Jenna de Beer

ST10092652

NEPR7312

Portfolio of Evidence: Part One

*Activity Two*

*Table of Contents*

**Table of Contents1**

**Introduction2**

**Purpose3**

**PowerShell Console4**

Section A: Getting Started Accessing Windows PowerShell**4-6**

*Set up and specify the default working directory*

Section B: PowerShell Console Window**7-8**

*How to use and modify PowerShell window customization options*

*How to change command console colour attributes*

Section C: PowerShell console editing features**9-10**

*How to use Windows PowerShell edit enhancements*

Section D: The Integrated Scripting Environment (ISE)**11-13**

*How to work with the Integrated Scripting Environment*

**PowerShell Script Concepts14**

Section E: Variables**14-16**

Section F: Arrays**17-19**

Section G: Hashes/Hash Tables**20-21**

**Definition of Terms22**

**References23**

*Introduction*

This Windows PowerShell tutorial covers the basics of how to use PowerShell under the following sections:

* Getting started accessing Windows PowerShell
* PowerShell Console Window
* PowerShell console editing features
* The integrated scripting environment (ISE)
* Variables
* Arrays
* Hashes/Hash Tables

Each section consists of a brief explanation of the topic covered with an explanation of the concept. After the explanations, the user will find a detailed How-To guide that walks through each step to take when learning the topic. If the user requires a more hands-on learning experience or more detailed instruction, the How-To guides each have a short, narrated video that can be found in the folder with this document. The section titles of the topics will correspond with the video guide.

Once this tutorial has been completed, the user will have a solid foundational understanding of Windows PowerShell and its uses.

*Purpose*

This tutorial has been designed to teach new ITC support and administrators the basic functions of Windows PowerShell. The tutorial will create a foundation which can be built upon in further depth as the users gain experience and encounter projects in which it may be used. Some fundamental points will be established, such as:

* How to create scripts.
* How to automate system tasks for efficiency.
* Short turnaround in deploying the system’s administrative tasks.

Once the tutorial has been completed, users will be able to use PowerShell’s most crucial features as listed from section A to section B. Users will also be able to refresh their memories with the tutorial if they wish.

*PowerShell Console*

# **Section A**: Getting Started Accessing Windows PowerShell

#### o Set up and specify the default working directory

## Brief Explanation

Windows PowerShell is a scripting and command-line tool for automating administrative tasks and managing system configurations. In PowerShell, the working directory is the location in which PowerShell starts when a new session is opened.

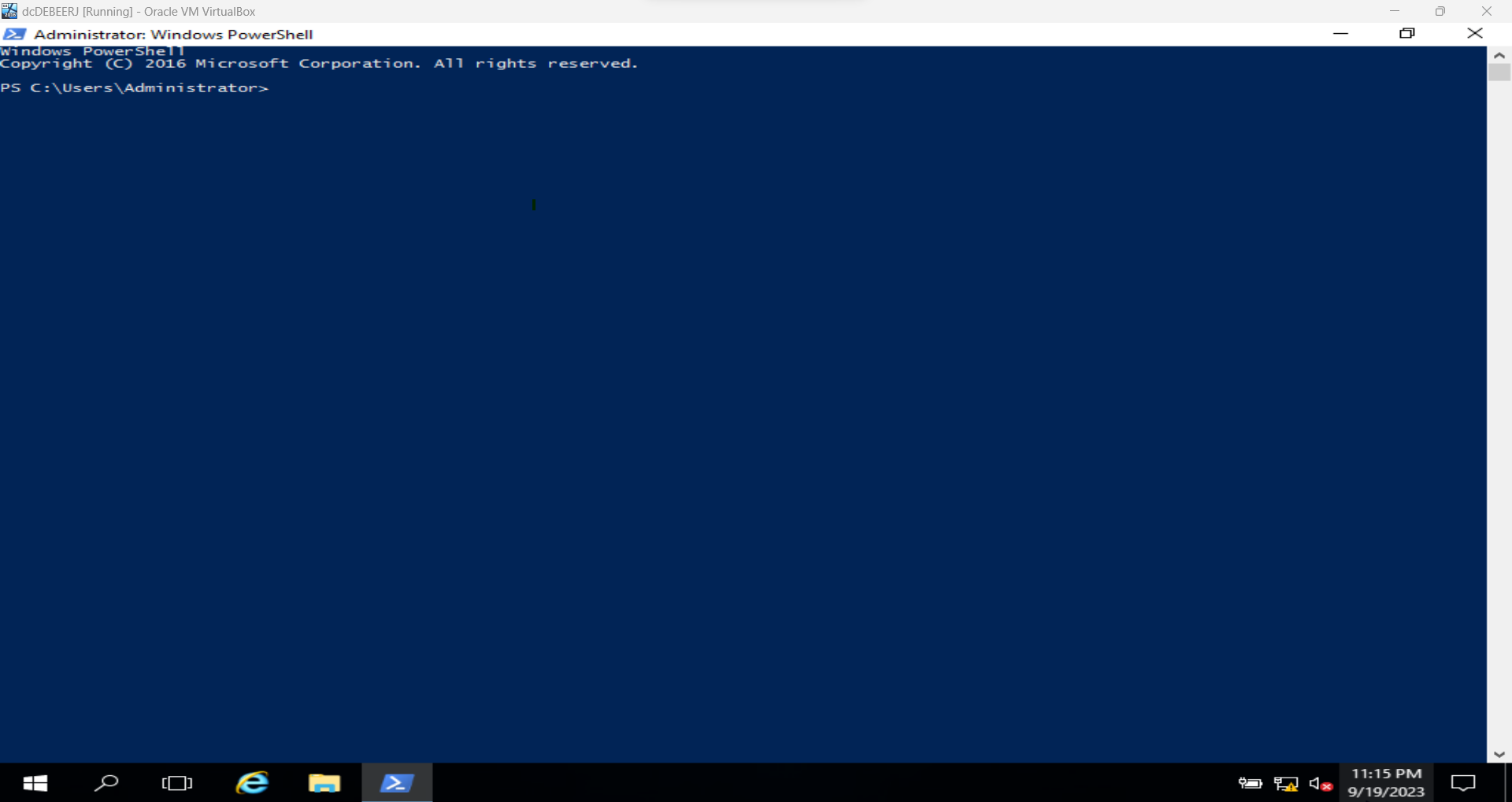
## Explanation of the Concept

The default working directory is generally found in the user profile which appears as: ‘C:\Users\YourUsername’. You can customise the working directory to a different folder location that you prefer. Changing the directory to a location that you prefer will save you the time and effort spent on navigating to that location each time you open a session.

