

ПАРКТИЧНЕ ЗАНЯТТЯ 2

СТАНДАРТНІ УТИЛІТИ WINDOWS ТА РОБОТА З WINDOWS POWERSHELL

2.1 Мета заняття

Познайомитися з видами деяких стандартних утиліт Windows та основними прийомами роботи з ними за допомогою Windows PowerShell.

2.2 Теоретичні положення

Windows PowerShell – командна оболонка Windows, яка розроблена для адміністрування і конфігурації ОС Windows, має консольний інтерфейс та використовує скриптову мову.

PowerShell розроблена на основі середовища CLR і платформи .NET Framework і на відміну від командного рядка, який приймає і повертає текст, Windows PowerShell працює з об'єктами. У кожного об'єкта в PowerShell є властивості і методи, які можна використовувати для управління цими об'єктами.

Для PowerShell була розроблена концепція командлетів (cmdlets), тобто система іменування команд «Дієслово-Іменник».

PowerShell надає наступні можливості:

- отримувати доступ до файлової системи;
- управляти реєстром;
- керувати службами;
- керувати процесами;
- налаштовувати операційну систему;
- встановлювати програмне забезпечення;
- встановлювати ролі і компоненти сервера;
- здійснювати адміністрування і конфігурування ролей та компонентів сервера;
- писати і використовувати сценарії для автоматизації управління та адміністрування;
- виконувати інші завдання системних адміністраторів.

PowerShell містить велику кількість утиліт і команди частого використання, що запускаються з командного рядка, наприклад ipconfig, ping та ін. Також для зручності команди та утиліти частого використання в PowerShell мають синоніми (Alias), наприклад cls – це синонім командлета Clear-Host, dir – синонім Get-ChildItem (повний список синонімів можна отримати за допомогою командлета Get-Alias).

Для спрощення пошуку потрібної команди в PowerShell є спеціальний командлет Get-Command, за допомогою якого можна здійснювати пошук, як за дієсловом, так і за іменником. Усі команди PowerShell згруповані у модулі (наприклад, Hyper-V, NetTCPIP), що також спрощує пошук потрібної команди. Після того як потрібна команда знайдена, можна переглянути довідку по роботі з цією командою (командлет Get-Help).

Приклад:

Отримати довідку для командлета Get-Command можна таким чином:
Get-Help Get-Command

Довідка у PowerShell може бути короткою і детальною (параметр – Detailed), повною (параметр – Full), або виводити тільки приклади (параметр – Examples).

Приклад:

Наступна команда покаже тільки приклади використання командлета Get-Command:

Get-Help Get-Command -Examples

Довідку PowerShell можна оновлювати за допомогою команди:
Update-Help.

Дізнатися поточну версію PowerShell можна за допомогою властивості PSVersion вбудованої змінної \$PSVersionTable.

Приклад:

\$PSVersionTable.PSVersion

або

Get-Variable -Name PSVersionTable -ValueOnly

де, значення PSVersion і буде версією PowerShell.

Коментарі у PowerShell починають з символу (#), блок коментарів обмежується символами (<#) з початку та (#>) у кінці.

Оболонка Windows PowerShell

Оболонка Windows PowerShell – це середовище виконання команд і сценаріїв мовою PowerShell. Оболонка має ті самі можливості що і командний рядок, наприклад, зберігання історії виконання команд, визначення вигляду оболонки, завершення виконання команд сполучення клавіш Ctrl + C та ін., яких немає в оболонці командного рядка, наприклад, "підсвічування синтаксису" (з'явилася в PowerShell 5.0).

Запустити PowerShell можна наступними способами:

- з командного рядка, набравши PowerShell;
- через діалогове вікно "Виконати"» (Win + R), також набравши PowerShell;
- у Windows 7 – Пуск -> Всі програми -> Стандартні -> Windows PowerShell -> Windows PowerShell;
- у Windows 8.1 або Windows Server 2012 R2 – Пуск -> Всі програми -> Службові -> Windows PowerShell;
- у Windows 10 або Windows Server 2016 – Пуск -> Всі програми -> Каталог Windows PowerShell (у групі W) -> Windows PowerShell.

