ГУАП

КАФЕДРА № 42

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент, канд. техн. наук |  |  |  | А.В. Аграновский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5  Разработка командлетов PowerShell |
| по курсу: Инструментальные средства информационных систем |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |
| --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4321 |  | Г.В. Буренков |
|  |  | подпись, дата | инициалы, фамилия |

Санкт-Петербург 2025

**СОДЕРЖАНИЕ**

[1 Цель работы 2](#_Toc197621190)

[2 Выполненные упражнения со скриншотами 3](#_Toc197621191)

[3 Справочная информация о командах из индивидуального задания 11](#_Toc197621192)

[4 Выводы о проделанной работе 16](#_Toc197621193)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17](#_Toc197621194)

**1 Цель работы**

Целью данной лабораторной работы является изучение командлетов PowerShell, усовершенствование навыков администрирования в ОС Windows, а также выполнение индивидуального варианта. На рисунке 1 иозображены варианты для выполнения данной работы.

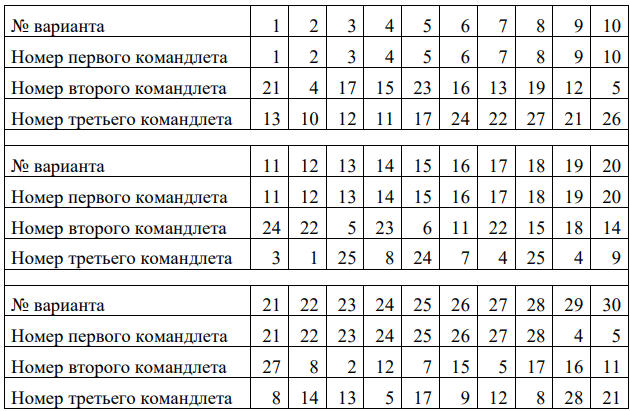


Рисунок 1 – Табличка вариантов

**2 Выполненные упражнения со скриншотами**

В процессе выполнения упражнений 2.5 - 2.11 я разобрался различных командлетах PowerShell. В первую очередь я разобрался с командлентами Get-Service, Sort-Object, Get-Process. На рисунках 2-17 изображены различные результаты командлет.