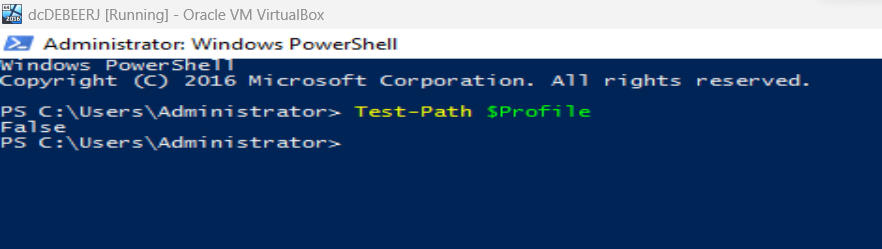
## How-To Guide

The steps below can be followed to set up and specify the default working directory in Windows PowerShell:

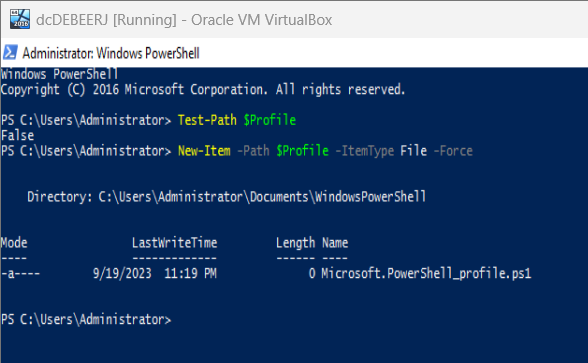
1. Open Windows PowerShell from the search bar in the Start Menu.



1. Locate or create your PowerShell profile script with the ‘Test-Path $Profile’ command. If the output is ‘True’, you have a profile script already. If the output is false, you need to create a profile with the following steps.



1. If you already have a profile, you can skip this step. To create your profile, you can run the ‘New-Item -Path $Profile -ItemType File -Force’ command. This creates the ‘Profile’ file.



1. To edit your profile, you must open it in a script editor like Notepad or Visual Studio Code. Use the following command with the name of your preferred text editor: ‘notepad.exe $Profile’.

A screenshot of a computer

Description automatically generated

1. Once in the text editor, add the ‘Set-Location -Path “C:\Your\Preferred\Path”’ line to set the default working directory. Replace the path in this example with the path that you want to use. You will need to save the changes in the script editor and restart PowerShell to test the new default directory.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

(Rath, 2020)

# **Section B**: PowerShell Console Window

#### o How to use and modify PowerShell window customisation options

#### o How to change command console colour attributes.

## Brief Explanation

Windows PowerShell allows users to customise the console window according to their own preferences for aesthetic or readability purposes. Although many features can be customised in the terminal, for the purpose of this explanation we will focus on the ability to change the colours of some elements.

## Explanation of the Concept

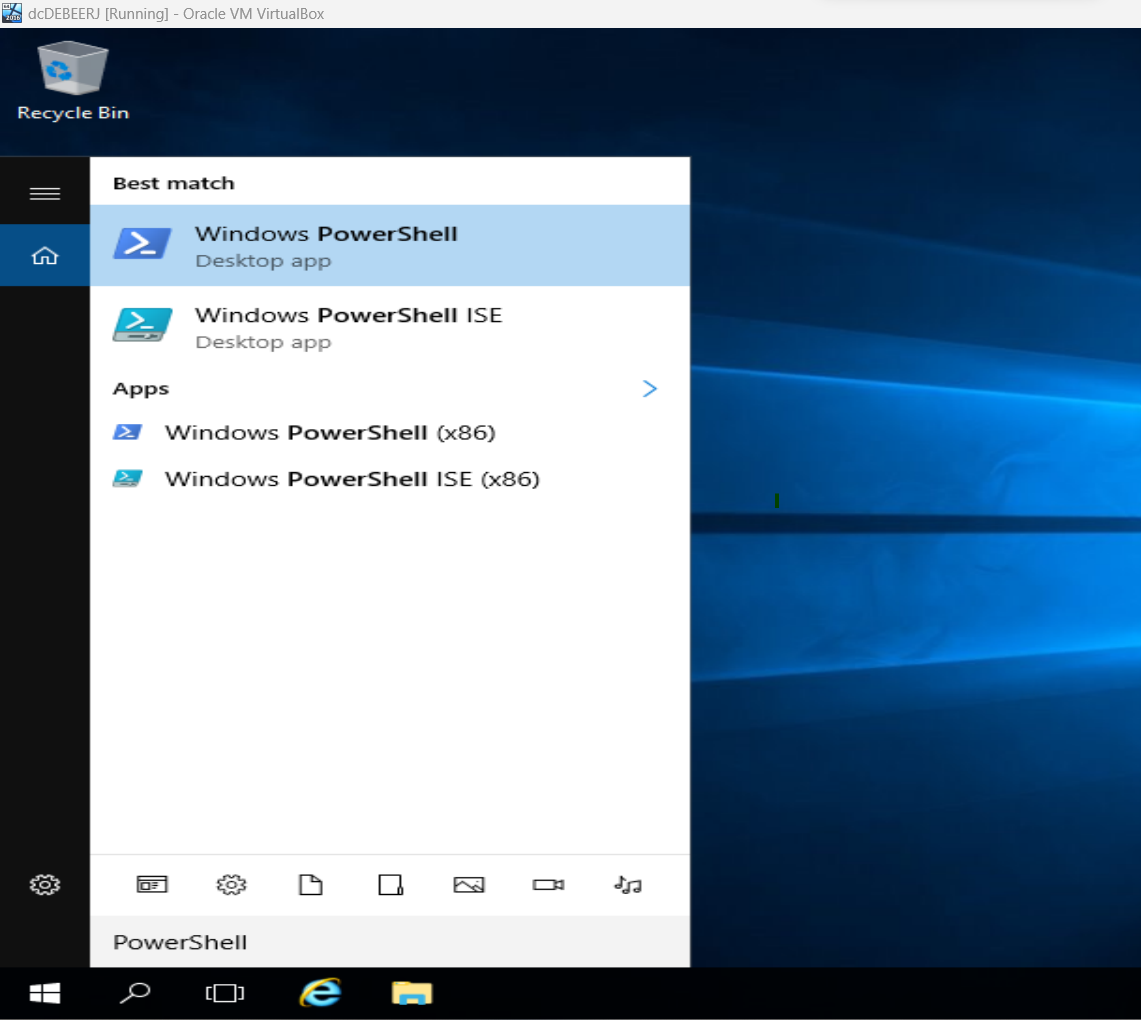
Windows PowerShell offers users a range of options for customising and personalising their console window. For example, users can change the text font or can choose colours to highlight text to enhance readability.