PowerShell пропонує як консольний інтерфейс, так і середовище для розробки PowerShell ISE (Integrated Scripting Environment, вбудоване скриптове оточення). Щоб запустити PowerShell ISE, потрібно через діалогове вікно "Виконати"» (Win + R) набрати PowerShell ISE.

Мова PowerShell

PowerShell – це об'єктно-орієнтована скриптова мова програмування. Вона використовується для написання команд управління всіма компонентами операційної системи Windows в оболонці PowerShell, а також для написання сценаріїв автоматизації завдань адміністрування в інтегрованому середовищі сценаріїв Windows PowerShell (ISE).

Мова PowerShell хоча і створена для завдань адміністрування, є повноцінною скриптовою мовою програмування, оскільки має програмні конструкції, які присутні в кожній мові програмування, наприклад: умови, цикли, обробка помилок, робота зі змінними, об'єктами, масивами.

Мова PowerShell має єдиний синтаксис написання команд і структуру іменування цих команд за принципом "Дієслово-Іменник", що робить дану мову інтуїтивно зрозумілим як для програмістів, так і для системних адміністраторів.

Командлети

Командлет (cmdlet) – це команда PowerShell з завчасно визначеною функцією, подібна до умовного оператора у інших мовах програмування, за допомогою якої можна здійснювати взаємодію з об'єктами операційної системи з метою їх управління.

Командлети побудовані за принципом «Дієслово-Іменник», розділені дефісом (-): спочатку вказуємо, що робити, а через дефіс - над чим. Наприклад, командлет Get-Help означає «Показати-Довідку».

Командлети повертають результати у вигляді об'єктів, що є однією з головних відмінностей від командного рядка Windows, в якому команди повертаються тільки у вигляді тексту на екрані.

Командлети мають наступні ключові особливості:

- існують системні, користувацькі та опціональні команд лети;
- результатом виконання командлета буде об'єкт або масив об'єктів;
- команд лети можуть обробляти дані та передавати їх іншим команд летам за допомогою конвеєра;
- команди несуттєві до реєстру;
- символ ";" є роздільником.

Приклад:

Get-Process — відобразити поточні процеси, запущені на комп'ютері;

Get-Service — відобразити список служб та їх статус;

Get-Content — відобразити вміст зазначеного файлу.

Наприклад: Get-Content C:\Windows\System32\drivers\etc\hosts.

Приклад:

Get-Command -CommandType cmdlet

У командлетів є параметри, за допомогою яких можна конкретизувати їх дії. Параметри бувають обов'язкові та необов'язкові, наприклад, у командлета Get-Command обов'язкових параметрів немає.

PowerShell ISE автоматично пропонує доступні параметри з відображенням їх типу. Наприклад, Get-Service-NameW* виведе список служб, у яких ім'я починається з W. Якщо необхідно переглянути параметри введеного командлета, підійде команда: Get-Member.

Приклад:

Get-Process | Get-Member:

Так само можливо отримати параметри за допомогою параметра Examples:

Окремі командлети можуть бути викликані за допомогою аласів, наприклад, замість Get-Help можна просто написати Help.

Крім командлетів на отримання даних (Get), існують і такі типи командлетів як:

- Add – додавання даних;
- Clear – очистити;
- Enable – ввімкнути;
- Disable – вимкнути;
- New – створити;
- Remove – видалити;
- Set – задати;
- Start – запустити;
- Stop – зупинити;
- Export – експортувати;
- Import – імпортувати

Повний перлік командлетів PowerShell можна отримати за допомогою Get-Command (Get-Help-Category).

