理工学図書館
後期LS講習会

プログラミング超々入門
コマンドラインの基本

日時：12月15日（金）　16:30～17:20
場所：理工学図書館　東館1F
担当LS：南 卓海（電気電子M１）

アウトライン

- １： はじめに
  - コンピュータのしくみ
- ２： PowerShell で遊んでみよう！
  - 基本的な操作
  - ちょっぴり応用
  - シェルスクリプト
- ３： 最後に…

はじめに…

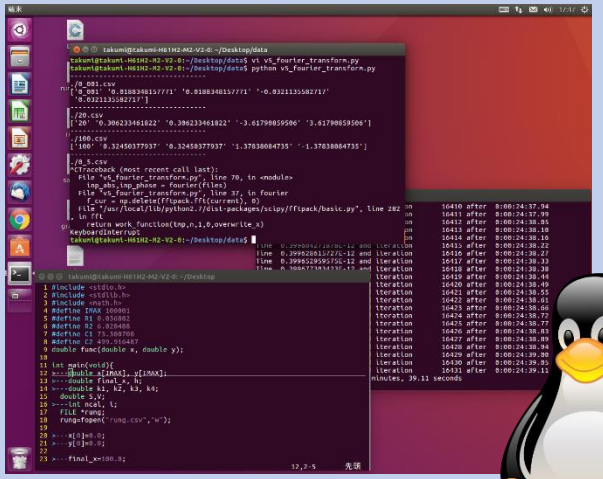
コンピュータの仕組み

OS

- “ハードウェア”と”ソフトウェア”をつなぐための”ソフトウェア”
- 人間とコンピュータの仲介役
- OSがなければコンピュータは動かない！

Windows

mac OS



Linux の一種、ubuntu

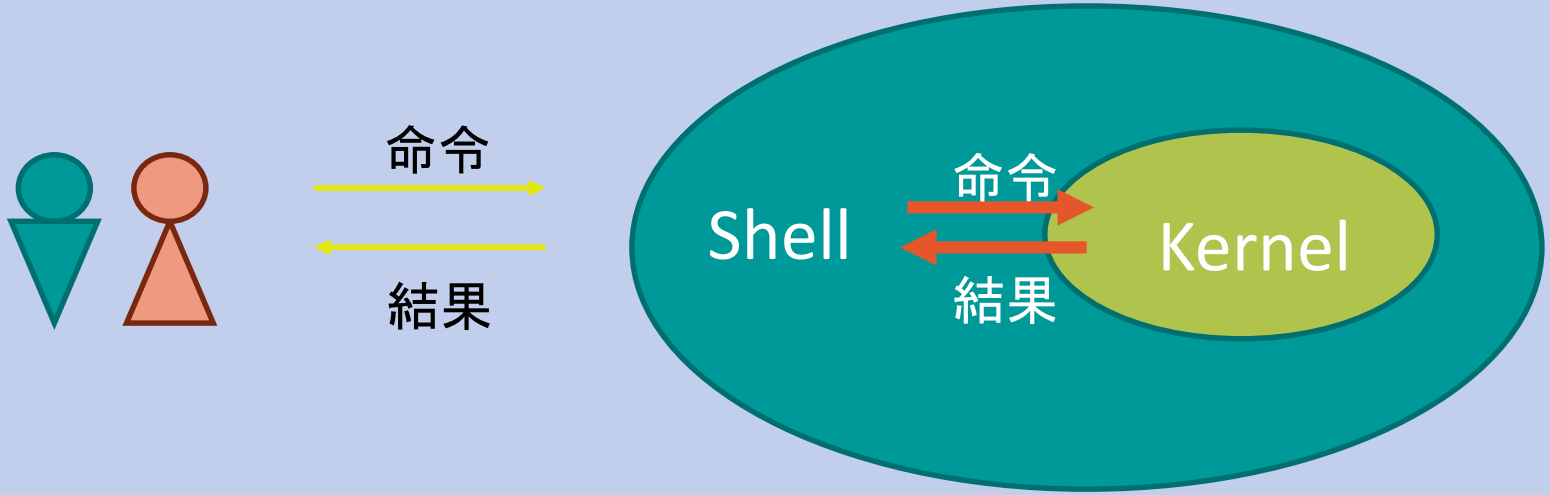
Kernel と Shell

Kernel

- パソコンのシステムの中核となるもの

Shell

- ユーザーがShellに対して指示を入力する
- Shellがカーネルに指示を伝える

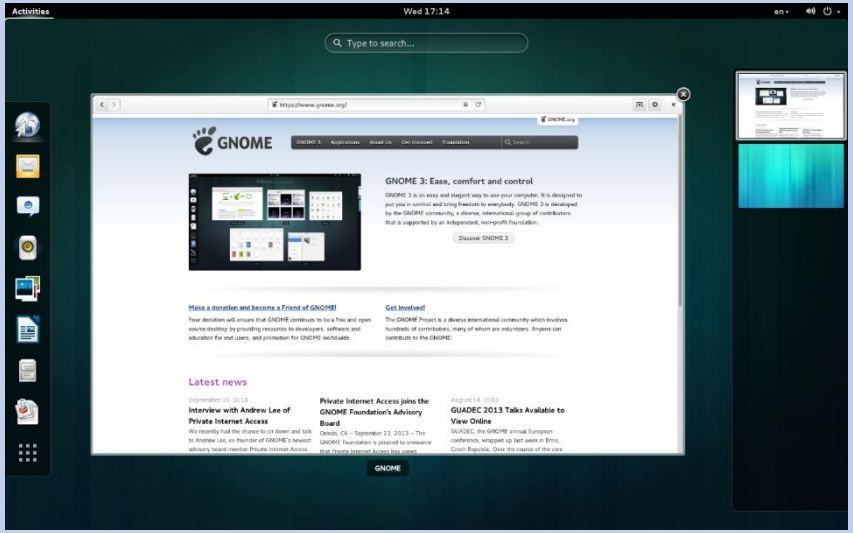


Shellはユーザーとコンピュータの仲介役！

GUI と CUI

GUI

- Graphical User Interface
- マウスでの操作がメイン



CUI

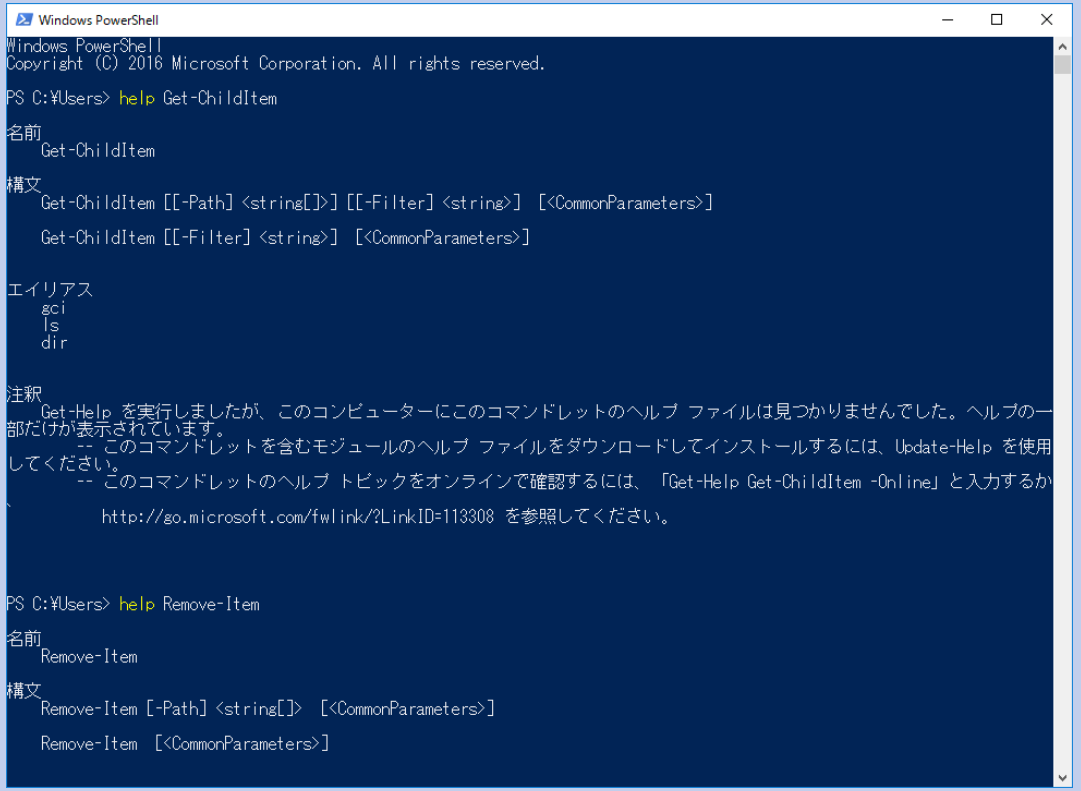
- Character User Interface
- キーボードでの操作がメイン
- Command Line Interfaceとも呼ぶ

```
mars@marsmain ~ $ pwd
/home/mars
mars@marsmain ~ $ cd /usr/portage/app-shells/bash