PowerShell provides users with this ability to customise the console and adjust elements like the fonts and colours so that the users can optimise readability in their programs. For example, changing the colour of some texts can make certain outputs or errors stand out for the user to see.

## How-To Guide

The steps below can be followed to customise your PowerShell console:

1. Open Windows PowerShell from the search bar in the Start Menu.

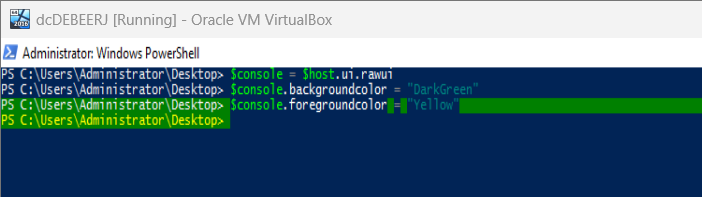


1. Before making changes, you can see the current console settings with the ‘Get-PSReadlineOption’ cmdlet.

A screenshot of a computer

Description automatically generated

1. You can change the text and background colours of the console according to your liking with the $host.ui.rawui command. You would first need to execute ‘$console = $host.ui.rawui’, then you could begin to customise the colours. For example, if you want to set the background colour to dark green, you would execute ‘$console.backgroundcolor = “DarkGreen”’. If you wanted to change the text colour to yellow, you would execute ‘$console.foregroundcolor = “Yellow”’. The changes will take effect immediately after execution. You can replace the colours in these examples according to your own preferences.



1. Once the colours have been adjusted to your preferences, the changes will be saved for future sessions.

(Kundu, 2017)

# **Section C:** PowerShell console editing features

#### o How to use Windows PowerShell edit enhancements.

## Brief Explanation

Windows PowerShell allows users to edit command lines and scripts within the console to streamline the editing process and to help users refine their input.

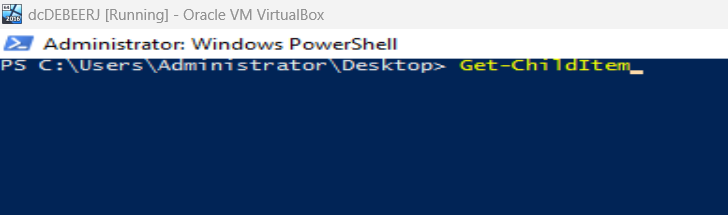
## Explanation of the Concept

PowerShell’s edit enhancements were created to make command line and script editing easier for users with functionalities like text selection, copy and paste, and keyboard shortcuts. When working with more complex scripts, these features reduce the risk of errors made by users, particularly typos.

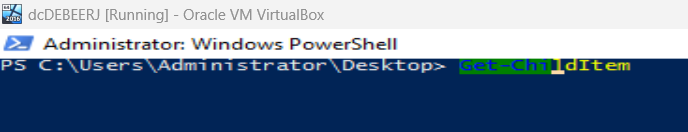
## How-To Guide

This how-to guide covers the basic editing features in PowerShell:

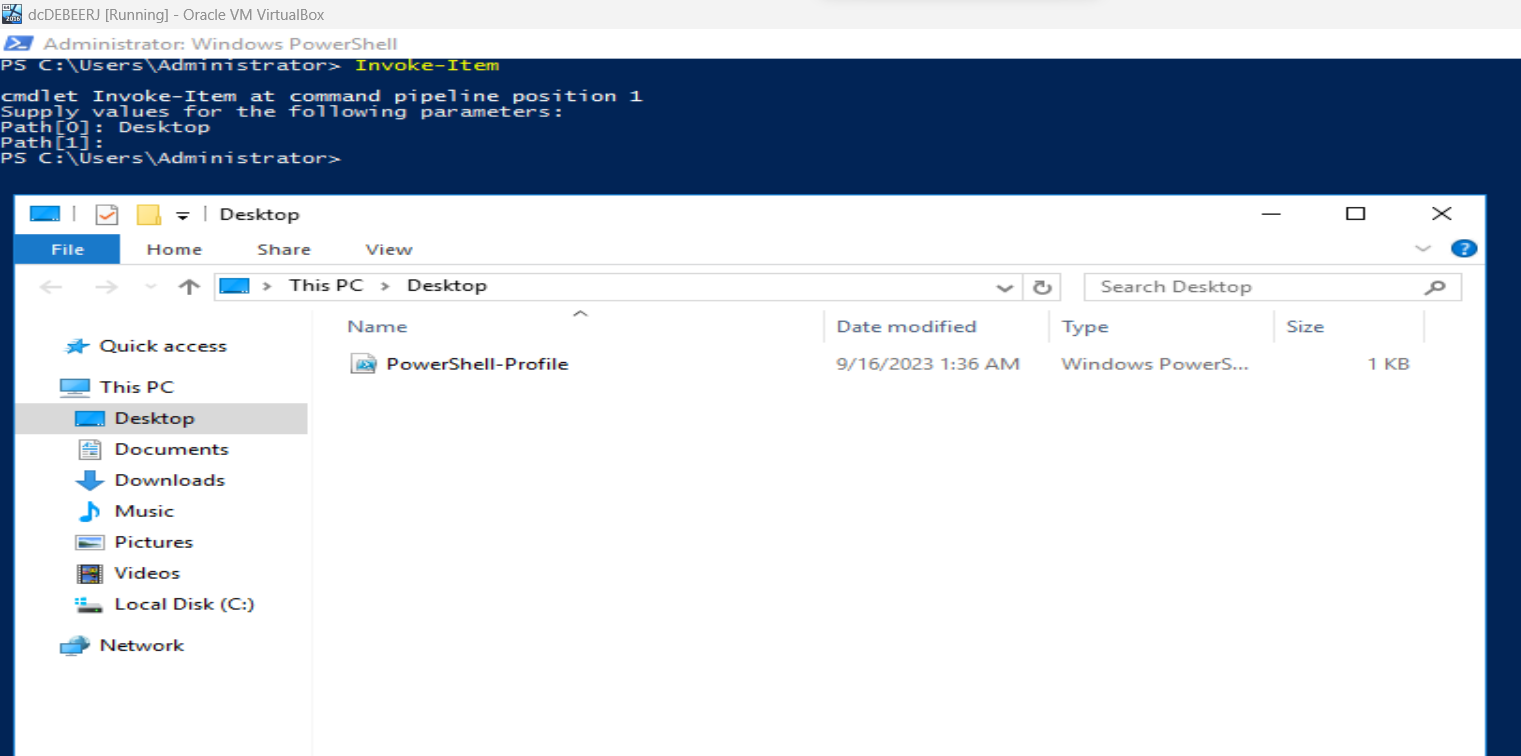
1. You can easily navigate through your command history with the ‘Up’ and ‘Down’ keys. For example, pressing the ‘Up’ key will show you the most previously executed command. If the previous executed line was ‘Get-ChildItem’ for instance, and you pressed the up arrow, the console will look like the image below:



1. You can also select text in the console with the ‘Shift’ and arrow keys to copy or remove it. For example, placing the cursor at the beginning of a line of text and holding ‘Shift’ while pressing the ‘Right’ arrow will highlight, or ‘select’, the text. The text in the image below is only partially highlighted.



1. Selected text can be copied or cut using keyboard shortcuts. Shortcuts may differ between operating systems, but generally to copy text you can use the ‘Ctrl + C’ shortcut and to paste text you can use the ‘Ctrl + V’ shortcut. There are other keyboard shortcuts than can be used in PowerShell. Some useful shortcuts to know are as follows:
   * ‘Ctrl + A’ to select all text
   * ‘Ctrl + X’ to cut selected text
   * ‘Ctrl + Z’ to undo text
   * ‘Ctrl + Y’ to redo text that was undone
2. If you want to check the current path in the folders using the graphical user interface, you can use the ‘Invoke-Item’ cmdlet.



(Doctor Scripto, 2014)

# **Section D**: The Integrated Scripting Environment (ISE)

*o How to work with the Integrated Scripting Environment.*

## Brief Explanation

The Integrated Scripting Environment is a scripting and automation tool that is used alongside Windows PowerShell. The ISE is a text editor for script development or debugging. The purpose of the ISE is to make it easier for users to create and manage PowerShell scripts.

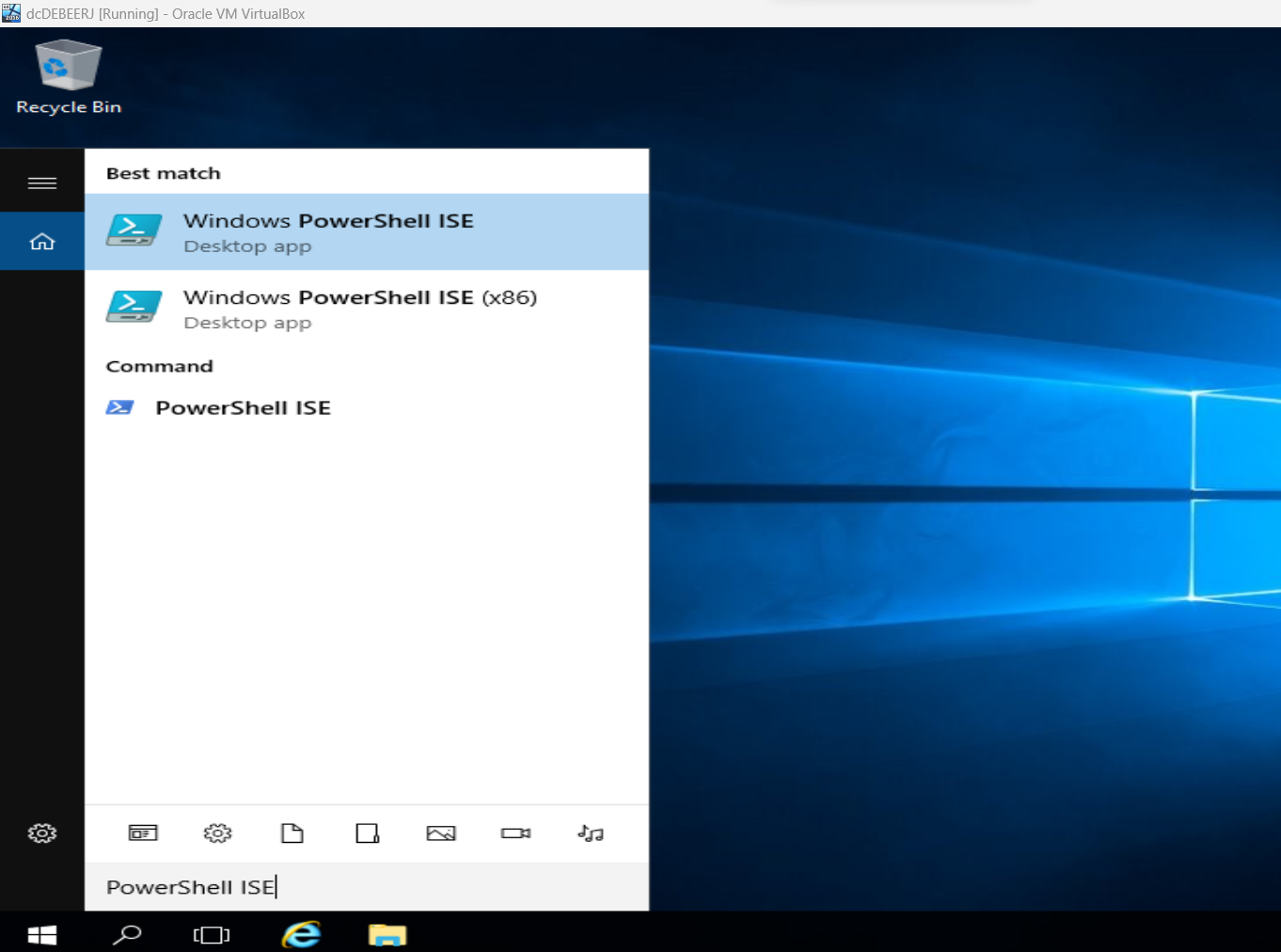
## Explanation of the Concept

The ISE is the text editor used for PowerShell scripts. It includes many features like syntax highlighting, debugging tools, and a command terminal for testing commands in the script. The ISE makes it easier to streamline script creation which makes it a good tool for PowerShell beginners.

