HACK YOUR EDUCATION 📝

WHAT IS SCRIPTING?

- Automation of processes at application level
- Short, simple programs that perform a specific task
- Usually runs in a shell which has a command-line interpreter (CLI)
    - Bourne shell (bash)
    - Command Prompt (cmd.exe)
    - Z shell (zsh)
    - PowerShell (pwsh)
- Can be written directly in the CLI or to files

USE CASES

- Automate a specific task
- Extract information from a dataset
- Build and deploy software

SCRIPTING LANGUAGES

- All scripting languages are programming languages
  - ...but not all programming languages are scripting languages
- Examples of scripting languages
  - Bash
  - Powershell
  - JavaScript
  - Python

POWERSHELL

- Cross platform task automation solution
  - Command-line shell
  - Scripting languages
  - Configuration management framework
- Accepts and returns .NET objects
- Aligns well with the technology stack we use at our department

POWERSHELL COMMANDS

- Powershell comes with hundreds of preinstalled commands
  - These are called *cmdlets* (pronounced as "command-lets")
- Some `cmdlets` examples
  - `Write-Host` —prints text to the console(docs)
  - `Write-Output` –writes specified objects to the pipeline(docs)
  - `Set-Location` –set the working directory to the specified location(docs)
  - `Get-ChildItem` –gets the items and child items in one or more specified location(docs)

HELLO WORLD

```
Write-Host "Hello 👋, Professor Kirk! 😏"
```

LIST FILES IN FOLDER

```
# Set current working directory
Set-Location -Path <YOUR_PATH>

# Print directory content to console
foreach($file in Get-ChildItem) {
  Write-Host $file
}

# Write file content to out_file
$out_file = New-Item -Path '../out/out_file.txt' -Force
foreach($file in Get-ChildItem) {
  $file | Out-File -FilePath $out_file -Append
}
```

GET ALL THE STUFF

HTTPS://GITHUB.COM/BVDA/HYE-SCRIPTING

ARGUMENTS

```powershell
param ($src_path, $dest_path)

New-Item -Path $dest_path `
  -ItemType Directory `
  -Force | Out-Null

Push-Location $src_path

foreach($file in Get-ChildItem) {
  Write-Host "Copying $file to $dest_path/$($file.Name)"
  Copy-Item $file -Destination $dest_path
}

Pop-Location
```

MERGE FILES

```
Set-Location -Path <YOUR_PATH>
$result_base = New-Object System.Collections.ArrayList
foreach($file in Get-ChildItem) {
  $json = Get-Content $file | ConvertFrom-Json
  foreach($person in $json) {
    $result_base.Add($person)
  }
}
$result_base | ConvertTo-Json | Out-File 'merged.json'
```