mars@marsmain /usr/portage/app-shells/bash $ ls -al
total 130
drwxr-xr-x  3 portage portage  1024 Jul 25 10:06 .
drwxr-xr-x 33 portage portage  1024 Aug  7 22:39 ..
-rw-r--r--  1 root    root    35808 Jul 25 10:06 ChangeLog
-rw-r--r--  1 root    root    27002 Jul 25 10:06 Manifest
-rw-r--r--  1 portage portage  4645 Mar 23 21:37 bash-3.1_p17.ebuild
-rw-r--r--  1 portage portage  5977 Mar 23 21:37 bash-3.2_p39.ebuild
-rw-r--r--  1 portage portage  6151 Apr  5 14:37 bash-3.2_p48-r1.ebuild
-rw-r--r--  1 portage portage  5988 Mar 23 21:37 bash-3.2_p48.ebuild
-rw-r--r--  1 portage portage  5643 Apr  5 14:37 bash-4.0_p10-r1.ebuild
-rw-r--r--  1 portage portage  6230 Apr  5 14:37 bash-4.0_p10.ebuild
-rw-r--r--  1 portage portage  5648 Apr 14 05:52 bash-4.0_p17-r1.ebuild
-rw-r--r--  1 portage portage  5532 Apr  8 10:21 bash-4.0_p17.ebuild
-rw-r--r--  1 portage portage  5560 May 30 03:35 bash-4.0_p24.ebuild
-rw-r--r--  1 root    root     5660 Jul 25 09:43 bash-4.0_p28.ebuild
drwxr-xr-x  2 portage portage  2048 May 30 03:35 files
-rw-r--r--  1 portage portage   468 Feb  9 04:35 metadata.xml
mars@marsmain /usr/portage/app-shells/bash $ cat metadata.xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE pkgmetadata SYSTEM "http://www.gentoo.org/dtd/metadata.dtd">
<pkgmetadata>
<herd>base-system</herd>
<use>
  <flag name='bashlogger'>Log ALL commands typed into bash; should ONLY be
    used in restricted environments such as honeypots</flag>
  <flag name='net'>Enable /dev/tcp/host/port redirection</flag>
  <flag name='plugins'>Add support for loading builtins at runtime via
    'enable'</flag>
</use>
</pkgmetadata>
mars@marsmain /usr/portage/app-shells/bash $ sudo /etc/init.d/bluetooth status
```