## How-To Guide

This how-to guide covers how to work with the ISE:

1. You can open the ISE by searching for it in the ‘Start menu’.

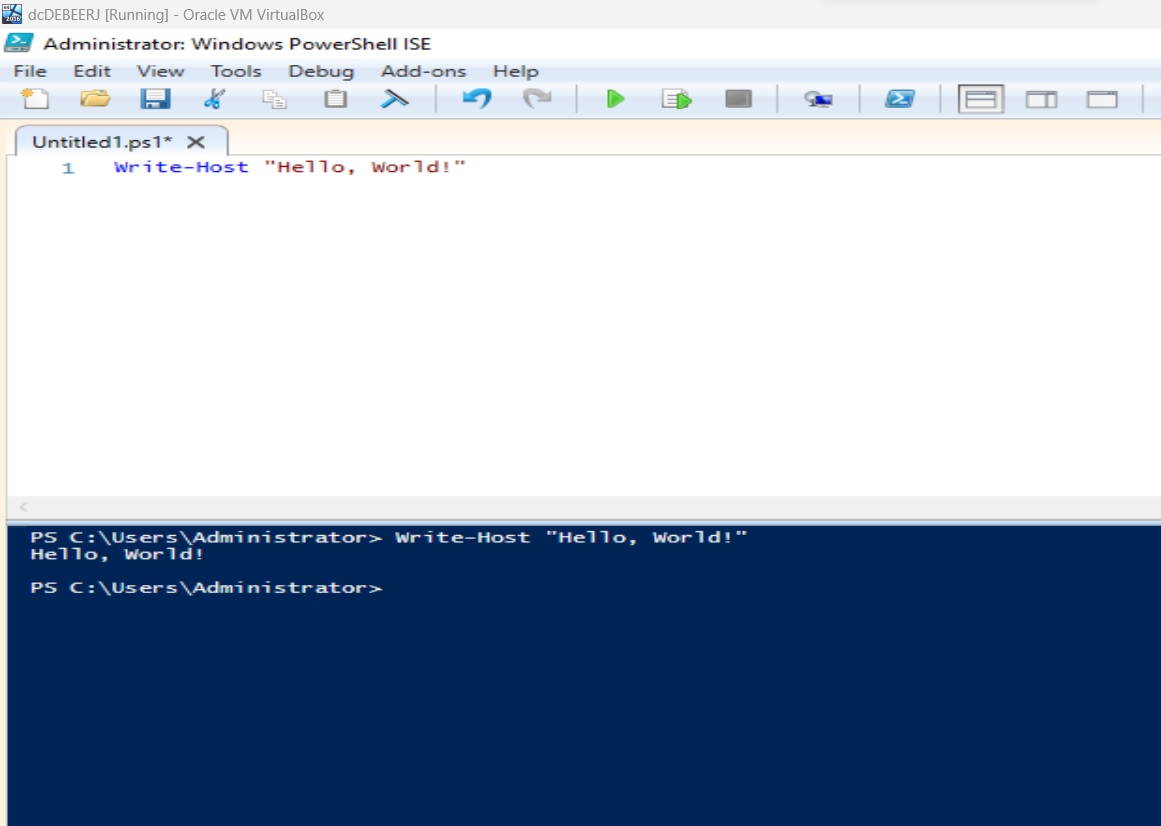


1. The ISE is made up of a few panes: The ‘Script’ pane is used for writing and editing scripts, the ‘Command’ pane is used for interactive command execution and is for displaying script output or errors.