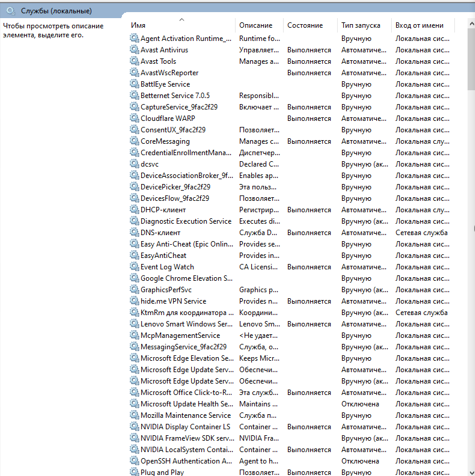


Рисунок 2 – Список служб

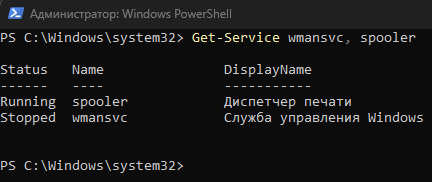


Рисунок 3 – Изменение шрифта в cmd.exe

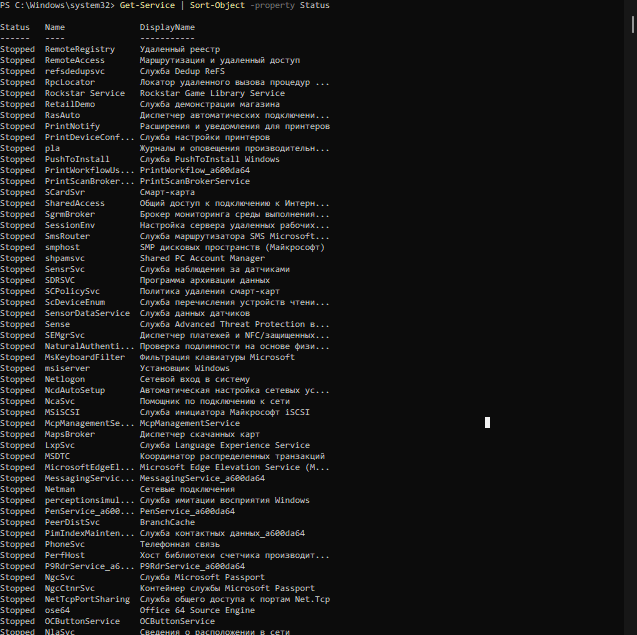


Рисунок 4 – Результат сортировки служб

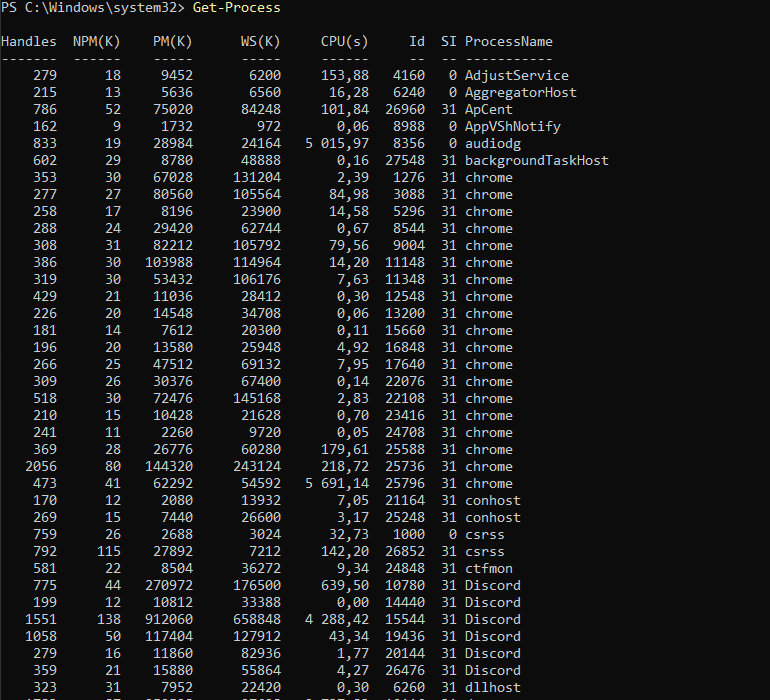


Рисунок 5 – Результат выполнения команды Get-Process

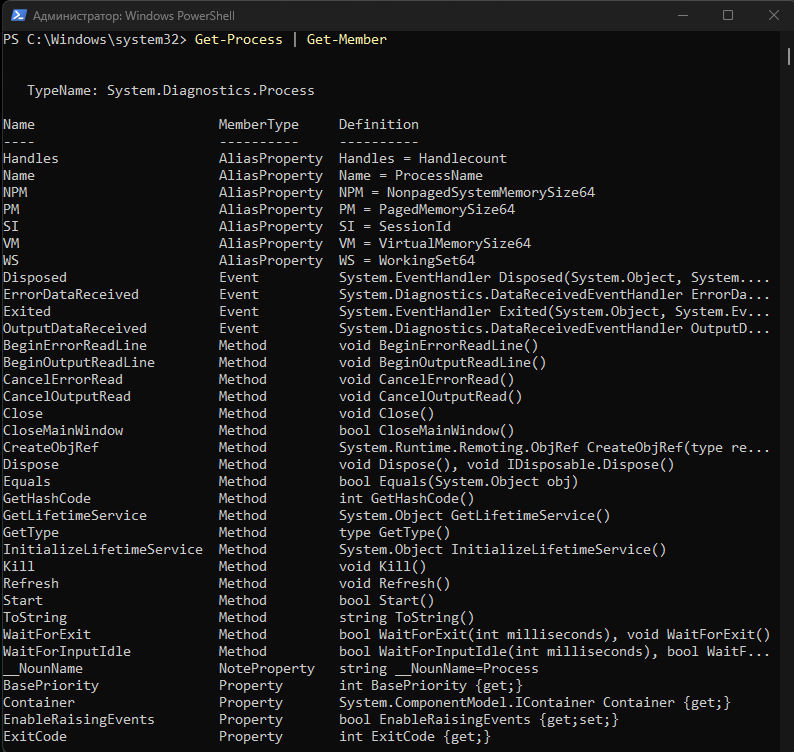


Рисунок 6 – Результат выполнении Get-Process в конвейере

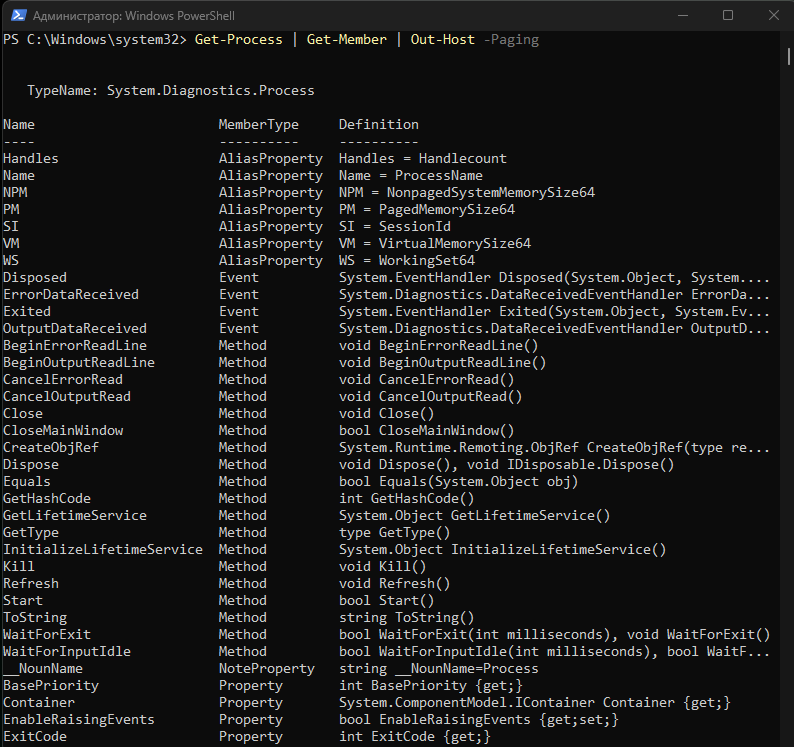


Рисунок 7 – Вывод процессов

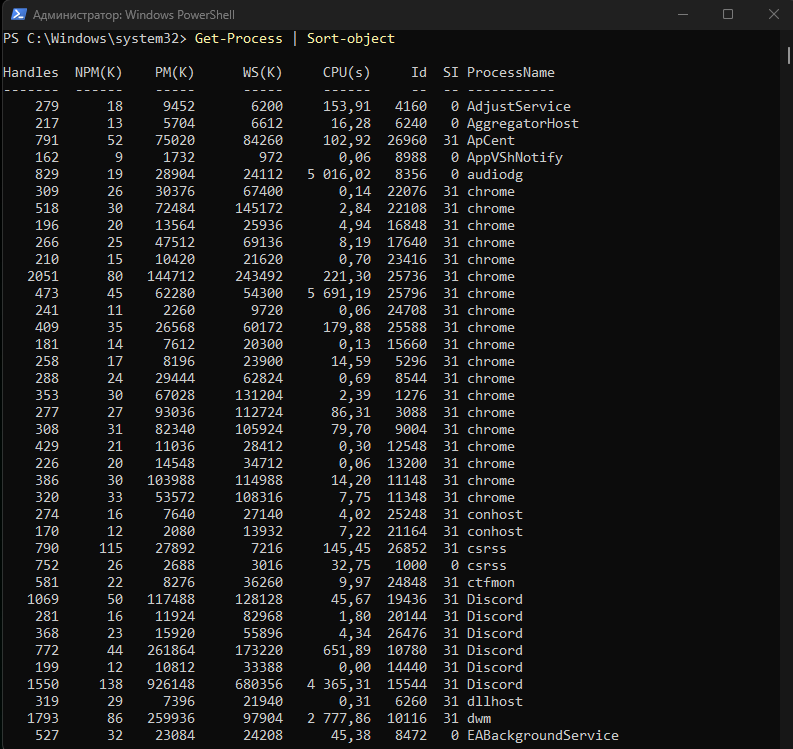


Рисунок 8 – Сортировка по времени использования ЦП

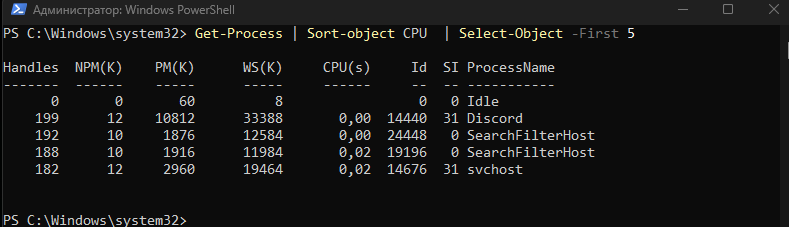


Рисунок 9 – Вывод первых 5 отсортированных процессов ЦП

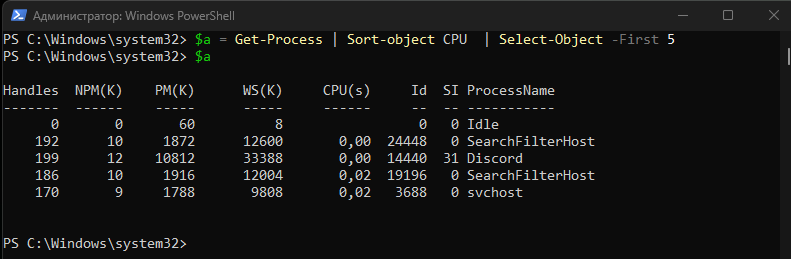