Windows PowerShellはCUIの仲間！

Windows PowerShell

- Windows 標準付属のコマンドライン
- 前身は「コマンドプロンプト」

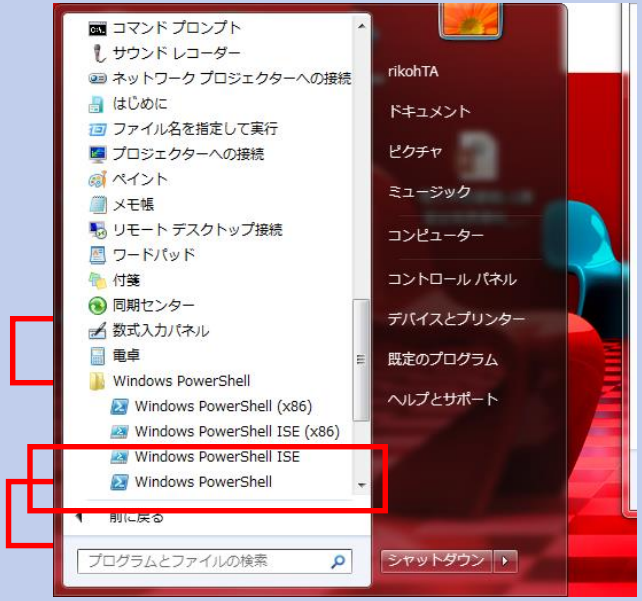


PowerShell で遊んでみよう！

起動方法

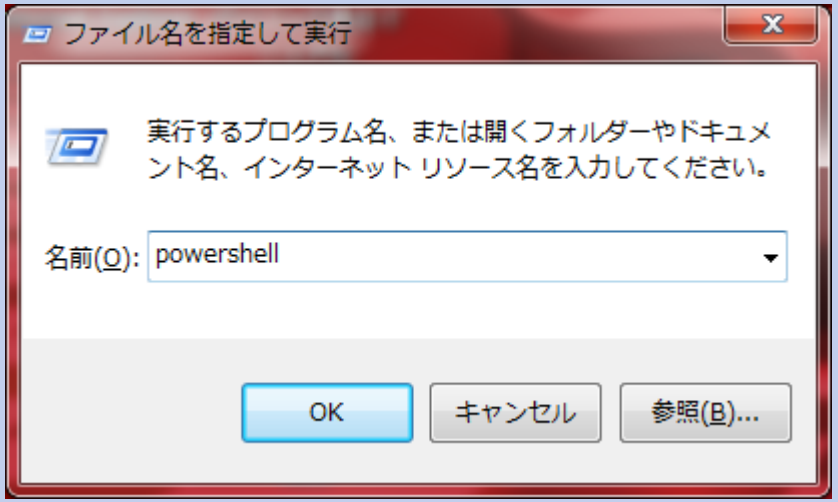
GUI

- 1：スタートボタンをクリック
- 2：プログラムの一覧の中から”Windows PowerShell”を探し、クリック



CUI

- 1：[Windows キー] + [R]
- 2：“powershell” と入力し、Enter



終了方法

GUI

- 右上、×印をクリックして終了

CUI

- “exit”と入力して Enter

カレントディレクトリ

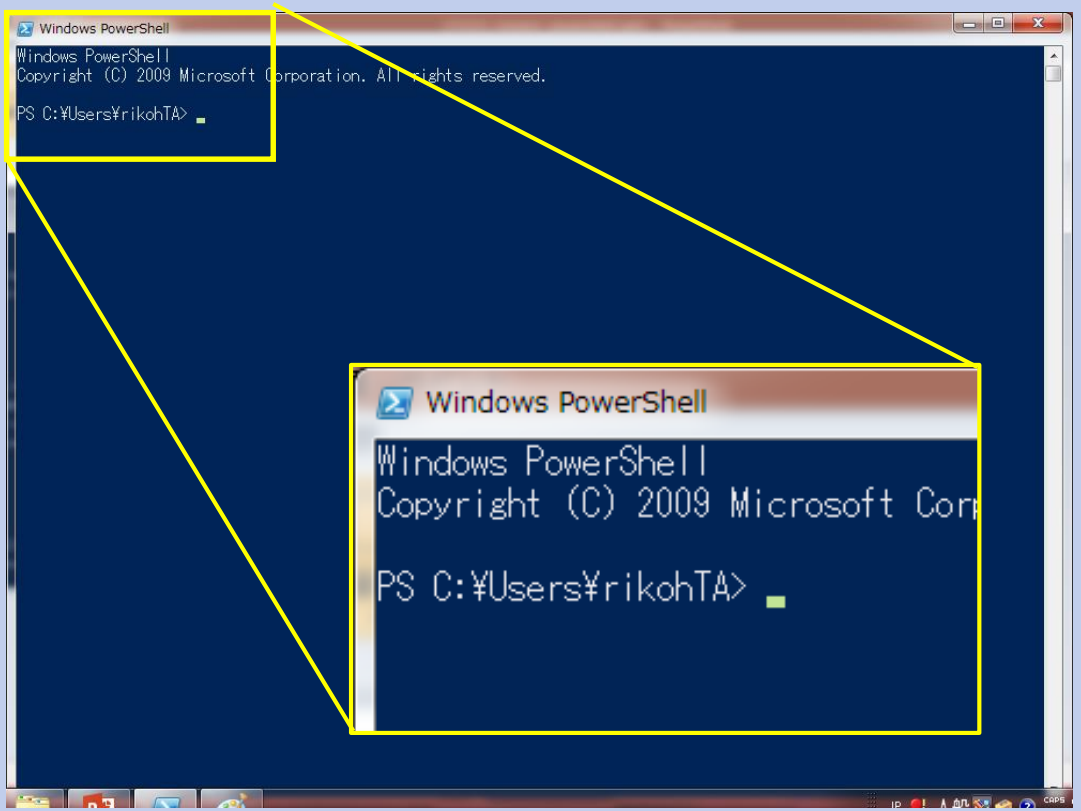
- コマンドライン上で、現在作業を行なっているフォルダ
- 右の例では、

  C:¥Users¥rikohTA
- PowerShell 上でファイル、フォルダを作成すると、カレントディレクトリに保存される。

```
PS C:¥Users¥rikohTA> mkdir sample


    ディレクトリ: C:¥Users¥rikohTA


Mode                 LastWriteTime     Length Name
----                 -------------     ------ ----
d----         12/13/2017   8:39 PM            sample
```



カレントディレクトリの移動

- カレントディレクトリを変更するには、”Set-Location” コマンドレットを用いる。

```
PS C:¥Users¥rikohTA> Set-Location sample
PS C:¥Users¥rikohTA¥sample>
```

カレントディレクトリが “sample” に変更されている。

```
PS C:¥Users¥rikohTA¥sample> Set-Location ..
PS C:¥Users¥rikohTA>
```

“..“ は親のディレクトリを表す

カレントディレクトリの内容を確認する

```
¥Desktop¥works> Get-ChildItem
```

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017  12:43 AM                data
-a----        12/15/2017   1:37 AM             32 aisatsu.ps1
-a----        12/15/2017   2:35 AM            286 replace_text.ps1
-a----        12/15/2017   2:40 AM             90 sample.txt
```

- Length の欄はファイルのサイズを表す。
- Length が空欄のものはディレクトリを表す。

ディレクトリを作成する

```
¥Desktop¥works> mkdir minami
```

Get-ChildItem :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017  12:43 AM                data
d-----        12/15/2017   3:07 AM                minami
-a----        12/15/2017   1:37 AM             32 aisatsu.ps1