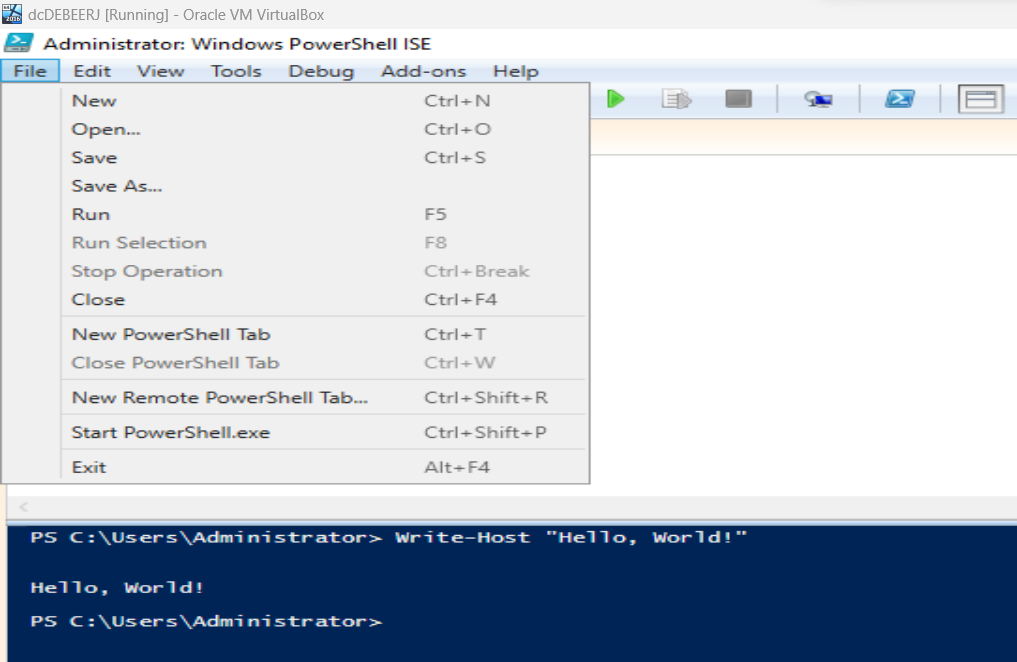
A screenshot of a computer

Description automatically generated

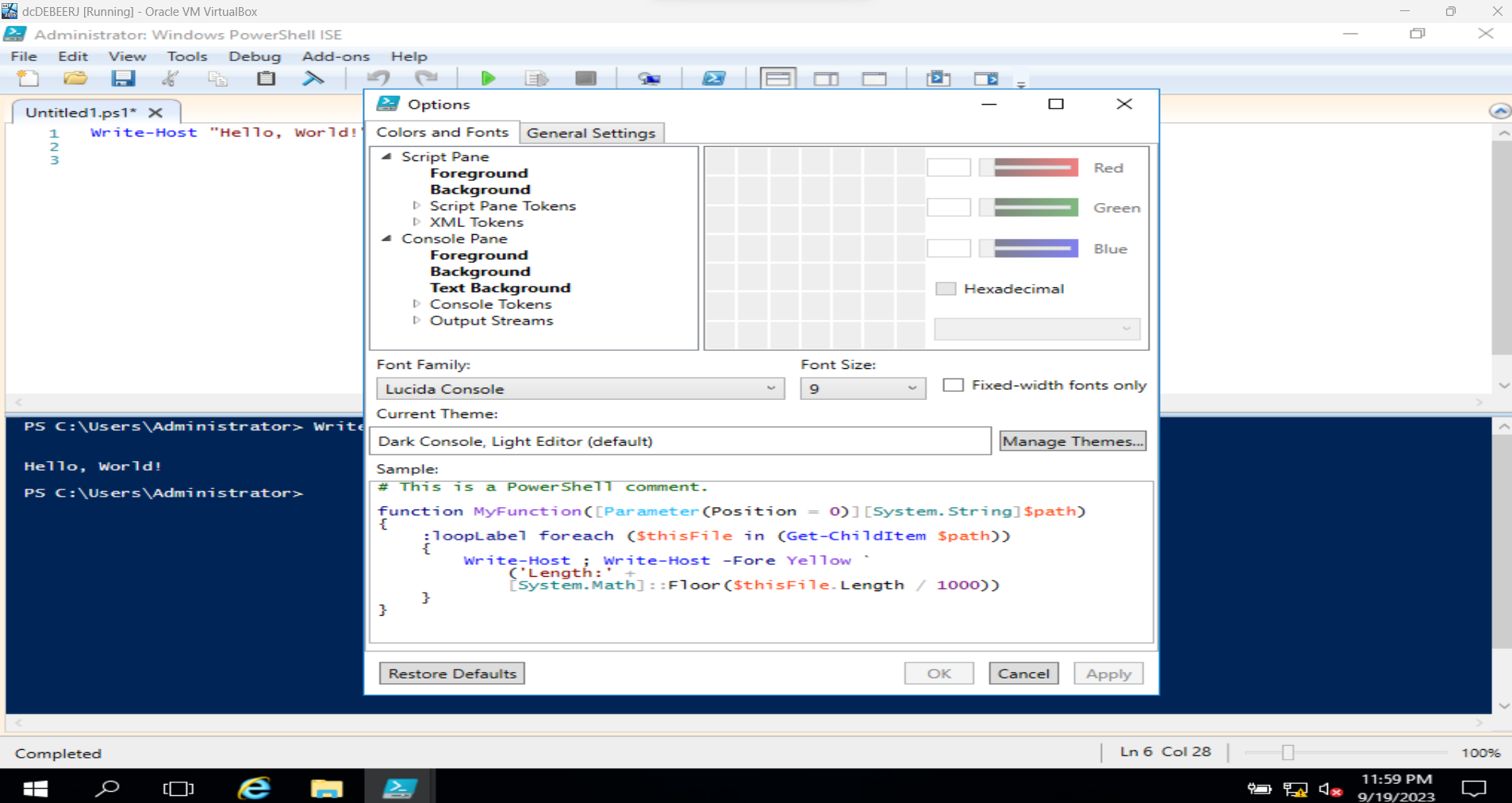
1. You can create scripts in the ‘Script’ pane. For example, a basic script could be ‘Write-Host “Hello, World!”.
2. To run the script, you will need to click the ‘Run Script’ button (green play button). The script will then be displayed in the ‘Output’ pane. If we were to follow the example from the previous step, once executed, “Hello, World!” will be displayed in the ‘Command’ pane.



1. The script can be saved into a location of your choice by selecting the ‘File’ menu and selecting ‘Save’. When editing the script once it has been saved to a file location, you can simply save using the ‘Ctrl + S’ shortcut.



1. If you wish to customise the ISE settings, you can select the ‘Options’ menu under ‘Tools’ where you can edit the font size, colours, and other advanced features like the script execution policies.



(Java T Point, n.d.)

*PowerShell Script Concepts*

**Section E:** Variables

## Brief Explanation

Variables in Windows PowerShell are like variables found in other programming languages. They are used to store data which makes it easier for users to work with information in scripts or commands. Variables are essentially containers that store values that are data types like text, numbers, or arrays.

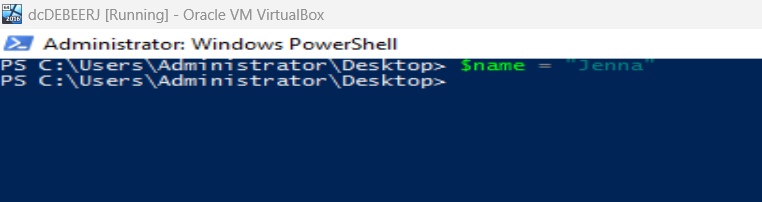
## Explanation of the Concept

Variables in PowerShell are defined with a ‘$’ followed by the name of the variable and assigned a value which will automatically be assigned a type. Variable names are not case sensitive and are assigned values with the ‘=’ operator. Variables should be declared before use to avoid errors and should be given clear, descriptive names to avoid confusion when used. It is generally best practice to include the data type in the name of the variable for users to refer to. For example, ‘$strMyVariable’ would specify to the user that the variable is a string.