На зображенні представлений спосіб пошуку командлета за словом (параметр Verb).

```
PS C:\Users\Администратор> Get-Command -Verb Restart
```

CommandType	Name	Version	Source
Function	Restart-NetAdapter	2.0.0.0	NetAdapter
Function	Restart-PcsvDevice	1.0.0.0	PcsvDevice
Function	Restart-PrintJob	1.1	PrintManagement
Cmdlet	Restart-Computer	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Restart-Service	3.1.0.0	Microsoft.PowerShell.Management

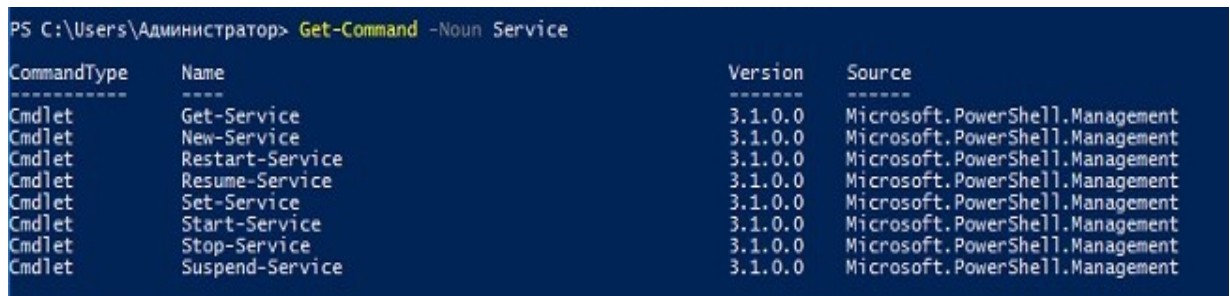
```
PS C:\Users\Администратор>
```

Рисунок 2.1 – Пошук за словом.

На рис. 2.1 наведено список командлетів, які вміють щось перезапускати.

Команда: `Get-Command -Verb Restart`

Для пошуку командлетів за іменником необхідно використовувати параметр `Noun`. Наприклад, на рис. 2.2 наведено список командлетів, які працюють зі службами.



CommandType	Name	Version	Source
-----	----	-----	-----
Cmdlet	Get-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	New-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Restart-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Resume-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Set-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Start-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Stop-Service	3.1.0.0	Microsoft.PowerShell.Management
Cmdlet	Suspend-Service	3.1.0.0	Microsoft.PowerShell.Management

Рисунок 2.2 – Команд лети, які працюють зі службами.

Команда: `Get-Command -Noun Service`

Якщо Ви не знайшли потрібний командлет за повною назвою, можна використати маску у форматі `* Текст *`.

Так само можна створювати власні командлети.

Конвеєр

Конвеєр – це передача результату роботи командлета через вертикальну риску (`|`) іншому команд летові (оскільки командлети працюють з об'єктами і повертають об'єкти, відповідно по конвеєру передаються також об'єкти).

За допомогою конвеєра можна виконувати складні завдання простим способом без написання складних алгоритмів.

Наприклад, отримати назву самого великого файлу у каталозі «`C:\Windows\System32`» (простий приклад конвеєра).

Команда:

`Get-ChildItem -Path "C:\Windows\System32" -File | Sort-Object length -Descending | Select-Object -First 1`

де, `Get-ChildItem` – командлет отримання об'єктів у зазначеному каталозі; `Sort-Object` – командлет для сортування об'єктів (у розглянутому випадку сортування за розміром файлу (`length -Descending`)); `Select-Object` – командлет вибору потрібних властивостей об'єкта (виводить стандартні поля і тільки найперший об'єкт, тобто великий файл (параметр `-First 1`)).

Усі командлети відокремлені один від одного вертикальною лінією. Результат роботи кожного командлета передається на вхід іншого командлета, тобто спочатку отримуємо всі об'єкти в зазначеному каталозі,

потім сортуємо отриманий результат і на закінчення вибираємо найперший об'єкт.

Приклади:

1. Сортувати запущені служби за статусом:

```
Get-Service | Sort-Object -property Status
```

2. Записати текст до файлу:

```
"Група ІТУ-19-1" | Out-File C:\Documents\test.txt
```

Так само можна використовувати декілька конвеєрів.

Приклад:

Вивести перелік служб ОС, за винятком тих, які зупинені:

```
Get-Service | WHERE {$_.status -eq "Running"} |  
SELECT displayname
```

Фонове виконання завдань

У PowerShell є можливість фонового виконання завдань, механізм, за допомогою якого можна запустити на виконання команду (наприклад, таку, яка довго виконується) у фоновому режимі, тобто після запуску Ви повертаєтеся у поточну сесію і можете продовжити роботу, не чекаючи закінчення роботи команди.