Рисунок 10 – Ввод и вывод переменной

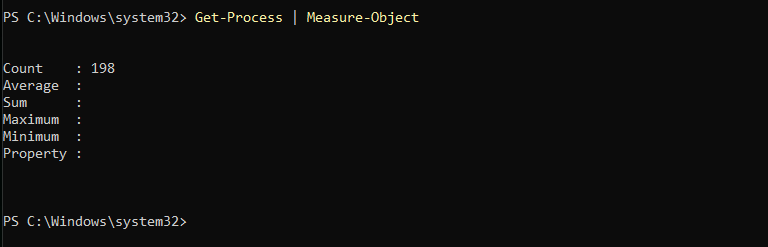


Рисунок 11 – Подсчет количества процессов

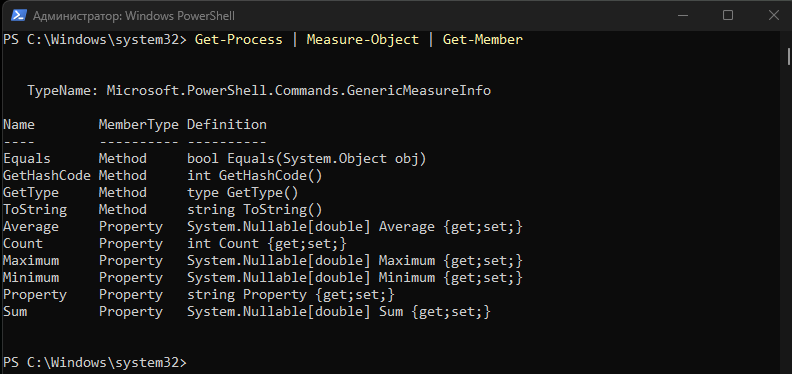


Рисунок 12 – Вывод Children объекта Measure-Object

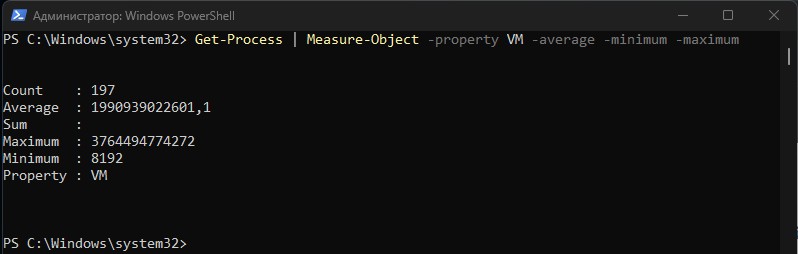


Рисунок 13 – Выполнение объекта с ключами

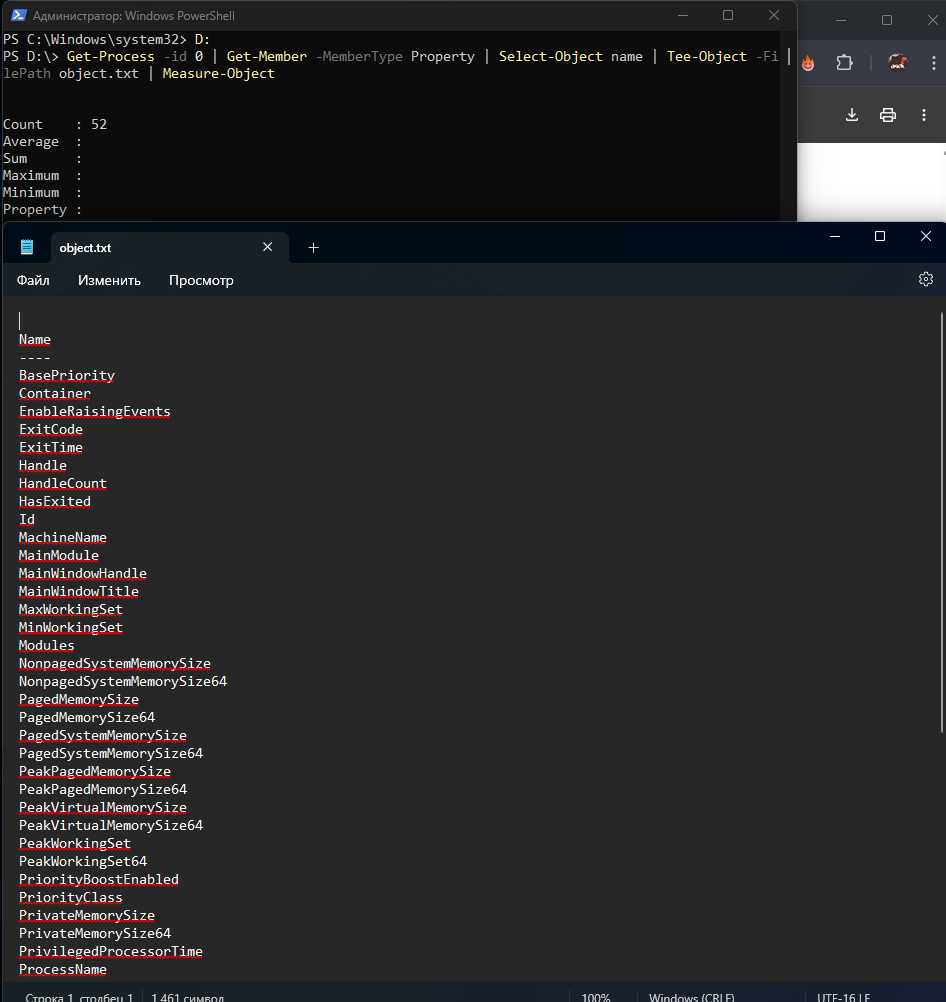


Рисунок 14 – Подсчет объектов и вывод результата в файл

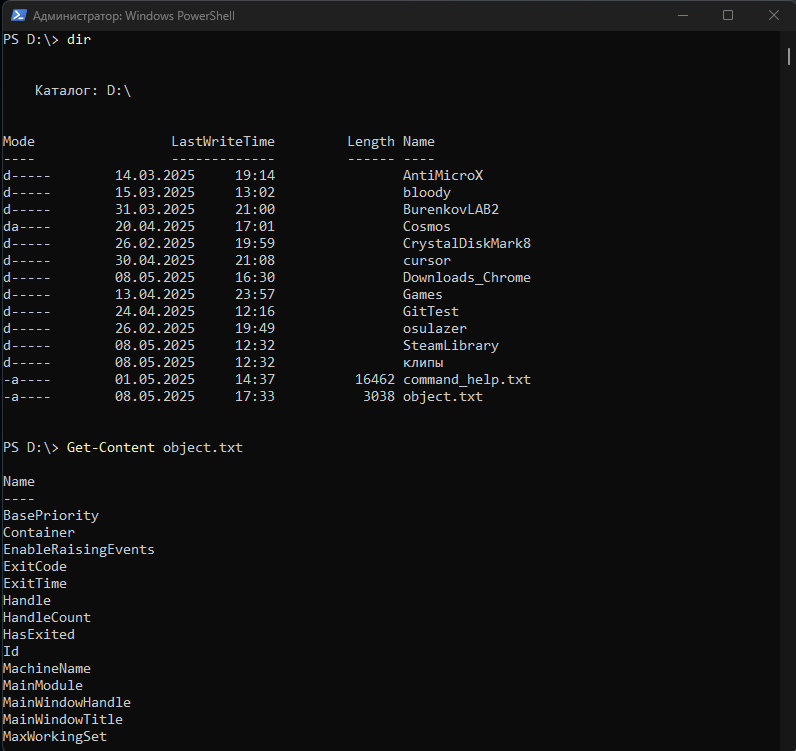


Рисунок 15 – Вывод файла

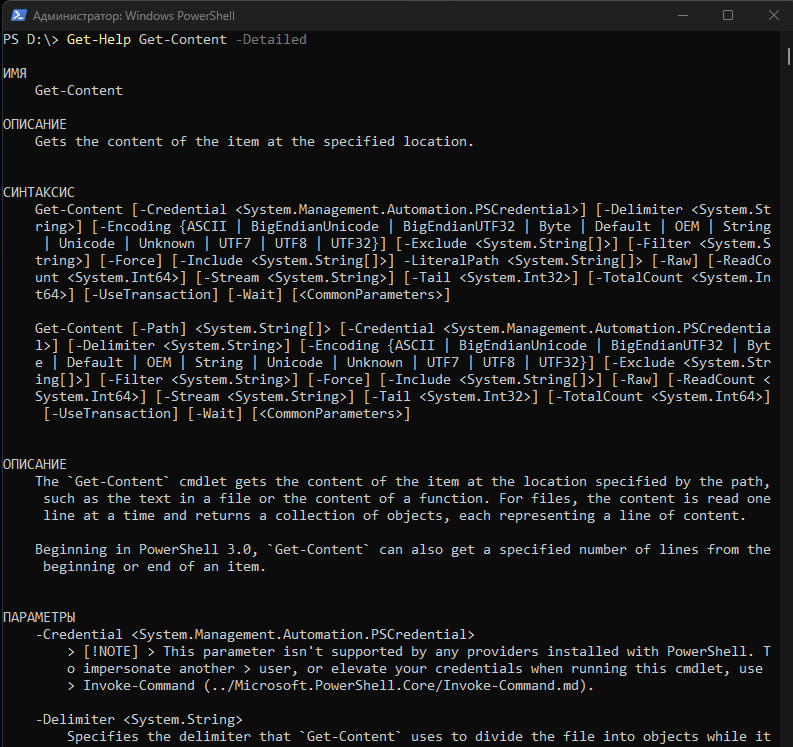


Рисунок 16 – Вывод справки команды Get-Content

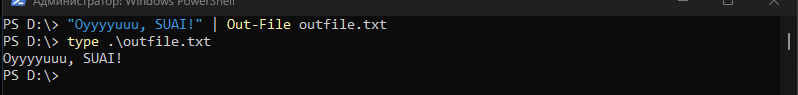


Рисунок 17 – Вывод справки команды Get-Content