-a----        12/15/2017   2:35 AM            286 replace_text.ps1
-a----        12/15/2017   2:40 AM             90 sample.txt
```

- ファイスサイズが 0 のディレクトリ “minami” が作成された

ファイル・ディレクトリの名前を変更する

```
¥Desktop¥works> Rename-Item minami "takumi"
```

Get-ChildItem :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017  12:43 AM                data
d-----        12/15/2017   3:23 AM                takumi
-a----        12/15/2017   1:37 AM             32 aisatsu.ps1
-a----        12/15/2017   2:35 AM            286 replace_text.ps1
-a----        12/15/2017   2:40 AM             90 sample.txt
```

- ディレクトリ “minami” の名前が “takumi” に変更された。

ファイル・ディレクトリを移動させる

```
¥Desktop¥works> Move-Item takumi data
```

:Get-ChildItem .¥data :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017   3:23 AM                takumi
-a----        12/14/2017   7:50 PM             47 001.txt
-a----        12/14/2017   7:50 PM             47 002.txt
-a----        12/14/2017   4:13 PM            698 003.txt
-a----        12/14/2017   5:15 PM            696 004.txt
```

- ディレクトリ“takumi”がディレクトリ“data”の中に移動された。
  - Get-ChildItem“ディレクトリ名” で指定したディレクトリの中を確認できる。
  - “ . ”はカレントディレクトリを表す。

```
¥Desktop¥works> Move-Item .¥data¥takumi¥ .¥minami
```

ファイル・ディレクトリを移動させる

```
¥Desktop¥works> Move-Item takumi data
```

Get-ChildItem .¥data :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017   3:23 AM                takumi
-a----        12/14/2017   7:50 PM             47 001.txt
-a----        12/14/2017   7:50 PM             47 002.txt
-a----        12/14/2017   4:13 PM            698 003.txt
-a----        12/14/2017   5:15 PM            696 004.txt
```

- ディレクトリ“takumi”がディレクトリ“data”の中に移動された。
  - Get-ChildItem“ディレクトリ名” で指定したディレクトリの中を確認できる。
  - “ . ”はカレントディレクトリを表す。