Дана можливість корисна тоді, коли виникає необхідність запустити команду, робота якої займає досить тривалий час. Протягом цього часу сесія оболонки PowerShell блокується до завершення роботи команди, а потрібно продовжувати працювати.

Усіма завданнями, які запущені у фоновому режимі, можна управляти, наприклад, подивитися список завдань, зупинити роботу завдання, видалити завдання та подивитися результат роботи завдання.

Командлети для роботи з фоновими завданнями:

- Start-Job – запустити фонове завдання;

- Stop-Job – зупинити фонове завдання;

- Get-Job – переглянути список фонових завдань;

- Receive-Job – переглянути результат виконання фонового завдання;

- Remove-Job – видалити фонове завдання;

- Wait-Job – перевести фонове завдання на передній план, для того щоб дочекатися його завершення.

Для запуску у фоновому режимі необхідно написати команду Start-Job, а у фігурних дужках {} команду або набір команд, які необхідно виконати у фоновому режимі.

Приклад:

1. Запустити задачу перегляду списку служб ОС у фоновому режимі:
Start-Job {Get-Service}

2. Переглянути список фонових завдань: Get-Job

3. Переглянути результати виконання завдання: Receive-Job Job1

```
PS C:\Users\Администратор> Start-Job {Get-Service}

Id      Name      PSJobTypeName State      HasMoreData Location      Command
----      -
1       Job1      BackgroundJob Running     True         localhost     Get-Service

PS C:\Users\Администратор> Get-Job

Id      Name      PSJobTypeName State      HasMoreData Location      Command
----      -
1       Job1      BackgroundJob Completed   True         localhost     Get-Service

PS C:\Users\Администратор> Receive-Job Job1

Status  Name      DisplayName
-----  -
Stopped AJRouter  Служба маршрутизатора AllJoyn
Stopped ALG    Служба шлюза уровня приложения
Stopped AppIDSvc Удостоверение приложения
Stopped AppInfo  Сведения о приложении
Stopped AppMgmt  Управление приложениями
Stopped AppReadiness Готовность приложений
Stopped AppVClient Microsoft App-V Client
Stopped AppXSvc  Служба развертывания AppX (AppXSVC)
Stopped AudioEndpointBu... Средство построения конечных точек ...
Stopped Audiosrv Windows Audio
Stopped AxInstSV Установщик ActiveX (AxInstSV)
Running BFE       Служба базовой фильтрации
Stopped BITS     Фоновая интеллектуальная служба пер...
Running BrokerInfrastru... Служба инфраструктуры фоновых задач
Stopped Browser  Браузер компьютеров
Stopped bthserv  Служба поддержки Bluetooth
Running CDPSvc   Служба платформы подключенных устро...
Stopped CDPUserSvc_id515 CDPUserSvc_id515
Stopped CertPropSvc Распространение сертификата
Stopped ClipSvc  Служба лицензий клиента (ClipSvc)
Stopped COMSysApp Системное приложение COM+
Running CoreMessagingRe... CoreMessaging
```

Рисунок 2.3 – Результаты выполнения задания у фоновому режимі.

Віддалене управління на PowerShell

PowerShell розрахована не тільки на локальне використання, але і на віддалене виконання команд (для управління віддаленими комп'ютерами). Існує кілька способів віддаленого управління.

1. За допомогою параметра `-ComputerName` (є у багатьох команд). У даному випадку ім'я комп'ютера, на якому необхідно виконати команду, передається як параметр. Спосіб має недолік: обмежується виконанням однієї команди.

2. За допомогою сесій. Командлет `Enter-PSSession` (інтерактивний сеанс) дозволяє під'єднатися до віддаленого комп'ютера і всі команди, набрані у PowerShell, будуть виконані на віддаленому комп'ютері. Спосіб має недолік: сеанс обмежується одним комп'ютером.

Приклад (під'єднання до віддаленого комп'ютера з ім'ям `ServerName`):
`Enter-PSSession ServerName`

3. За допомогою командлета `Invoke-Command`. За допомогою даного способу можна виконувати команди або сценарії як на одному комп'ютері, так і на кількох.