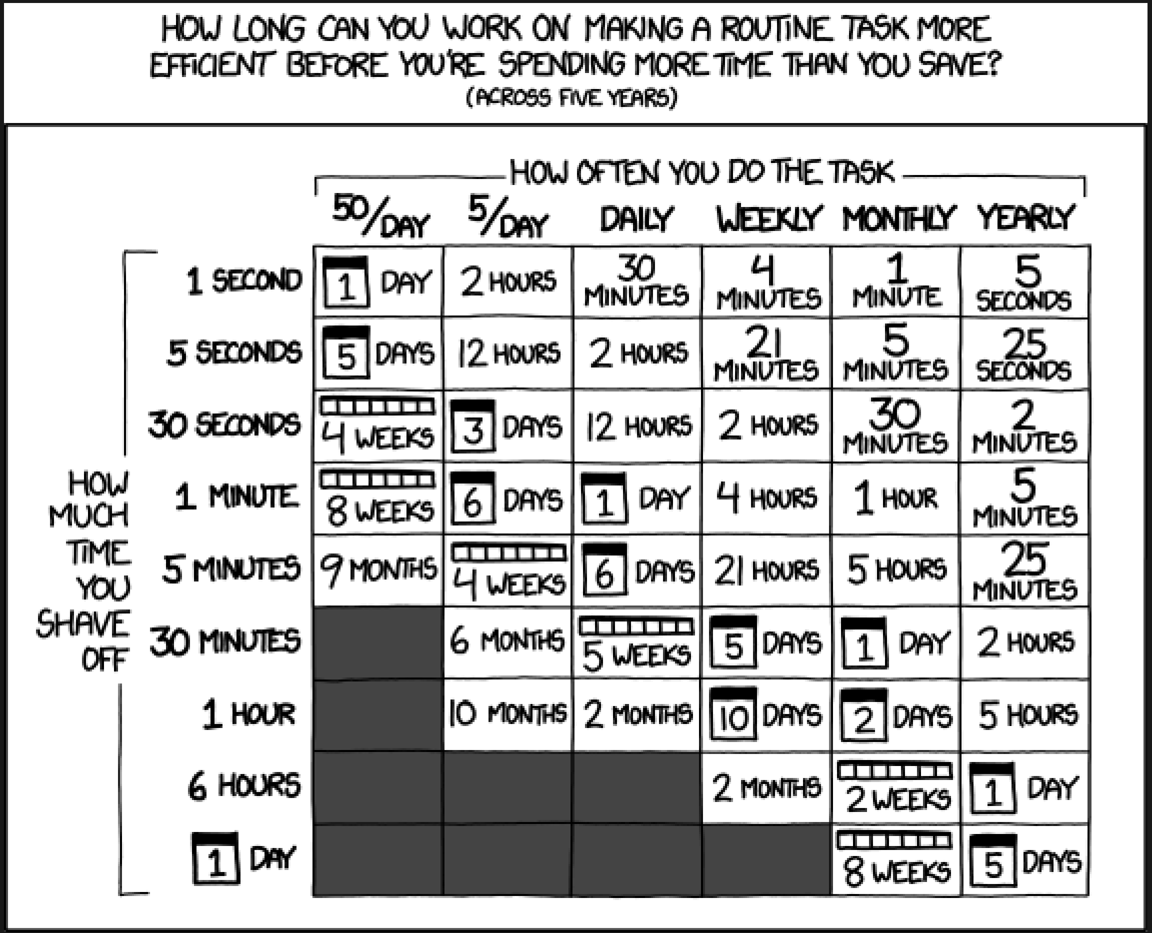
RUN EXECUTABLE

```
Set-Location <YOUR_PATH>
New-Item "./out" -ItemType Directory -Force

$fizzbuzz = (3,5), (5,7), (7,9), (11,13), (13, 15)
$start = 0
$end = 1000

foreach($t in $fizzbuzz) {
  $fizz = $t.Get(0)
  $buzz = $t.Get(1)
  Start-Process `
    -FilePath "./FizzBuzz_osx-x64" `
    -ArgumentList $start, $end, $fizz, $buzz `
    -RedirectStandardOutput "./out/$fizz-$buzz.out"
}
```

IS IT WORTH THE TIME?



HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE?

(ACROSS FIVE YEARS)

HOW MUCH TIME YOU SHAVE OFF	HOW OFTEN YOU DO THE TASK					
	50/DAY	5/DAY	DAILY	WEEKLY	MONTHLY	YEARLY
1 SECOND	1 DAY	2 HOURS	30 MINUTES	4 MINUTES	1 MINUTE	5 SECONDS
5 SECONDS	5 DAYS	12 HOURS	2 HOURS	21 MINUTES	5 MINUTES	25 SECONDS
30 SECONDS	4 WEEKS	3 DAYS	12 HOURS	2 HOURS	30 MINUTES	2 MINUTES
1 MINUTE	8 WEEKS	6 DAYS	1 DAY	4 HOURS	1 HOUR	5 MINUTES
5 MINUTES	9 MONTHS	4 WEEKS	6 DAYS	21 HOURS	5 HOURS	25 MINUTES
30 MINUTES		6 MONTHS	5 WEEKS	5 DAYS	1 DAY	2 HOURS
1 HOUR		10 MONTHS	2 MONTHS	10 DAYS	2 DAYS	5 HOURS
6 HOURS				2 MONTHS	2 WEEKS	1 DAY
1 DAY					8 WEEKS	5 DAYS

WRAP-UP

- Scripting can be applied to a lot of problems
- It can help prevent human errors
- Things to consider
  - How often am i going to do this task?
  - It is worth the time investment?
- Learn the basics
  - Pick the rest up as you go
  - Be careful not to fall into the rabbit hole!