```
¥Desktop¥works> Move-Item .¥data¥takumi¥ .¥minami
```

ファイル・ディレクトリを削除する

- “work”ディレクトリの中

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017  12:43 AM                data
d-----        12/15/2017   3:07 AM                minami
-a----        12/15/2017   1:37 AM             32 aisatsu.ps1
-a----        12/15/2017   2:35 AM            286 replace_text.ps1
-a----        12/15/2017   2:40 AM             90 sample.txt
```

```
¥Desktop¥works> Remove-Item .¥minami
```

Get-ChildItem :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        12/15/2017  12:43 AM                data
-a----        12/15/2017   1:37 AM             32 aisatsu.ps1
-a----        12/15/2017   2:35 AM            286 replace_text.ps1
-a----        12/15/2017   2:40 AM             90 sample.txt
```

- ディレクトリ“minami”が削除されている。

ここまでのおさらい

コマンドレット	略コマンド	操作
Set-Location	cd	カレントディレクトリを移動する
Get-ChildItem	ls, dir	カレントディレクトリの内容を確認する
mkdir	md	ディレクトリを作製する
Rename-Item	ren	ファイル・ディレクトリの名前を変更する
Move-Item	mv, move	ファイル・ディレクトリを移動させる
Remove-Item	rm, del	ファイル・ディレクトリを削除する

略コマンドのことを”エイリアス”という！

```
¥Desktop¥works> alias
```

パイプ・リダイレクト

- パイプ “ | “
  - コマンドレットの結果を、次のコマンドレットの目的語に渡す役割をする。

```
¥Desktop¥works> alias | more
```

- リダイレクト “ > “
  - コマンドレットの結果を、” > “の右に示されたファイルに入力する。

```
¥Desktop¥works> alias > alias.txt
```

```
¥Desktop¥works> start notepad alias.txt
```

```
¥Desktop¥works> Get-Content alias.txt
```

Get-Content はファイルの内容を表示する！

ファイルの中から文字列を検索する

```
¥Desktop¥works> Select-String -path alias.txt -pattern "Remove-Item"

alias.txt:28:Alias          del -> Remove-Item
alias.txt:37:Alias          erase -> Remove-Item
alias.txt:114:Alias          rd -> Remove-Item
alias.txt:117:Alias          ri -> Remove-Item
alias.txt:119:Alias          rm -> Remove-Item
alias.txt:120:Alias          rmdir -> Remove-Item
alias.txt:124:Alias          rp -> Remove-ItemProperty
```

- Select-String -path “ファイルの名前” -pattern “検索する文字列”
- ファイルの名前は相対パスでも、絶対パスでも可能。

```
¥Desktop¥works> Select-String "data¥001.txt" -pattern "222"
```

```
¥Desktop¥works> Select-String "C:¥Users¥Tanaka¥Desktop¥temp¥001.csv" -pattern "222"
```

2つのファイルの中身を比較する

```
¥Desktop¥works¥data> Compare-Object .¥001.txt .¥002.txt

InputObject SideIndicator
----------- -------------
.¥002.txt   =>
.¥001.txt   <=
```

```
¥Desktop¥works¥data> Compare-Object (Get-Content .¥001.txt) (Get-Content .¥002.txt)

InputObject SideIndicator
----------- -------------
555         =>
888         =>
333         <=
777         <=
```

異なる部分だけが出力される。
右側のファイル(002.txt)で555の行が、
左側(001.txt)では333

```
¥Desktop¥works¥data> Compare-Object (Get-Content .¥003.txt) (Get-Content .¥004.txt)
```

フォルダ内のファイルの名前を“再帰的に”変更する

- フォルダ内にある全てのテキストファイル（.txt）を、カンマ区切り（.csv)に変更する
  - ex）テキストファイルで出力された実験データを Excel で解析できるようにする。

Get-ChildItem :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        12/14/2017   7:50 PM             47 001.txt
-a----        12/14/2017   7:50 PM             47 002.txt
-a----        12/14/2017   4:13 PM            698 003.txt
-a----        12/14/2017   5:15 PM            696 004.txt
```

```
¥Desktop¥works¥data> Get-ChildItem *.txt | Rename-Item -NewName {$_.Name -replace "txt","csv"}
```

Get-ChildItem :