Наприклад, щоб підключитися до віддаленого комп'ютера (в прикладі нижче) інтерактивним сеансом виконайте наступну команду:

Сценарії, функції та модулі у PowerShell

PowerShell дозволяє зберігати, а потім використовувати, написані блоки команд у вигляді сценаріїв (зберегти текстовий файл з розширенням .ps1). Для виконання сценаріїв необхідно запустити в оболонці PowerShell. При цьому необхідно або вказати повний шлях до файлу сценарію, або перейти до каталогу зі сценарієм і викликати його за ім'ям.

Примітка: За замовчуванням виконання сценаріїв у ОС Windows заборонено!

Для того щоб подивитися політику виконання сценаріїв виконайте командлет Get-ExecutionPolicy. Результатом буде одне зі значень:

- Restricted – блокується виконання будь-яких сценаріїв (значення за замовчуванням);
- AllSigned – дозволено виконання сценаріїв, які мають цифровий підпис;
- RemoteSigned – дозволено виконання локальних сценаріїв (власних скриптів), усі викачані сценарії повинні мати цифровий підпис;
- Unrestricted – дозволено виконання будь-яких сценаріїв (не рекомендується, оскільки небезпечно!).

Для дозволу виконання сценаріїв необхідно використати командлет Set-ExecutionPolicy з одним із параметрів.

Приклад:

Set-ExecutionPolicy RemoteSigned

До сценаріїв можна передавати параметри, робити їх обов'язковими або задавати значення за замовчуванням.

У PowerShell передбачений механізм створення власних функцій, які так само як і вбудовані командлети можна використати в оболонці. Для цього необхідно вказати ключове слово Function, а потім у фігурних дужках {} написати алгоритм роботи цієї функції, тобто набір команд (наприклад, створити користувача з певними правами або видалити файли з обраної теки). Потім необхідно зберегти все це у сценарій, але тільки вже з розширенням .psm1 (тепер цей файл буде вже модулем). Файл необхідно помістити у каталог, в якому PowerShell шукає модулі. Таких каталогів декілька (спеціальний каталог у профілі користувача, каталог, де встановлений PowerShell), їх можна подивитися у змінних оточення PowerShell. Для цього потрібно виконати команду:

Get-ChildItem Env: \ PSModulePath | Format-Table –AutoSize

Після цього можна звертатися до власноруч створених функцій як до звичайних команд PowerShell.

Інтегроване середовище сценаріїв Windows PowerShell (ISE)

Для того щоб було зручно писати сценарії, функції та модулі, існує спеціальна графічна програма Integrated Scripting Environment.

Запустити її можна наступним чином:

- Windows 7 – Пуск -> Всі програми -> Стандартні -> Windows PowerShell -> Windows PowerShell ISE;
- Windows 10 або Windows Server – Пуск -> Всі програми -> Каталог Windows PowerShell (в групі W) -> Windows PowerShell ISE.

Примітка! ISE не працюватиме на системі Windows Server, встановленій у варіанті Server Core.

Довідник командлетів

Щоб отримати повний перелік командлетів у PowerShell, виконайте наступну команду:

`Get-Command -CommandType cmdlet`

1. Командлети, які часто використовують:

`Get-Help` – показує довідку по командлети, функції та загальну довідку по Windows PowerShell. Довідка буває декількох типів: коротка, детальна, повна та тільки приклади.

`Update-Help` – завантажує і встановлює нові файли довідки, тобто оновлення довідки;

`Get-Command` – командлет пошуку потрібної команди, можна шукати як за словом, так і по іменнику, також можливе використання маски, якщо точна назва дієслова або іменника невідома;

`Get-Alias` – показує псевдоніми, всі або конкретної команди;

`Get-PSDrive` – показує приєднані диски;

`Get-Member` – виводить властивості і методи, які є у об'єкта;

`Get-WindowsFeature` – інформує про доступні ролі і компоненти сервера;

`Install-WindowsFeature` (еквівалентний `Add-WindowsFeature`) – встановлює ролі або компоненти на визначений сервер;

`Uninstall-WindowsFeature` (еквівалентний `Remove-WindowsFeature`) – видаляє ролі або компонента сервера;

`Get-History` – повертає список команд, введених під час поточної сесії.

