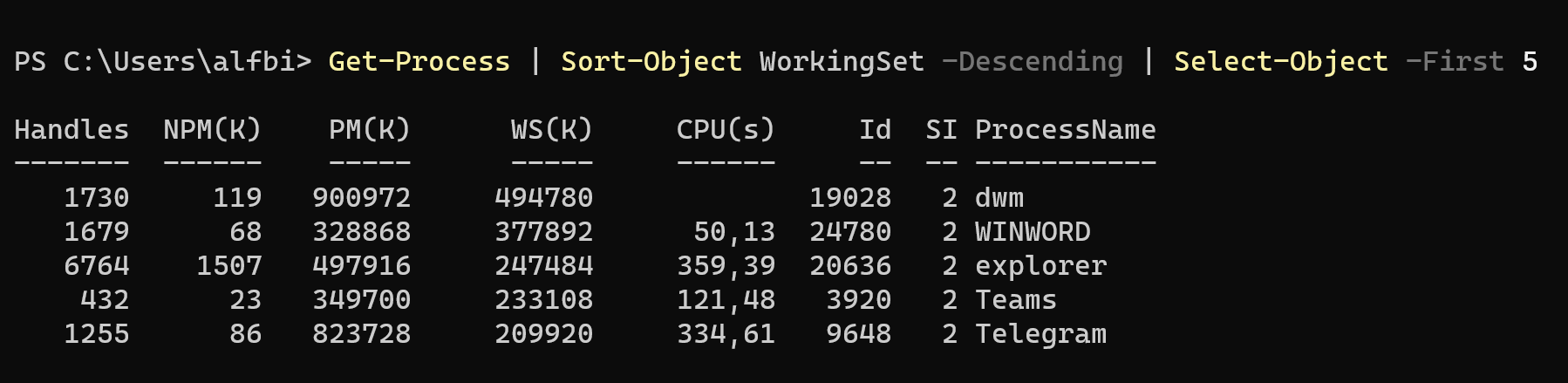


Первая команда Get-Process получает список всех запущенных процессов, а затем сортирует их по процессорному времени в порядке возрастания с помощью Sort-Object CPU. Вторая команда также использует Get-Process, но сортирует процессы в порядке убывания процессорного времени с помощью параметра -Descending.

8. Сформировать команду-конвейер для получения пяти процессов,

использующих наибольшие объемы памяти.

Для получения пяти процессов, использующих наибольшие объемы памяти в командной оболочке PowerShell, вы можете использовать команду Get-Process, сортировать процессы по используемой памяти и затем выбрать первые пять результатов. Вот как это можно сделать:



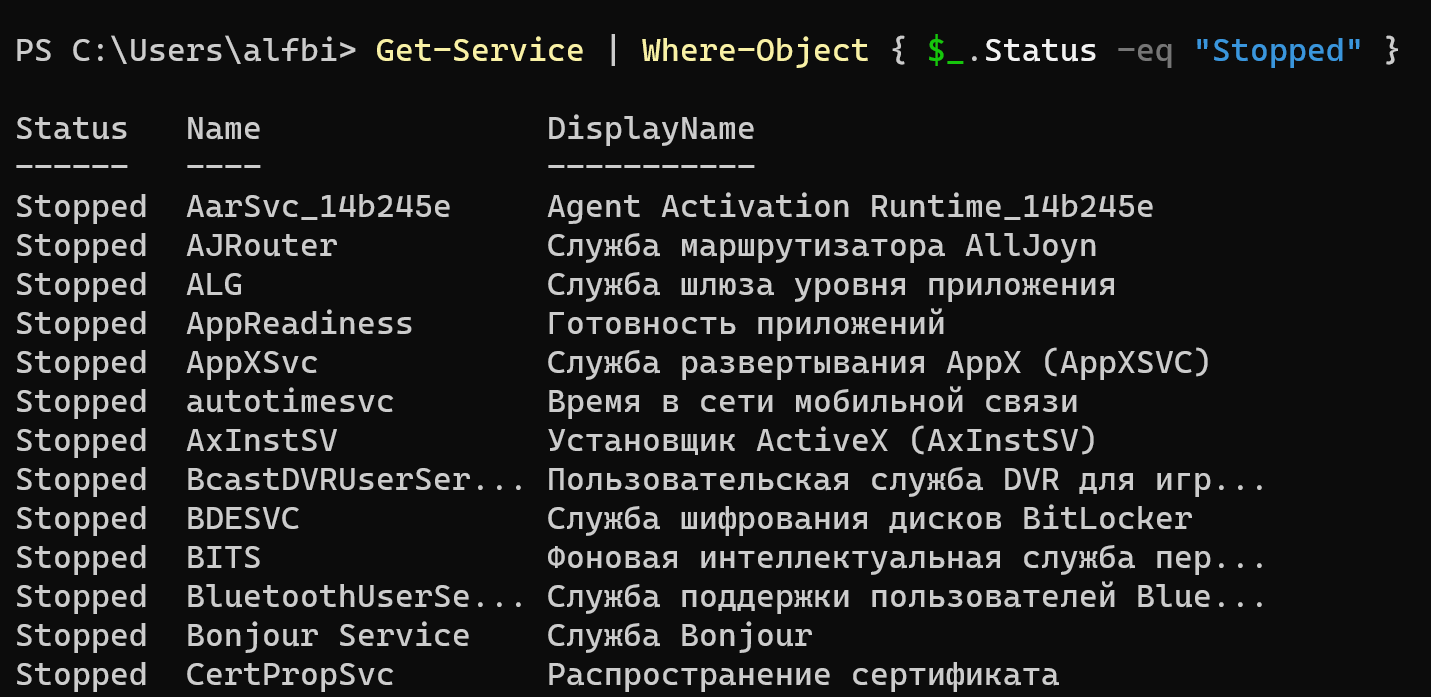
Эта команда выполняет следующие шаги:

1. Get-Process получает список всех запущенных процессов.
2. Sort-Object WorkingSet -Descending сортирует процессы по используемой памяти (WorkingSet) в порядке убывания (-Descending).
3. Select-Object -First 5 выбирает первые пять процессов из отсортированного списка, которые используют наибольшие объемы памяти.

Таким образом, вы получите список пяти процессов с наибольшими объемами используемой памяти.

9. Отфильтровать данные по остановленным службам, используя

команду-конвейер из двух командлет (свойство «Status» равно «Stopped»).Для фильтрации данных по остановленным службам в PowerShell, вы можете использовать команду Get-Service, а затем применить фильтр с помощью командлета Where-Object. Вот как это можно сделать:



Эта команда выполняет следующие шаги:

1. Get-Service получает список всех служб на компьютере.
2. Where-Object { $\_.Status -eq "Stopped" } фильтрует службы, оставляя только те, у которых свойство Status равно "Stopped" (остановленные службы).

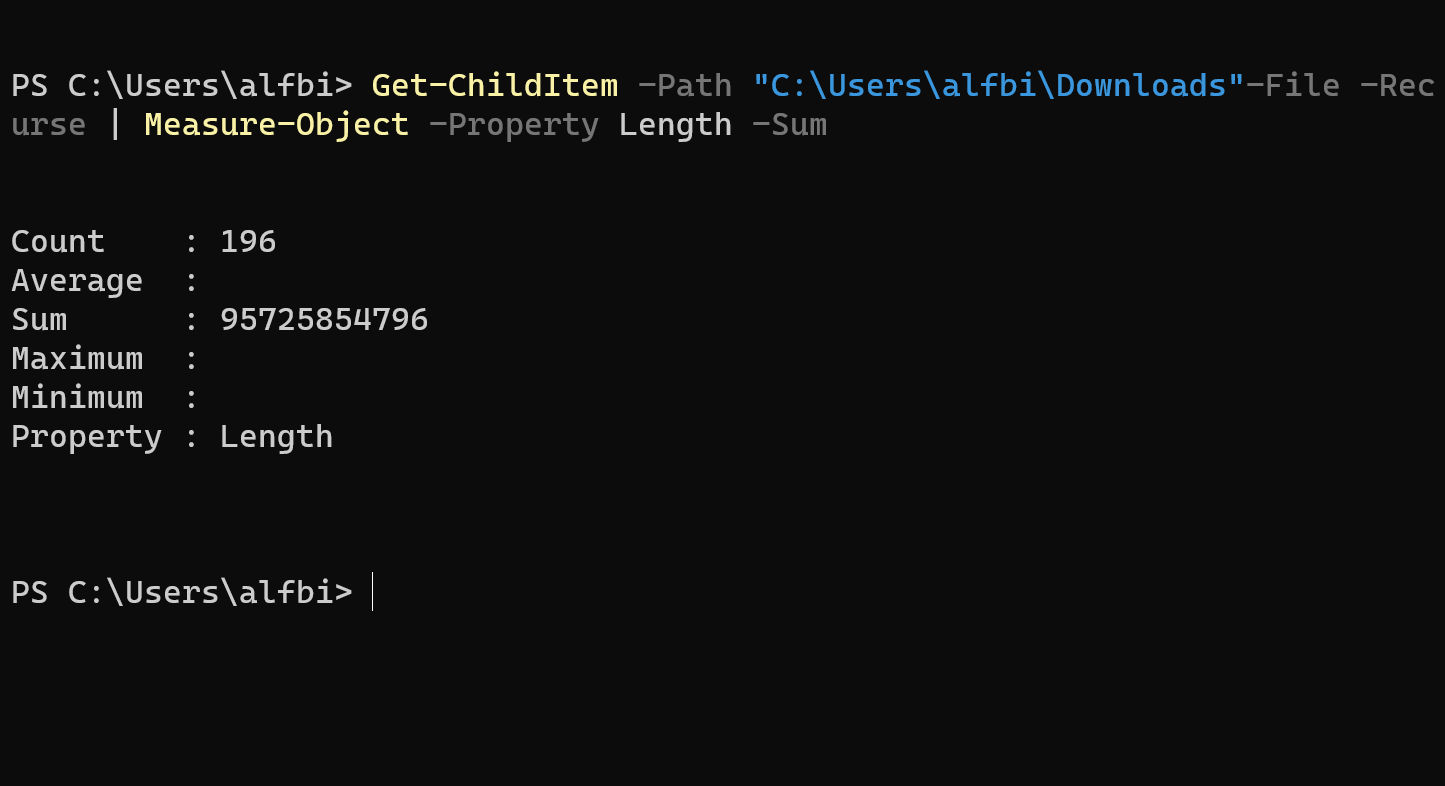
Таким образом, вы получите список служб, которые находятся в состоянии "Stopped" (остановлены).

