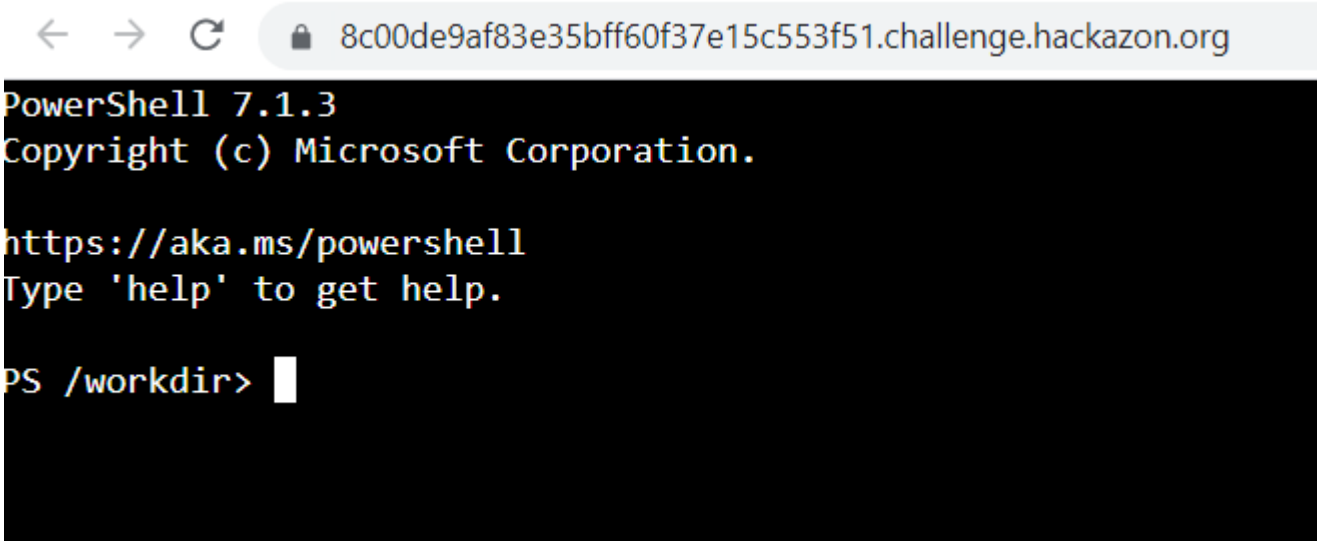
# Challenge Information

Are you the very best PowerShell user? Try this challenge to get better acquainted with PowerShell's functionality. You need to come up with commands that result in a specific output. You can check your output by piping the result to the "Check" function.
E.g. Get-Content words | dosomething | Check

Note: do not use any format- function before piping to the Check function. Also note that the checker may not understand all sorts of inputs. Try piping your output to the Out-String function first, or make sure your output matches more closely to the given example. Lastly, instead of write-host, use write-output to be able to pipe to the checker.

https://8c00de9af83e35bff60f37e15c553f51.challenge.hackazon.org/




## Flag 1:

### Description:

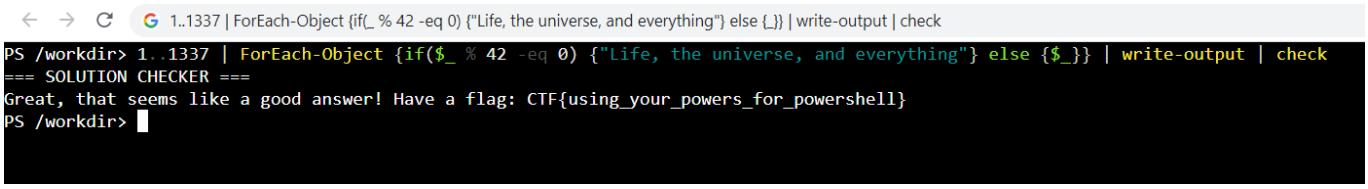
Write a script that writes out all numbers (1 per line) from 1 to 1337, inclusive. However, if the number is divisible by 42, instead, print the string "Life, the universe, and everything". Example excerpt given below:

..
40
41
Life, the universe, and everything
43
..

