**VARIABLES AND OPERATORS**

**Variables**

Let’s start by discussing how variables work. In computer programming, *variables* are used to store a piece of data. In PowerShell, variables can store the results of commands and expressions like values, names, paths, and settings.

**Create a Variable**

To create a variable, we must assign it a value. Variables in PowerShell are referenced using a dollar sign $ followed by a variable name. After a variable reference, we use an equal sign = followed by the value we’d like to assign to that variable.

$my\_string\_variable = "Hello, World!"

In the example above, we are initializing a variable called my\_string\_variable to a string value of Hello, World!.

Variables names consist of alphanumeric characters. They are NOT case-sensitive and can include spaces and special characters when enclosed in curly braces, {}. However, following the convention of using only alphanumeric characters and the underscore \_ character is recommended because variables with special characters in their name are difficult to use.

**Use a Variable**

PS > $my\_string\_variable  
Hello, World!

Variable reference allows us to use or manipulate variables. As shown above, referencing the variable my\_string\_variable we defined earlier in the PowerShell terminal prints the value we assigned to it.

**User Input**

Now that we can hold data using variables, we can look at a useful command that enables user input through the terminal.

PS > $my\_input = Read-Host -Prompt "Enter a number"

The above example will output the -Prompt string and then wait for the user to input a value and hit Enter. The value will then be stored in $my\_input for use later.

These a just a few ways we can use variables to collect, process, and output data within our terminal and scripts.