10. Подсчитать объем памяти, занимаемый каталогом (каталог по

выбору). Решение задачи – обеспечить последовательностью команд

(см. рис. 1.35). Исправить некорректную запись в команде-конвейер.

Для подсчета объема памяти, занимаемого выбранным каталогом в PowerShell, вы можете использовать команду Get-ChildItem для получения файлов и подкаталогов внутри каталога, а затем использовать команду Measure-Object для подсчета суммарного размера файлов. Вот как это можно сделать:



Эта команда выполняет следующие действия:

1. Get-ChildItem получает список файлов и подкаталогов внутри выбранного каталога (-Path "C:\Путь\К\_Вашему\Каталогу").
2. -File фильтрует только файлы, исключая подкаталоги.
3. -Recurse позволяет просматривать подкаталоги рекурсивно.
4. Measure-Object -Property Length -Sum подсчитывает суммарный размер файлов в каталоге, используя свойство Length (размер файла).

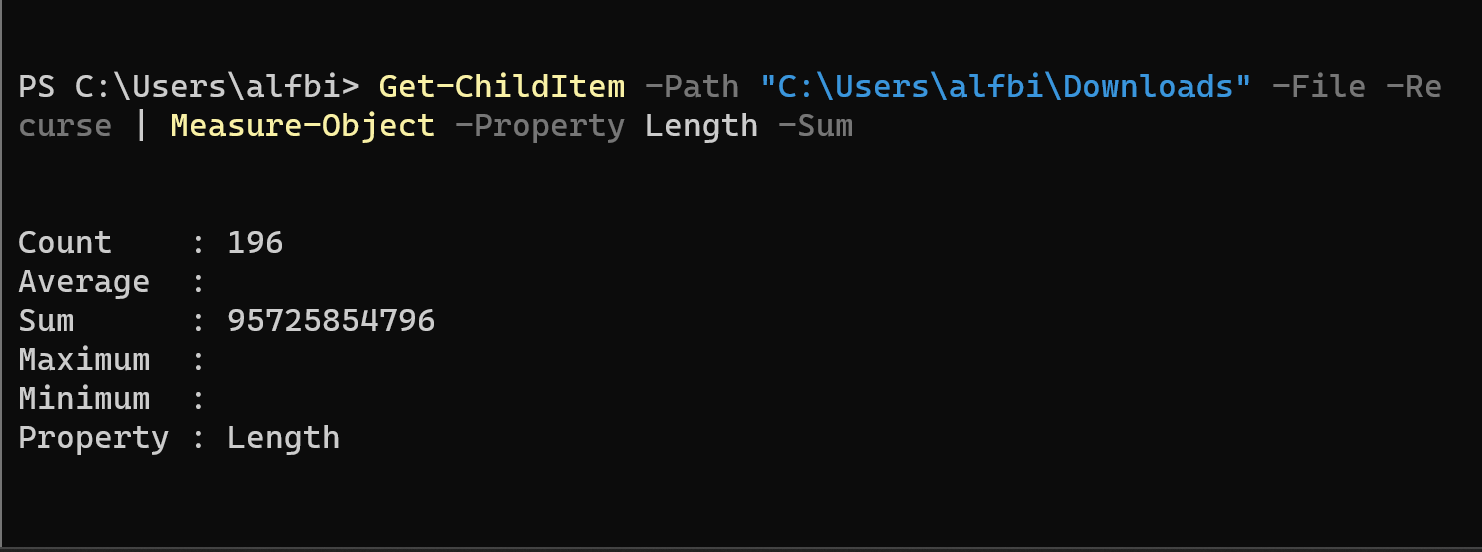
11. Подсчитать объем памяти, занимаемый каталогом (каталог по