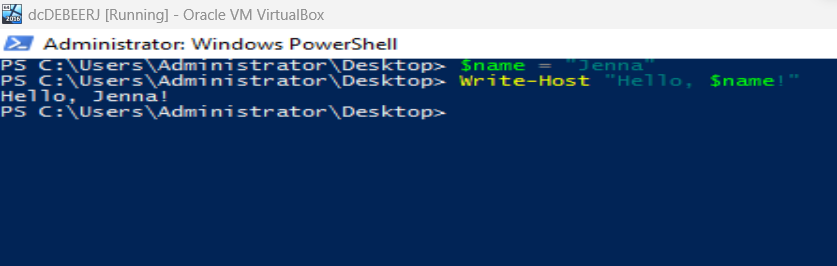
## How-To Guide

This how-to guide will explain how to define and assign a variable:

1. When creating a variable, the variable name should have a ‘$’ sign in front of it and a ‘=’ operator to assign a value. For example, you could create a variable that stores a name with ‘$name = “Jenna”’.



1. To display the value of a variable, you can use an output cmdlet like ‘Write-Host’ followed by the variable. In this example, the variable is displayed inside of a string.

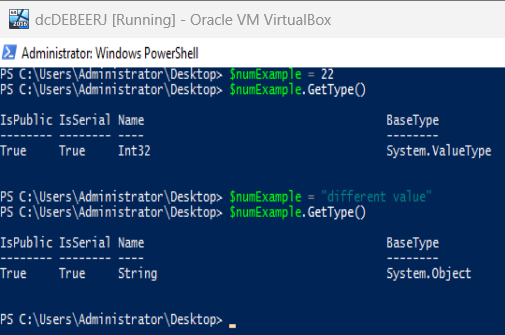


1. Windows PowerShell is different from many programming languages because it automatically assigns data types to variables, so you do not need to do it manually. You can check the data type of a variable with the ‘GetType()’ method.

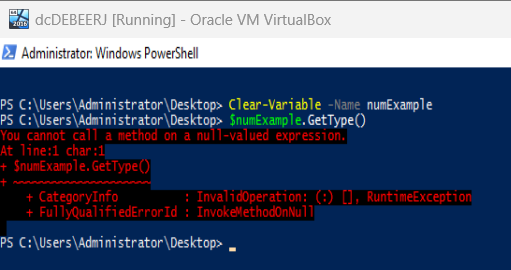
A screenshot of a computer

Description automatically generated

1. When reusing a variable name to declare a new value, it will overwrite the variable value that previously used the name. It is important to use clear and descriptive names to avoid overwriting other variables by mistake.



1. Variables can be cleared using the ‘Clear-Variable’ cmdlet followed by the name of the variable. In the example below, you can see that the variable was cleared because when the ‘GetType()’ method was called, an error was displayed explaining that the variable has a value of null.



(Microsoft, 2022)

# **Section F:** Arrays

## Brief Explanation

Windows PowerShell, like many other programming languages, uses arrays to store collections of data in the form of numbers, strings, or objects.

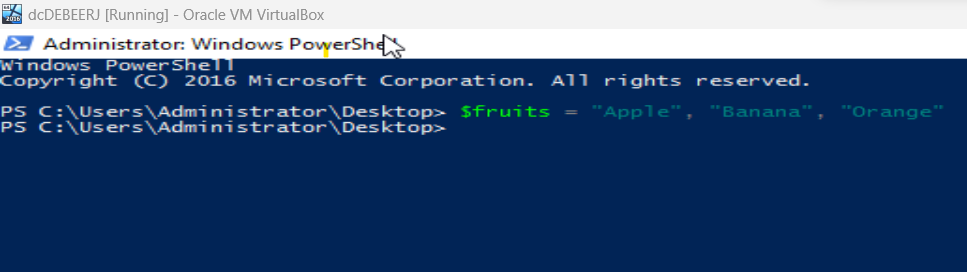
## Explanation of the Concept

An array is a data structure designed to store collections of data. Data stored in arrays can be the same or different data types, like numbers or strings. Using arrays allows users to work with multiple values at once as a single unit which is important when working with lists or other sets of data.

## How-To Guide

In this how-to guide, we will go over how to create and use arrays in PowerShell:

1. When creating an array in PowerShell, it is crucial to use descriptive names to enhance readability. The name of the array should be preceded by the ‘$’ symbol. For example, an array created to store the names of different fruits could be initialised as ‘$fruits’.
2. In the same line as the array name, set it equal to the data inside it with the ‘=’ operator. For example, ‘$fruits = “Apple”, “Banana”, “Orange”’. Alternatively, you can use the ‘@()’ operator. This would look like: ‘$fruits = @(“Apple”, “Banana”, “Orange”)’.

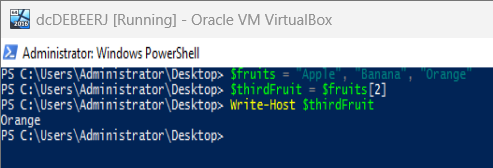


1. When initialising an array, it is important to assign it a value even if it is empty to avoid unexpected behaviour. To do this, you can use the ‘@()’ operator like this: ‘$fruits = @()’.

A screenshot of a computer

Description automatically generated

1. You can access individual elements of an array and declare them as variables using the ‘[]’. For example, if you want to access the third element in the ‘fruits’ array, you will execute ‘$thirdFruit = $fruits[2]’.



1. You could also change the elements in the array by assigning the indices new values. If you wanted to change the first element in the ‘fruits’ array for instance, you would execute ‘$fruits[0] = “Peach”’. Now the original value of ‘Apple’ has been replaced.

A screenshot of a computer program

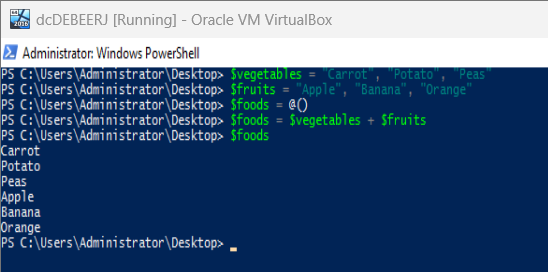
Description automatically generated

1. Elements can be added to the array with the ‘+=’ operator. For example, ‘$fruits += “Watermelon”’ adds the ‘Watermelon’ value into a new index. This operator should only be used for small arrays because it recreates the array each time it is used rather than simply adding to it.