**3 Справочная информация о командах из индивидуального задания**

В соответствии с вариантом 1 мне необходимо рассмотреть команды Clear-Variable, Start-Process, Group-Object. Первая команда из индивидуального задания — Clear-Variable. Это командлет PowerShell, предназначенный для удаления значений, присвоенных переменным в текущем сеансе. Он не удаляет саму переменную, а лишь очищает ее содержимое, то есть делает переменную «пустой», без значения. Это удобно, когда нужно сбросить значение переменной, не удаляя ее из области видимости скрипта или консоли. Команда может применяться как к одной переменной, так и к множеству переменных одновременно. Clear-Variable полезен при написании скриптов, в которых требуется обнуление данных перед повторным использованием переменных, что помогает избежать ошибок, связанных с сохранёнными ранее значениями.

Одним из ключей является -Name, который указывает имя переменной, значение которой нужно очистить. Например, Clear-Variable -Name myVar очистит содержимое переменной myVar. Второй важный ключ — -Scope, определяющий область действия команды (например, текущая функция, скрипт или глобальная область). Это позволяет точно контролировать, где именно нужно произвести очистку значения. Третий ключ — -Force, он используется для очистки защищённых переменных, которые в обычных условиях могут быть недоступны для изменений. Использование этого ключа требует повышенных прав и должно применяться с осторожностью, чтобы не нарушить выполнение важных процессов скрипта. На рисунках 18, 19 изображена справка по командлету Clear-Variable.