выбору. Решение задачи обеспечить командлетом «Measure-Object» в

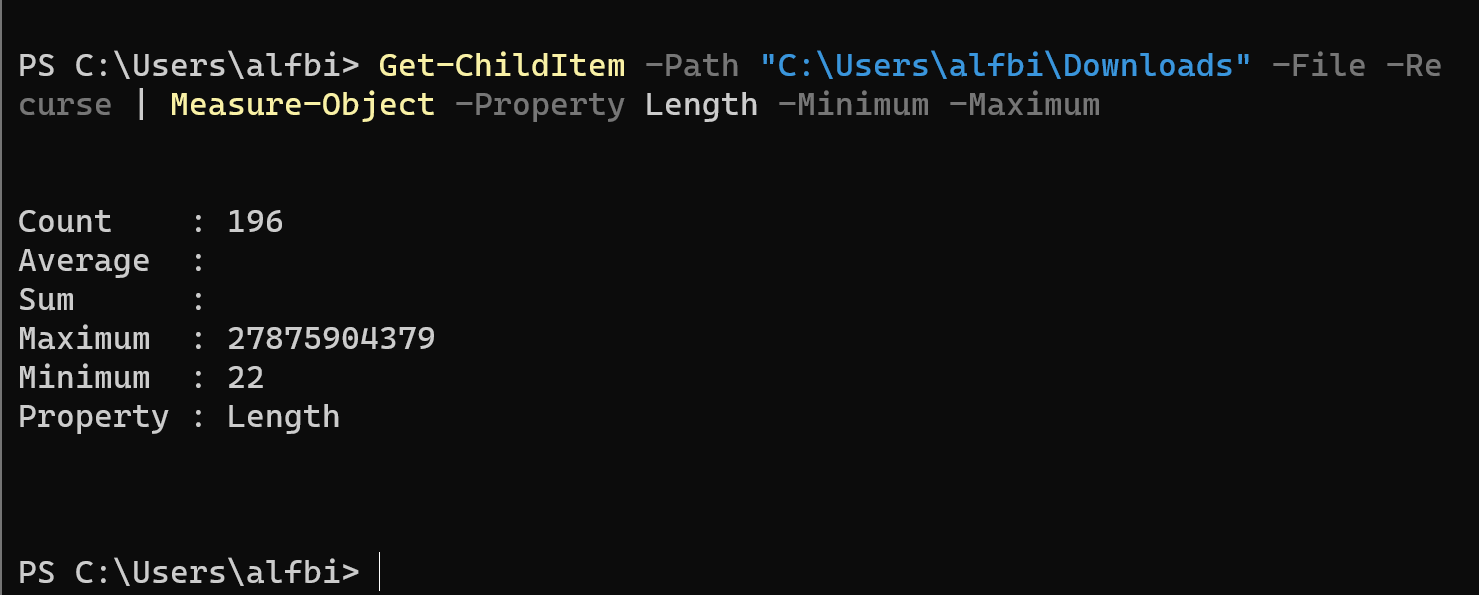
двух вариантах:

Для подсчета объема памяти, занимаемого каталогом в PowerShell с использованием командлета Measure-Object, в двух вариантах, вы можете использовать следующие команды:

Вариант 1: Расчет памяти, занимаемой каталогом:



Вариант 2: Расчет памяти, занимаемой каталогом с минимумом и минимумом



Этот код также использует командлет Measure-Object, но помимо суммарного размера файлов, он также выводит минимальный и максимальный размер файла в выбранном каталоге.