2. Робота зі змінними

У PowerShell для того щоб створити змінну, задати їй значення або отримати це значення зазвичай використовують знак долар (\$), але для створення змінних також існують спеціальні командлети.

`Get-Variable` – виводить список змінних і їх значення (або одну вказану змінну);

`New-Variable` – створює нову змінну;

`Set-Variable` – задає значення змінної. Якщо змінна з вказаним ім'ям не існує, то вона буде створена;

`Clear-Variable` – видаляє значення змінної;

`Remove-Variable` – видаляє змінну і її значення.

3. Форматування у PowerShell

У PowerShell існує набір командлетів, які призначені для форматування виводу результату роботи командлетів. Вони дозволяють користувачеві відобразити результат у вигляді, зручному для перегляду.

Format-List – відображення результату команди у форматі списку властивостей (новий рядок – окрема властивість);

Format-Table – відображення результату команди у вигляді таблиці;

Format-Wide - відображення результату команди у вигляді широкої таблиці, в якій відображається тільки одна властивість кожного об'єкта;

Format-Custom – у даному випадку форматування результату відбувається з використанням призначеного для користувача подання.

4. Імпорт та експорт

PowerShell дозволяє імпортувати/експортувати дані в різних поширених форматах, наприклад, CSV або XML, а також перенаправляти висновок результату роботи команди у зовнішній файл або на принтер.

Export-Csv – експорт даних у формат CSV;

Import-Csv – імпортує дані з CSV-файлу;

Export-Clixml – експорт даних у формат XML;

Import-Clixml – імпортує файл CLIXML і створює відповідні об'єкти в оболонці PowerShell;

Out-File – посилає вивід результату роботи командлета у зовнішній файл (наприклад, до TXT);

Out-Printer – вивід результату роботи команди на принтер;

Import-Module – додає модулі у поточній сесії.

5. Робота з мережею

Для адміністрування мережі у PowerShell існують наступні командлети.

Disable-NetAdapter – командлет вимикає мережевий адаптер;

Enable-NetAdapter – даний командлет вмикає мережевий адаптер;

Rename-NetAdapter – перейменовує мережевий адаптер;

Restart-NetAdapter – перезапускає мережевий адаптер;

Get-NetIPAddress – виводить інформацію про конфігурацію IP-адреси;

Set-NetIPAddress – змінює конфігурацію IP-адреси;

New-NetIPAddress – створює і налаштовує IP-адресу;

Remove-NetIPAddress – видаляє IP-адресу та її конфігурацію;

Get-NetRoute – виводить таблицю маршрутизації IP;

Set-NetRoute – змінює таблицю маршрутизації IP;

New-NetRoute – створює запис у таблиці маршрутизації IP;

Remove-NetRoute – видаляє один або кілька записів (IP маршрутів) з таблиці маршрутизації IP;

Get-NetIPv4Protocol – виводить інформацію про конфігурацію протоколу IPv4;

Get-NetIPv6Protocol – виводить інформацію про конфігурацію протоколу IPv6;

Get-NetIPInterface – виводить інформацію про властивості інтерфейсу IP;

Get-NetTCPSetting – показує інформацію про параметри та конфігурацію TCP;

Test-Connection – командлет посилає ICMP пакети до одного або декількох комп'ютерів, тобто "пінгує" комп'ютери.

6. Робота з елементами

У PowerShell є командлети, які вміють працювати з елементами (файли, теки, ключі реєстру та ін.).

Clear-Item – очищає вміст елемента, але не видаляє сам елемент;

Copy-Item – копіює елемент;

Get-Item – отримує елемент у зазначеному місці;

Invoke-Item – виконує дію за замовчуванням над зазначеним елементом;

Move-Item – переміщує елемент;

New-Item – створює новий елемент;

Remove-Item – видаляє зазначені елементи;

Rename-Item – перейменовує елемент у просторі імен постачальника PowerShell;

Set-Item – змінює елемент;

Get-ChildItem – повертає елементи і дочірні елементи в одному або декількох визначених місцях;

Get-Location – виводить інформацію про поточне місцезнаходження.