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        12/14/2017   7:50 PM             47 001.csv
-a----        12/14/2017   7:50 PM             47 002.csv
-a----        12/14/2017   4:13 PM            698 003.csv
-a----        12/14/2017   5:15 PM            696 004.csv
```

解説

```
¥Desktop¥works¥data> Get-ChildItem *.txt | Rename-Item -NewName {$_.Name -replace "txt","csv"}
```

- Get-ChildItem “条件” --- 条件に合ったファイル・ディレクトリのみを表示する。
- * --- ワイルドカード(*には1文字以上の任意の文字が当てはまる)
- $_ --- パイプにより渡された結果(Get-ChildItem *.txt)を格納している変数

```
¥Desktop¥works¥data> $a = Get-ChildItem *.csv
¥Desktop¥works¥data> $a
```

- .Name --- Get-ChildItem *.txt により得られた結果から、ファイルの名前のみを抜き出す

```
¥Desktop¥works¥data> Get-ChildItem *.csv | %{$_.Length}
```

- -replace “txt”,”csv” --- $_.Name に対して”txt”という文字列を”csv”という文字列に置換する

ファイル内の文字列を置換する

- あるファイルに記されている特定の文字列すべてを、他の文字列に置き換える。

Get-Content sample.txt :

```
abcde - fghij - klmno
pqrst - uvwxy - z
```

```
¥Desktop¥works> $(Get-Content "sample.txt") -replace "abcde","12345" > sample_new.txt
```

Get-Content sample.txt :

```
12345 - fghij - klmno
pqrst - uvwxy - z
```

シェルスクリプト

- ある目的をもって様々なコマンドが書かれたファイル
  - 長いコマンドをいちいちタイプするのが面倒なとき…
  - 1つのコマンドでは対応できない複雑な処理をさせたいとき…
- PowerShell では “.ps1” というファイル
  - コマンドプロンプト “.bat” mac&linux “.sh”
- これだけでも色々なプログラムが書ける！

```
Get-Content aisatsu.ps1 : Write-Output "Hello, world!"
```

```
¥Desktop¥works> .¥aisatsu.ps1
```

.ps1を実行できない…

```
.¥aisatsu.ps1 : このシステムではスクリプトの実行が無効になっているため、ファイル
C:¥Users¥Tanaka¥Desktop¥works¥aisatsu.ps1 を読み込むことができません。詳細については、「about_Execution_Policies」(http
://go.microsoft.com/fwlink/?LinkID=135170) を参照してください。
発生場所 行:1 文字:1
+ .¥aisatsu.ps1
+
    + CategoryInfo          : セキュリティ エラー: (:) []、PSSecurityException
    + FullyQualifiedErrorId : UnauthorizedAccess
```

- デフォルトでは .ps1 を実行できないようにセキュリティがかかっている。
  - PowerShellは非常に強力なアプリケーションである
  - そのため、悪いプログラムを簡単に実行できないようにしている

→セキュリティを外せばよい。

セキュリティを外す

- 管理者権限でPowerShellを起動する

```
¥Desktop¥works> Start-Process powershell.exe -Verb runas
```

  - タスクバーのPowerShellアイコンを右クリック、「管理者として実行する」でもよい

- 以下のコマンドをタイプする

```
¥system32> Set-ExecutionPolicy RemoteSigned
```

- “Y” をタイプしてEnter

  - セキュリティを元に戻したければ、

```
¥system32> Set-ExecutionPolicy Restricted
```

.ps1 の実行

```
¥Desktop¥works> .¥aisatsu.ps1
```

- 文字列の置換のためのスクリプト　replace_text.ps1

```
¥Desktop¥works> .¥replace_text.ps1 sample.txt klmno 98765
```

- 色々なスクリプトを作ってみてください！

最後に…

- 実際にプログラミングを始めるなら
  - 統合開発環境 (IDE : Integrated Development Environment) を用いる
    - Python –> Anaconda,　C++ –> Visual Studio　etc.
  - Cygwin を用いる
  - Linux (もしくはmac OS) を用いる

- プログラムは「調べる」「書く」「実行する」「また調べる」… の繰り返し
- 根気がいる…‼

プログラミング超々入門
コマンドラインの基本

日時：12月15日（金）　16:30～17:00
場所：理工学図書館　東館1F
担当LS：南 卓海（電気電子M１）

理工学図書館
後期LS講習会