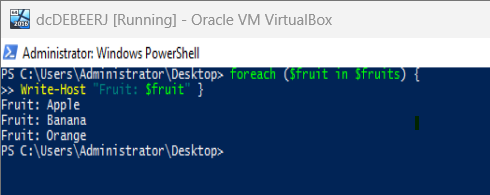
A screenshot of a computer program

Description automatically generated

1. You could alternatively merge the values of two arrays with the following syntax:



1. To iterate through the elements in an array, you could use a ‘foreach’ loop. For example, ‘foreach ($fruit in $fruits) { Write-Host “Fruit: $fruit” }’.



(Everything you wanted to know about arrays, 2022) (PowerShell add to array, 2023)

# **Section G**: Hashes/ Hash Tables

## Brief Explanation

In Windows PowerShell hashes, also known as hash tables or dictionaries, are data structures that store key-value pairs. This means that you can organise or fetch data by associating keys with corresponding values.

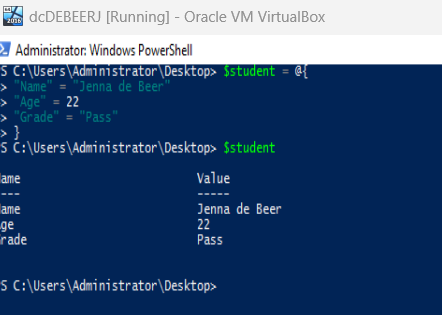
Explanation of the Concept

Hashes are collections of data which each have a unique key-value pair. The keys map to the value of the hash. Hashes can be used for storing, retrieving, or organising data in a way that is structured even though the data is not stored in any order.

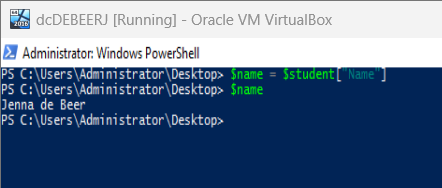
## How-To Guide

This how-to guide explains how to use hashes:

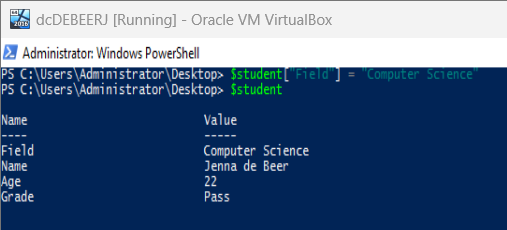
1. When you create a hash you will use the ‘@{}’ syntax and specify the hash’s key pairs.



1. Values in hashes can be accessed using keys with the ‘[]’. For example, if you wanted to find the ‘Name’ value in the ‘$student’ array, you will execute ‘$name = $student[“Name”]’ to declare the value to the ‘$name’ item which can be displayed when called.



1. The key-value pairs in the hash can be modified or a new key-value pair could be added with the following syntax: ‘$student[“Field”] = “Computer Science”’.



(How-to: Use Hash Tables in PowerShell, n.d.)

# Definition of Terms

* **Scripting and command-line tool**: A scripting capability in a programming language enables users to write programs that can be run on the system from its command line.
* **Automating administrative tasks**: Use of technology like scripts to simplify or streamline administrative tasks.
* **System configurations**: System configurations, or SC, are the makeup of IT infrastructure. It consists of the process of configuring software, hardware, and network settings in a computer system or network of computers.
* **Default working directory**: The working directory in PowerShell refers to the file path on the computer. The default directory is the location the computer sets for files that have an unspecified path.
* **Command**: Commands in PowerShell, often referred to as cmdlets (command-lets), are the smallest unit of the scripting language that can be invoked to perform actions or to return objects to a pipeline.
* **Pipeline**: Pipelines in PowerShell are used like real pipelines to connect segments of the script. The pipe operator, ‘|’, uses the output of a command as the input of the next command. (Script Runner, n.d.)
* **Script editor**: A script editor can be used to write and edit code files but do not have the building aspects of an integrated development environment (IDE).
* **Readability**: Readability refers to the ease with which code is read and understood by programmers who create a script or edit an existing one.
* **Initialise**: To initialise a value is to assign it a variable name of a particular data type.
* **Execute**: Execution of a line of code or program means to run a line of code or program.

# References

Doctor Scripto. (2014, June 17). *A Better PowerShell Command-Line Edit*. Retrieved from Microsoft: https://devblogs.microsoft.com/scripting/a-better-powershell-command-line-edit/

*Everything you wanted to know about arrays*. (2022, November 17). Retrieved from Microsoft: https://learn.microsoft.com/en-us/powershell/scripting/learn/deep-dives/everything-about-arrays?view=powershell-7.3

*How-to: Use Hash Tables in PowerShell*. (n.d.). Retrieved from SS64: https://ss64.com/ps/syntax-hash-tables.html

Java T Point. (n.d.). *Windows PowerShell ISE*. Retrieved from Java T Point: https://www.javatpoint.com/windows-powershell-ise

Kundu, K. (2017, December 9). *How to Change Windows PowerShell Color Scheme on Windows 10*. Retrieved from Beebom: https://beebom.com/how-change-powershell-color-scheme-windows-10/

Microsoft. (2022, September 19). *about\_Variables*. Retrieved from Microsoft: https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about\_variables?view=powershell-7.3

*PowerShell add to array*. (2023, March 8). Retrieved from Educba: https://www.educba.com/powershell-add-to-array/

Rath, M. (2020, November 21). *Set and Change the PowerShell default Working Directory*. Retrieved from matrix post: https://blog.matrixpost.net/set-powershells-default-working-directory/

Script Runner. (n.d.). *PowerShell Command*. Retrieved from Script Runner: https://www.scriptrunner.com/lexicon/scripting/powershell-commands/#:~:text=for%20PowerShell%20Commands-,Definition,goes%20for%20their%20derived%20Microsoft%20.