7. Командлети для роботи з Active Directory (AD)

PowerShell дозволяє працювати зі службою каталогів Active Directory (тут наведені тільки окремі командлети).

New-ADUser – створення нового користувача в Active Directory;

Get-ADUser – виводить інформацію про користувачів Active Directory;

Set-ADUser – змінює користувача Active Directory;

Remove-ADUser – видаляє користувача Active Directory;

New-ADGroup – командлет створює групу в Active Directory;

Get-ADGroup – виводить інформацію про групу або виконує пошук, щоб отримати кілька груп з Active Directory;

Set-ADGroup – командлет змінює групу в Active Directory;

Remove-ADGroup – видалення групи в Active Directory;

Add-ADGroupMember – командлет додає облікові записи користувачів, комп'ютерів або груп в якості нових членів групи Active Directory;

Get-ADGroupMember – виводить інформацію про членів групи Active Directory;

Remove-ADGroupMember – видалення елементів з групи Active Directory;

Set-ADAccountPassword – скидання пароля облікового запису Active Directory;

Disable-ADAccount – від'єднує обліковий запис Active Directory;

Enable-ADAccount – під'єднує обліковий запис Active Directory;

Unlock-ADAccount – розблокує обліковий запис Active Directory;

New-ADComputer – створення нового облікового запису комп'ютера в Active Directory;

Get-ADComputer – виводить інформацію про один або декілька комп'ютери в Active Directory;

Set-ADComputer – зміна облікового запису комп'ютера в Active Directory;

Remove-ADComputer – видалення комп'ютера з Active Directory.

8. Робота з Hyper-V

New-VM – створення нової віртуальної машини;

Set-VM – налаштування віртуальної машини;

Start-VM – запуск віртуальної машини;

Stop-VM – закриття, вимикання або збереження віртуальної машини;

Import-VM – імпорт віртуальної машини з файлу;

Move-VM – переміщення віртуальної машини на новий Hyper-V хост;

Remove-VM – видалення віртуальної машини;

Rename-VM – перейменування віртуальної машини;

New-VHD – створення одного або декількох нових віртуальних жорстких дисків;

Set-VHD – налаштування віртуального жорсткого диска;

Test-VHD – тестування віртуального жорсткого диска на предмет виявлення проблем, які зробили б його непридатним для використання;

Add-VM DVD Drive – додає DVD диск до віртуальної машини;

Remove-VM DVD Drive – видаляє DVD-диск з віртуальної машини;

Add-VM Hard Disk Drive – додає жорсткий диск до віртуальної машини;

Remove-VM Hard Disk Drive – видаляє один або кілька віртуальних жорстких дисків (VHD) з віртуальної машини;

Add-VM Network Adapter – додає віртуальний мережевий адаптер на віртуальній машині;

Remove-VM Network Adapter – видаляє один або кілька віртуальних мережевих адаптерів з віртуальної машини;

Copy-VM File – копіювання файлів на віртуальну машину;

Get-VM Video – виводить інформацію про налаштування відео для віртуальних машин;

Move-VM Storage – переміщення сховища віртуальної машини.

9. Робота з фоновими завданнями

Start-Job – запустити фонове завдання;

Stop-Job – зупинити фонове завдання;

Get-Job – подивитися список фонових завдань;

Receive-Job – подивитися результат виконання фонових завдань;
Remove-Job – видалити фонове завдання;
Wait-Job – перевести фонове завдання на передній план, для того щоб чекати його закінчення.

10. Робота з об'єктами

Measure-Object – командлет дозволяє розраховувати на основі властивостей об'єктів такі числові агрегуючі параметри як: мінімальне, максимальне, середнє значення, суму і кількість. Наприклад, потрібно дізнатися максимальний або середній розмір файлу у визначеному каталозі, або просто дізнатися кількість файлів (запущених процесів, служб та ін.);

Select-Object – за допомогою даного командлет можна вибрати певні об'єкти або властивості цих об'єктів. Наприклад потрібно вивести тільки назву файлу та його розмір;

Sort-Object – сортує об'єкти за значеннями властивостей;

Where-Object – командлет вказує умову для вибірки об'єктів на основі значень їх властивостей;

Group-Object – групує об'єкти, які містять однакове значення для заданих властивостей;

ForEach-Object – перебір об'єктів з метою виконання певної операції над кожним з цих об'єктів.