Command→

```
1..1337 | ForEach-Object {if($_ % 42 -eq 0) {"Life, the universe, and
everything"} else {$_}} | write-output | check
```

Output→




flag: CTF{using_your_powers_for_powershell}

## Flag 2:

### Description:

You're playing scrabble with your friends. You have the letters "iydhlao". Which are the words you can form? First sort them by increasing size, then alphabetically. Only include words of two letters and more. Make use of the dictionary file "dictionary" in /workdir. Example excerpt given below for different letters:

..
pad
pea
aped
deaf
..

Command→

```
    $dict = @(Get-Content ./dictionary | Sort-Object Length, { $_ })
    $word = 'iydhlao'
    $dict |
        Where-Object { $_.length -gt 1 } |
        ForEach-Object {
            $dictwordLetters =
[System.Collections.Generic.List[char]]::new($_.ToCharArray())
            $word.ToCharArray() | ForEach-Object {
```

```
                $dictwordLetters.Remove($_) | Out-Null
            }
            if (-not $dictwordLetters.Count) {
                $_
            }
        } | % { $_.Trim() } | check
```

Output→

8c00de9af83e35bff60f37e15c553f51.challenge.hackazon.org

```
PowerShell 7.1.3
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS /workdir> $dict = @(Get-Content ./dictionary | Sort-Object Length, { $_ })
PS /workdir> $word = 'iydhlao'
PS /workdir> $dict |
>>         Where-Object { $_.length -gt 1 } |
>>         ForEach-Object {
>>             $dictwordLetters = [System.Collections.Generic.List[char]]::new($_.ToCharArray())
>>             $word.ToCharArray() | ForEach-Object {
>>                 $dictwordLetters.Remove($_) | Out-Null
>>             }
>>             if (-not $dictwordLetters.Count) {
>>                 $_
>>             }
>>         } | % { $_.Trim() } | check
=== SOLUTION CHECKER ===
Great, that seems like a good answer! Have a flag: CTF{using_your_holidays_for_learning_powershell}
PS /workdir>
```

Flag: CTF{using_your_holidays_for_learning_powershell}

## Flag 3:

### Description:

In the tab-separated file "passwords.tsv" you get an overview of often-used passwords. Can you give us an overview of the number of passwords per category? Sort the result by descending count. Example excerpt given below:

Name Count
---- -----
.. ..
nerdy 30

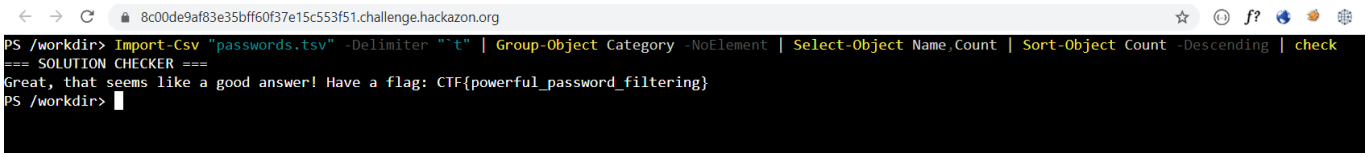
animal 19

.. ..

Command→

```
Import-Csv "passwords.tsv" -Delimiter "`t" | Group-Object Category -NoElement |
Select-Object Name,Count | Sort-Object Count -Descending | check
```

Output→




Flag: CTF{powerful_password_filtering}

# Flag 4:

## Description: Names

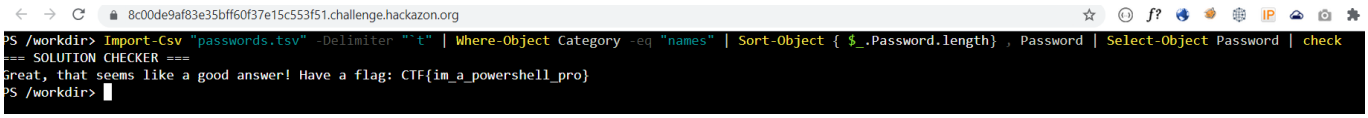
Given the passwords file, supply a list of passwords from the 'names' category ordered first by ascending password length, then alphabetically. Example excerpt given below:

..
scott
steve
albert
alexis
..

Command→

```
Import-Csv "passwords.tsv" -Delimiter "`t" | Where-Object Category -eq "names"
| Sort-Object { $_.Password.length} , Password | Select-Object Password | check
```

Output→




Flag:CTF{im_a_powershell_pro}