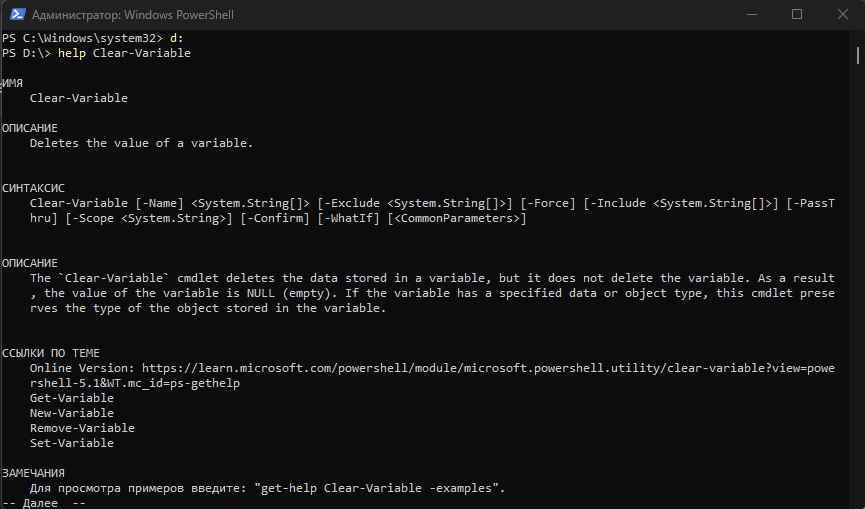


Рисунок 18 – справка по Clear-Variable

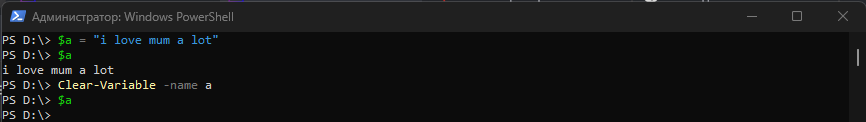


Рисунок 19 – пример использования Clear-Variable

Вторая команда из индивидуального задания — Start-Process. Этот командлет используется для запуска новых процессов в операционной системе Windows. Он предоставляет гибкие возможности по запуску внешних программ, сценариев, приложений или команд с различными параметрами. Командлет Start-Process особенно полезен в автоматизированных скриптах и административных задачах, когда необходимо инициировать выполнение программ с определёнными условиями, например, в фоновом режиме, с повышенными правами или в указанной рабочей директории.

