**Міністерство освіти і науки України Національний технічний університет**

**«Дніпровська політехніка» Факультет інформаційних технологій Кафедра інформаційних технологій та комп’ютерної інженерії**

**Звіт**

**з лабораторної роботи №5 дисципліни “Операційні системи”**

**Тема роботи: «Microsoft Windows Powershell»**

Виконав:

студент гр. *шифр групи Ініціали та прізвище студента*

Перевірив:

*посада* каф. ІТКІ *Ініціали та прізвище викладача*

Дніпро 2024

**Лабораторна робота №5**

**Microsoft Windows Powershell**

**Мета роботи:** набути навичок роботи в командній оболонці Microsoft Windows PowerShell, вивчивши основні команди і прийоми роботи.

**Хід роботи**

1. **Скрипт для запису всих команд PowerShell**

# Define paths

$homeDir = [Environment]::GetFolderPath("UserProfile")

$dataDir = Join-Path $homeDir "Data"

# Create Data directory if it doesn't exist

if (!(Test-Path $dataDir)) {

New-Item -ItemType Directory -Path $dataDir

}

# Get all PowerShell commands and save to commands.txt

Get-Command | Out-File (Join-Path $dataDir "commands.txt")

# Parse commands.txt for About help

Get-Content (Join-Path $dataDir "commands.txt") | Where-Object {

$\_.StartsWith("about\_")

} | ForEach-Object {

$commandName = $\_.Split(" ")[0]

Get-Help $commandName -Full | Out-File (Join-Path $dataDir "$commandName.txt")

}

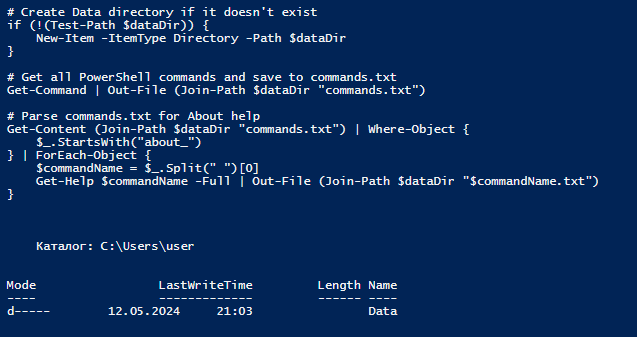


Рис.1. – Виконання скрипта в консолі

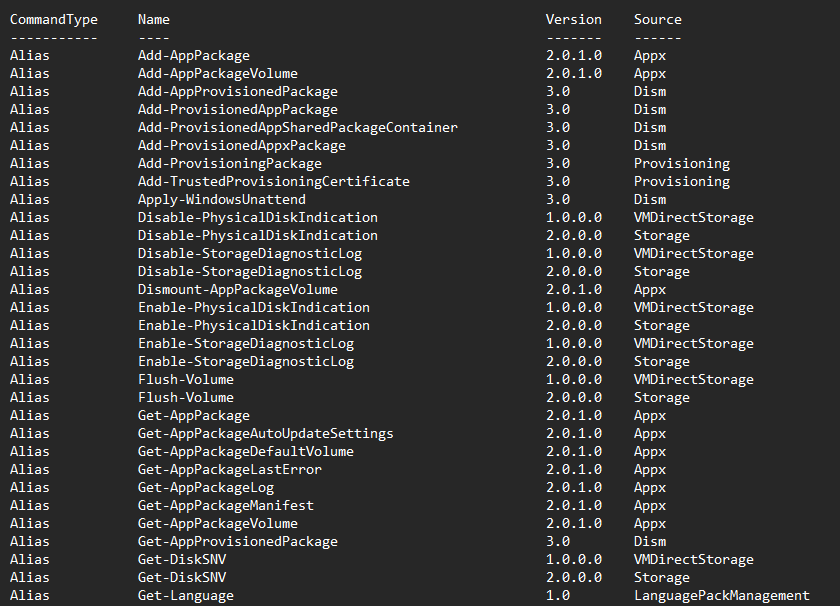
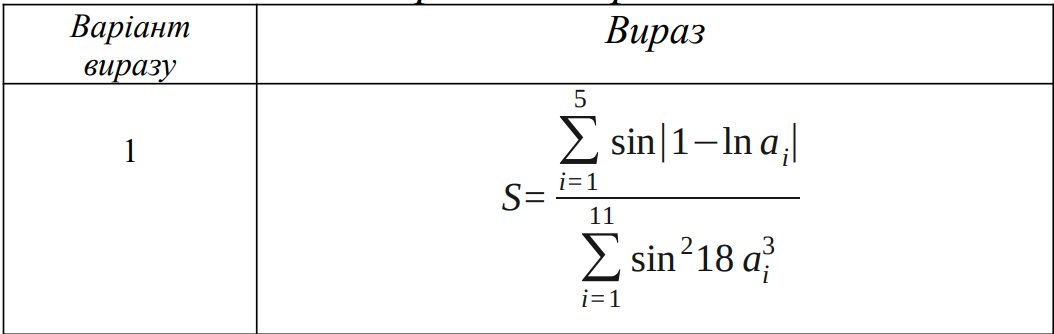
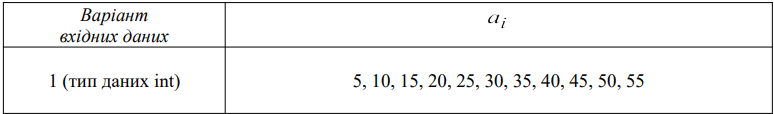