11. Командлети для віддаленого управління

За допомогою PowerShell можна виконувати команди не тільки на локальному комп'ютері, але і на одному/декількох віддалених.

Enter-PSSession – запускає інтерактивний сеанс з віддаленим комп'ютером;

Exit-PSSession – завершує інтерактивний сеанс з віддаленим комп'ютером;

New-PSSession – створює постійне підключення до локального або віддаленого комп'ютера;

Remove-PSSession – закриває один або кілька сеансів Windows PowerShell;

Disconnect-PSSession – від'єднується від сеансу;

Connect-PSSession – під'єднується до вимкнених сеансів;

Get-PSSession – отримує сеанси Windows PowerShell на локальних і віддалених комп'ютерах;

Invoke-Command – виконує команди на локальному і віддалених комп'ютерах.

12. Робота зі службами і процесами

Get-Process – виводить інформацію про запущені процеси на комп'ютері;

Start-Process – запускає один або кілька процесів на комп'ютері;

Stop-Process – зупиняє один або кілька запущених процесів;

Get-Service – виводить інформацію про служби;
Restart-Service – перезапускає службу;
Start-Service – запускає службу;
Stop-Service – зупиняє службу;
Suspend-Service – призупиняє роботу служби;
Set-Service – за допомогою даного командлет можна змінити властивості служби, наприклад, опис, псевдонім та режим запуску. Також його можна використовувати для запуску, зупинки або припинення служби.

13. Робота з комп'ютером

PowerShell дозволяє виконувати адміністративні завдання для операційної системи і комп'ютера загалом, наприклад, запустити знову операційну систему або перейменувати комп'ютер.

Restart-Computer – командлет перезапускає операційну систему (перезавантажує комп'ютер);

Stop-Computer – вимикає комп'ютер;

Rename-Computer – перейменовує комп'ютер;

Checkpoint-Computer – створює точку відновлення системи на локальному комп'ютері;

Restore-Computer – запускає відновлення системи на локальному комп'ютері;

Disable-ComputerRestore – вимикає функцію відновлення системи на зазначеному диску файлової системи;

Enable-ComputerRestore – вмикає функцію відновлення системи на зазначеному диску файлової системи;

Remove-Computer – видаляє локальний комп'ютер з домену;

Get-EventLog – виводить інформацію про події з журналу подій, або список журналів подій на локальному або віддаленому комп'ютері;

Clear-EventLog – видаляє записи з зазначених журналів подій.

14. Робота з контентом

Для управління контентом, так як і з текстом у файлі, існують спеціальні командлети.

Get-Content – отримує вміст елемента (наприклад, зчитує файл);

Add-Content – додає вміст до заданого елементу, наприклад, текст у файл;

Clear-Content – видаляє вміст елемента, але не видаляє сам елемент;

Set-Content – записує/замінює вміст елементу новим вмістом.

15. Інші командлети PowerShell

Get-ExecutionPolicy – за допомогою даного командлет можна дізнатися діючу політику виконання Windows PowerShell для поточного сеансу;

Set-ExecutionPolicy – командлет змінює політику виконання Windows PowerShell;

Write-Host – виводить інформацію на екран (пише текст);

Read-Host – зчитує рядок введення з консолі;
Write-Warning – виводить попередження;
Write-Error – оголошує помилку і виводить її в потік помилок;
Get-Date – повертає поточну дату і час;
Set-Date – командлет змінює системну дату і час.

Додаткова інформація:

- 1) <https://docs.microsoft.com/ru-ru/powershell/scripting/learn/ps101/02-help-system?view=powershell-7.1>
- 2) <https://docs.microsoft.com/ru-ru/powershell/scripting/overview?view=powershell-7.1>
- 3) <https://docs.microsoft.com/ru-ru/powershell/scripting/learn/ps101/04-pipelines?view=powershell-7.1>

2.3 Контрольні завдання.

Використовуючи командлети PowerShell виконати завдання ПЗ 1 відповідно до варіанту.