PowerShell

Programmierung mit PowerShell
Kontrollfluss

Vorbereitung

Erstellen Sie für jede Aufgabe dieser Aufgabenserie ein eigenes PowerShell-Skript mit der Endung ps1. Führen Sie die Skripte aus, indem Sie mit einer PowerShell (ggf. als Administrator) in das Verzeichnis navigieren, in dem das Skript liegt.

Aufgabe 1

a. Legen Sie eine Variable *$geld* an und weisen Sie ihr den Wert 89 zu.

b. Teilen Sie den Wert mit einem weiteren Befehl durch 3.

c. Legen Sie eine Konstante *$const_euro* mit dem Wert *Euro in meinem Portemonnaie* an.

d. Lassen Sie den Satz *Ich habe x Euro in meinem Portemonnaie.* ausgeben. Statt ‚x' soll dabei der Inhalt der Variablen *$geld* verwendet werden.

Aufgabe 2

a. Legen Sie eine Arrayvariable *$proc* an, die als Wert die Namen der aktuell auf Ihrem System laufenden Prozesse enthält.

b. Lassen Sie sich den 5. Eintrag des Arrays anzeigen.

Aufgabe 3

a. Prüfen Sie, ob *Julius* mit *julius* identisch ist. Verwenden Sie ein Mal die Unterscheidung von Groß- und Kleinschreibung und ein Mal nicht.

b. Prüfen Sie, ob $\frac{14}{3}$ größer oder gleich 4,6 ist.

c. Finden Sie heraus, ob im Wert der Variablen *$HOME* der Buchstabe ‚o' enthalten ist.

Aufgabe 4

a. Speichern Sie eine Zahl zwischen 1 und 10 in einer Variablen. Lassen Sie *Zahl ist größer als 5* ausgeben, wenn die Zahl größer als 5 ist.

b. Lassen Sie nun *Zahl ist 5* ausgeben, wenn die Zahl genau 5 ist.

c. Geben Sie in allen anderen Fälle *Zahl ist kleiner als 5* aus.

PowerShell

Programmierung mit PowerShell
Kontrollfluss

Aufgabe 5

Programmieren Sie eine Schleife, die die Werte 10 bis 100 in Zehnerschritten ausgibt.

Geben Sie jeweils mithilfe einer For- oder einer ForEach-Schleife die positiven ganzen Vielfachen der Zahl 5 bis zum Wert 100 aus.

Aufgabe 6

a. Schreiben Sie ein Skript, das einen kontinuierlichen Ping (wie unter GNU/Linux) durchführt. Die Wartezeit zwischen zwei Pings soll dabei 1 Sekunde betragen.

b. Entwickeln Sie Pseudo-Code, der beschreibt, wie eine Pipeline mit einer while-Schleife aus der bash in PowerShell umgesetzt werden kann. Achten Sie darauf, die Syntax möglichst wenig zu verändern.

Aufgabe 7

Schreiben Sie ein Skript, das die Maschinengenauigkeit anhand einer Folge von Zweierpotenzen bestimmt. Geben Sie dann den kleinsten Exponenten an, der ein von null verschiedenes Ergebnis liefert.

Aufgabe 8

Gegeben ist der folgende PowerShell Code:

```
gci | % { mi $_ .. }; $d=$(gl); sl ..; ri $d
```

a. Beschreiben Sie die Funktionalität des Skripts so genau wie möglich.

b. Schreiben Sie den Code als Skript, wobei Sie die korrekten Langformen aller verwendeten Cmdlets und Parameter verwenden.

Aufgabe 9

Schreiben Sie Ihre eigene Version des Parameters *-Include* für das Cmdlet *Get-ChildItem*. Lesen Sie dazu mit *Get-ChildItem* die Objekte im aktuellen Verzeichnis aus, iterieren Sie geeignet darüber und geben Sie nur die Namen mit der passenden Endung aus. Die Endung kann dabei zunächst eine Variable im Skript sein. Erweitern Sie Ihr Programm dann dahingehend, dass ein Parameter von der PowerShell übernommen wird, der die zu filternde Endung darstellt.