## **Control Structures, Functions and Standards Workshop**

**Work through the below tasks to practice what has been covered in today’s workshop. Rather than focusing on getting the correct answer, it’s the process of getting to the answer that’s important; ask for help but do not copy others, it’s more important to think about these tasks then getting the answers right.**

**Task 1**

Write code that will print ten asterisks (\*) like the following:



Notes:

write-host "\*\*\*\*\*\*\*\*\*\*"

**Task 2**

Write code that will print the following:



Notes:

Write-Host "\*\*\*\*\*\*\*\*\*\*"

Write-Host "\*\*\*\*\*"

Write-Host "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

**Task 3**

Use two for loops, one of them nested inside the other, to print the following 10x10 rectangle:



*The cmdlet ‘Write-Host “Hello” –NoNewLine’ will write to the same line, use this in one loop to write to the columns but not in the other to dictate your rows*

Notes:

$num1 = 1,2,3,4,5,6,7,8,9,10

$num2 = 0,1,2,3,4,5,6,7,8,9

foreach ($Nummber in $num1) {

foreach ($Number2 in $num2) {

Write-Host "\*" -NoNewline

}

Write-Host ""

}

**Task 4**

Use two for loops, one of them nested, to print the following 5x10 rectangle:



Notes:

$num1 = 1,2,3,4,5,6,7,8,9,10

$num2 = 1,2,3,4,5

foreach ($Number in $num1) {

foreach ($Number2 in $num2) {

Write-Host "\*" -NoNewline

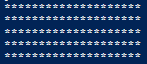
}

Write-Host ""

}

**Task 5**

Use two for loops, one of them nested, to print the following 20x5 rectangle:



Notes:

$num1 = 1,2,3,4,5

$num2 = 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20

foreach ($Number in $num1) {

foreach ($Number2 in $num2) {

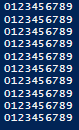
Write-Host "\*" -NoNewline

}

Write-Host ""

**Task 6**

Write code that will print the following:



Notes:

$num1 = 1,2,3,4,5,6,7,8,9,10

$Num2 = 0,1,2,3,4,5,6,7,8,9

foreach ($number in $Num1) {

foreach ($Number2 in $Num2) {

Write-Host $Number2 -NoNewline

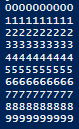
}

Write-Host ""

}

**Task 7**

Adjust the code in task 6 to print:



Notes:

$num1 = 0,1,2,3,4,5,6,7,8,9

$Num2 = 0,1,2,3,4,5,6,7,8,9

foreach ($number in $Num1) {

foreach ($Number2 in $Num2) {

Write-Host $Number -NoNewline

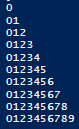
}

Write-Host ""

}

**Task 8**

Write code that will print the following:



Notes:

$Rows = 11

$Columns = 0,1,2,3,4,5,6,7,8,9

$Counter = 1

while ($Counter -lt $Rows) {

$Counter2 = 0

while ($Counter2 -lt $Counter) {

Write-Host $Columns[$Counter2] -NoNewline

$Counter2 += 1

}

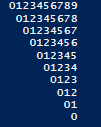
Write-Host ""

$Counter += 1

}

**Task 9**

Write code that will print the following:



Notes:

$Columns = 0,1,2,3,4,5,6,7,8,9

$Rows = 10

$ColumnCount = $Columns.Count

$Counter = 0

while ($Counter -lt $Rows) {

$Counter2 = 0

$Counter3 = 0

$ColumnsThisRow = $ColumnCount - $Counter

while ($Counter3 -lt $Counter) {

write-host " " -NoNewline

$Counter3 += 1

}

$Counter3 = 0

while ($Counter2 -lt $ColumnsThisRow) {

Write-Host $Columns[$Counter2] -NoNewline

$Counter2 += 1

}

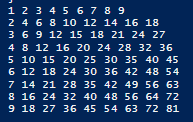
Write-Host ""

$Counter += 1

}

**Task 10**

Write code that will print the following (Getting the alignment is hard, at least get the numbers):



Notes:

$Columns = 1,2,3,4,5,6,7,8,9

$Rows = 9

$Counter = 1

while ($Counter -le $Rows) {

foreach ($Num in $Columns) {

$NewNum = $Num \* $Counter

Write-Host $NewNum -NoNewline

if ($NewNum -gt 10) {

Write-Host " " -NoNewline

} else {

Write-Host " " -NoNewline

}

}

Write-Host " "

$Counter += 1

}

**Task 11**

Write a function that will:

Have a string parameter

Lookup a service using that string parameter

Write to the console the current status and startup type of that service

If the function is unable to find the service return the string “ERROR – FLUXCAPACITOR OVERLOAD” with a red background and white text

Notes:

function CheckServices ([string]$ServiceName) {

    $LookupService = Get-Service | Where {$\_.Name -like "\*$ServiceName\*"}

    $ServiceStatus = $LookupService.Status

    $ServiceStartup = $LookupService.StartType

    if ($LookupService -eq $null) {

        Write-Host "ERROR - FLUXCAPACITOR OVERLOAD" -BackgroundColor Red

        break

    }

    Write-Host $("Status: $ServiceStatus, Startup: $ServiceStartup")

}