Командлет может быть использован для запуска как простых программ, вроде блокнота (Start-Process notepad), так и сложных процессов с аргументами и настройками. Это делает его мощным инструментом для администраторов и разработчиков. Кроме того, он позволяет задать, следует ли дожидаться завершения процесса, что особенно важно при последовательной автоматизации действий.

Ключ -FilePath является обязательным и указывает путь к исполняемому файлу или приложению, которое нужно запустить. Например, Start-Process -FilePath "notepad.exe" инициирует запуск блокнота. Второй важный ключ — -ArgumentList, который позволяет передавать аргументы запускаемой программе. Это полезно при работе с программами, поддерживающими параметры командной строки, например: Start-Process -FilePath "cmd.exe" -ArgumentList "/c echo Hello". Третий ключ — -Wait — указывает PowerShell ждать завершения процесса перед выполнением следующих команд. Это особенно важно при написании скриптов, где действия должны выполняться последовательно. На рисунке 20-21 изображена справка по командлету Start-Process.

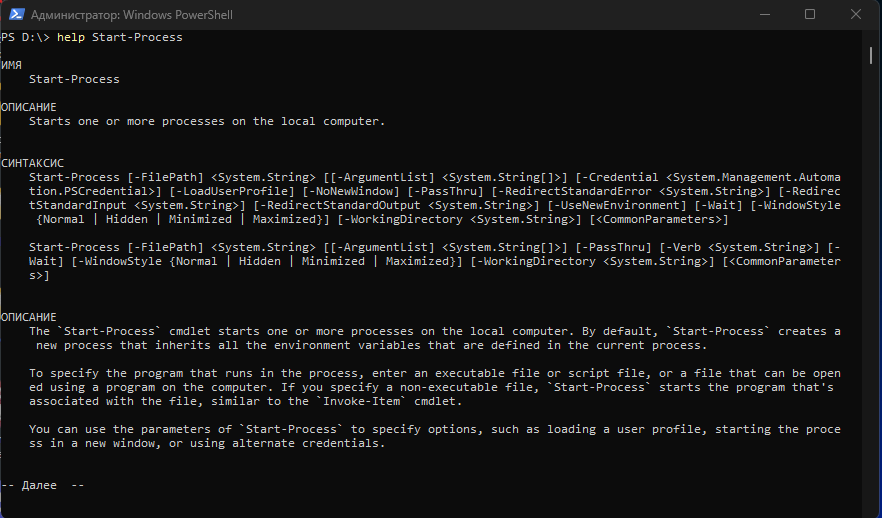


Рисунок 20 – справка по Start-Process

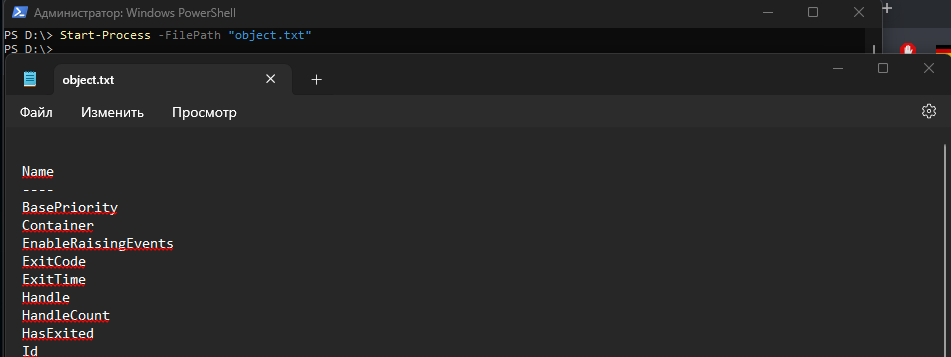


Рисунок 21 – вывод файла со свойством FilePath

Третья команда из индивидуального задания — Group-Object. Этот командлет PowerShell используется для группировки объектов по значению одного или нескольких их свойств. Он часто применяется при анализе больших массивов данных, полученных из различных источников, таких как списки процессов, файлы, службы и другие системные объекты. Командлет создает на выходе сгруппированный результат, где для каждого уникального значения свойства указывается количество соответствующих объектов и сами объекты, попавшие в эту группу. Это позволяет быстро выявлять повторяющиеся элементы и проводить статистический анализ.

Например, если получить список процессов и сгруппировать их по имени, можно легко узнать, сколько экземпляров каждой программы запущено. Или при анализе данных из файла можно сгруппировать записи по какому-либо полю, чтобы увидеть структуру распределения значений.