Рис.2. – Частина вихідного файлу зі списком команд, який створив скрипт

1. **Скрипт для підрахунку виразу**

****



# Define the values for ai

$aiValues = 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55

# Function to calculate S for a given ai

function CalculateS($ai) {

$numerator = [Math]::Sin([Math]::Abs(1 - [Math]::Log($ai)))

$denominator = [Math]::Pow([Math]::Sin(18 \* $ai \* $ai \* $ai), 2)

return $numerator / $denominator

}

# Calculate S for each ai value

foreach ($ai in $aiValues) {

$S = CalculateS($ai)

Write-Host "For ai = $ai, S = $S"

}

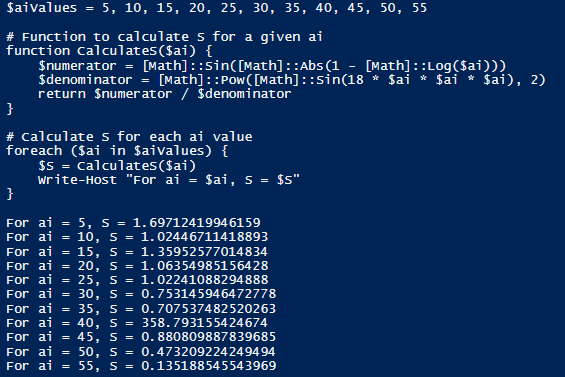


Рис.3. – Приклад виконання роботи скрипта

1. **Відповіді на запитання**
2. Версію Windows PowerShell можна визначити, використовуючи команду **$PSVersionTable.PSVersion**.
3. Командлет - це невеликий фрагмент коду, що виконує певну дію. В PowerShell це функції, які виконують конкретні завдання.
4. Для виклику довідки по командлету або інструкції використовується команда **Get-Help** за наступним шаблоном: **Get-Help назва\_командлету**.
5. Для виконання скриптів на Windows PowerShell необхідно, щоб виконувані скрипти були увімкнені. Для цього слід виконати команду **Set-ExecutionPolicy**, вибравши потрібний рівень безпеки.
6. Щоб отримати відомості про завантажені процеси в PowerShell, можна скористатися командою **Get-Process**.
7. У PowerShell існують оператори умов, такі як **-eq** (рівно), **-ne** (не рівно), **-lt** (менше), **-gt** (більше), **-le** (менше або рівне), **-ge** (більше або рівне).
8. Умовні вирази в PowerShell можна побудувати, використовуючи ключові слова **if**, **else**, **elseif**.
9. Інструкцію **Switch** використовують, коли потрібно здійснити вибір дії на основі значення змінної або виразу.
10. Для обробки виняткових ситуацій використовують конструкцію **try**/**catch**.
11. Для побудови циклів в PowerShell можна використовувати **for**, **foreach**, **while**, **do-while**. Наприклад:

for ($i = 0; $i -lt 10; $i++) {

Write-Output $i

}

1. Спецсимволи в PowerShell включають **$**, **@**, **(**, **)**, **{**, **}**, **[**, **]**, **;**, **,**, **|**, **<**, **>**, **&**, **.**, **"**, **'**, **?**, **!**, **\***, **/**, **\**, **=**, **:**.
2. Для виводу на консоль кількох рядків через роздільник можна використовувати команду **Write-Output**, або **Write-Host**, якщо потрібен вивід тільки на консоль.
3. Для виводу на консоль даних в табличному вигляді використовується команда **Format-Table**.
4. Для створення своїх об'єктів .NET Framework в PowerShell можна використовувати конструктори класів, наприклад:

$dateTime = New-Object -TypeName System.DateTime -ArgumentList 2024, 5, 12

1. Для керування файлами і каталогами використовуються командлети **Get-ChildItem**, **Copy-Item**, **Move-Item**, **Remove-Item**, **New-Item**.
2. Математичні функції можна викликати внутрішніми скриптами PowerShell, використовуючи стандартні оператори, такі як **+**, **-**, **\***, **/**, **%**. Наприклад: **$result = 10 + 5**.

**Висновок**

Під час виконання лабораторної роботи з Windows PowerShell було отримано поглиблені знання про основні аспекти роботи з цим інструментом. Зокрема, було навченося визначати версію Windows PowerShell, розуміти призначення командлетів та викликати довідку по ним. Також були опрацьовані умови необхідні для виконання скриптів, отримано відомості про процеси, виведено умовні вирази та побудовано цикли. Крім того, вивчено спецсимволи, вивід даних у табличному вигляді, створення об'єктів .NET Framework та використання командлетів для керування файлами і каталогами. Таким чином, лабораторна робота дозволила отримати практичні навички в роботі з PowerShell, які можуть бути корисні при автоматизації рутинних завдань у системі Windows.