Один из ключевых параметров — -Property, который определяет имя свойства, по которому производится группировка. Это основной ключ, без которого командлет не будет работать осмысленно. Пример использования: Get-Process | Group-Object -Property ProcessName. Второй ключ — -AsString, который указывает, что свойство следует рассматривать как строку. Это полезно, если значения по умолчанию интерпретируются как объекты, а не как текст. Третий ключ — -NoElement, при использовании которого в результатах группировки не отображаются сами объекты, только имя группы и количество элементов. Это удобно, если интересует только статистика, без лишней детализации. На рисунке 22-23 изображена справка по командлету Group-Object.

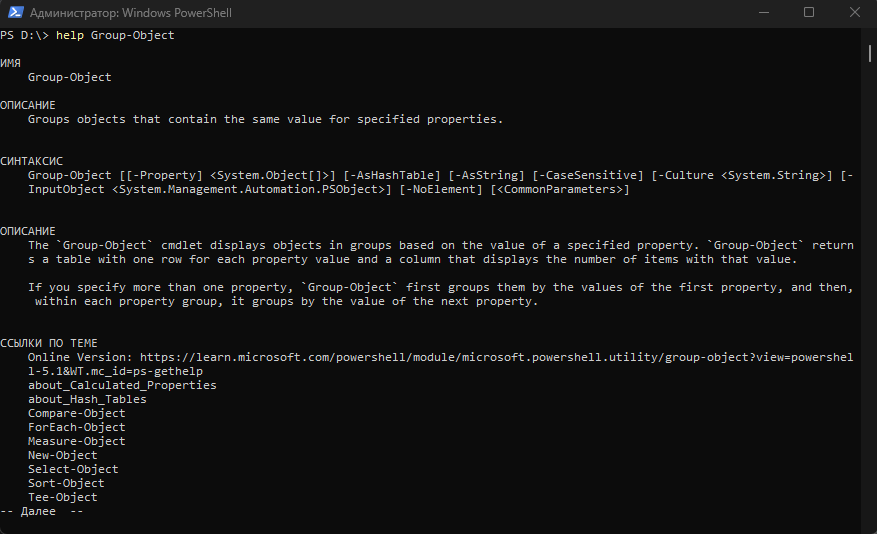


Рисунок 22 – справка Group-Object

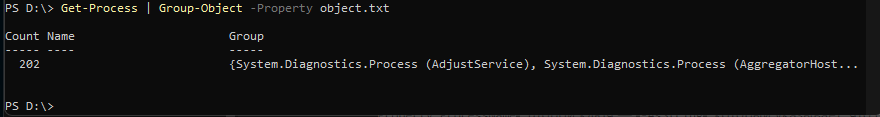
****

Рисунок 23 – вывод с помощью Group-Object

**4 Выводы о проделанной работе**

В ходе выполнения лабораторной работы была достигнута основная цель — изучение и практическое применение командлетов PowerShell. На этапе выполнения упражнений были исследованы базовые команды оболочки PowerShell, такие как Get-Service, Sort-Object, Get-Process, Measure-Object, Select-Object, Get-Content и другие. Они продемонстрировали принципы работы с объектами, конвейерной обработки данных, сортировки, фильтрации, подсчета объектов и вывода информации в файлы. Были получены практические навыки управления службами, анализа запущенных процессов, работы с переменными и текстовыми файлами. Эти упражнения позволили освоить синтаксис командлетов, научиться комбинировать их с помощью конвейеров и использовать параметры для получения нужных результатов.

В рамках индивидуального задания были подробно изучены три команды: Clear-Variable, Start-Process и Group-Object. Каждая из них играет важную роль в автоматизации административных задач. Clear-Variable позволяет очищать значения переменных без их удаления, что необходимо при повторном использовании переменных в скриптах. Start-Process обеспечивает запуск внешних процессов с передачей аргументов и управлением их выполнением, что полезно при написании универсальных скриптов. Group-Object служит для группировки объектов по значению их свойств, позволяя эффективно анализировать и обрабатывать данные. Для каждого командлета были рассмотрены ключевые параметры и приведены пояснения к их применению.

Полученные знания позволяют уверенно ориентироваться в среде PowerShell, создавать скрипты для автоматизации типовых задач и эффективно управлять объектами операционной системы. Это создаёт прочную основу для дальнейшего изучения инструментальных средств информационных систем и их применения в профессиональной деятельности.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. ГУАП, документация для учебного процесса. – URL: <https://guap.ru/regdocs/docs/uch> (дата обращения 22.04.2025)
2. Полезные команды в cmd на Windows – URL: <https://htmlacademy.ru/blog/soft/windows-command-line> (дата обращения 26.04.2025)
3. CMD Полная документация для Windows – URL: <https://learn.microsoft.com/ru-ru/windows-server/administration/windows-commands/cmd> (дата обращения 26.04.2025)