Scripting

TOULOUSE
ynov
CAMPUS

Scripting?

Le scripting consiste à coder pour automatiser des processus fastidieux à faire manuellement

Objectif du cours

Scripting?

- Langage “haut niveau”

Scripting?

- Langage “haut niveau”
- Langage interprété

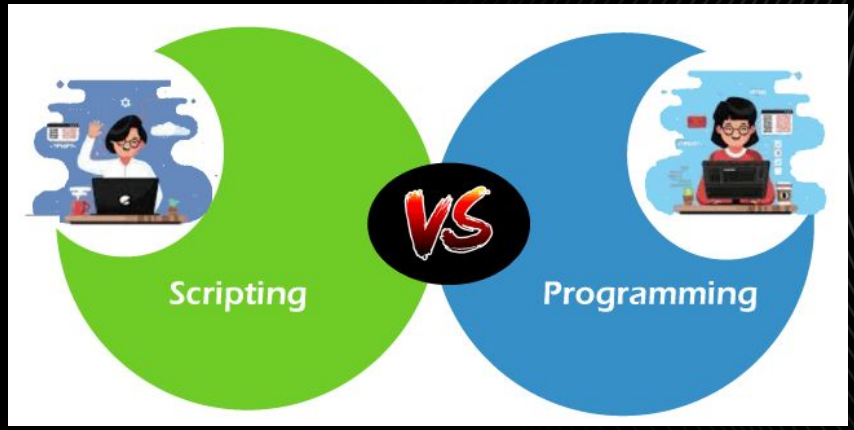
Scripting?

- Langage “haut niveau”
- Langage interprété
- Pas de compilation

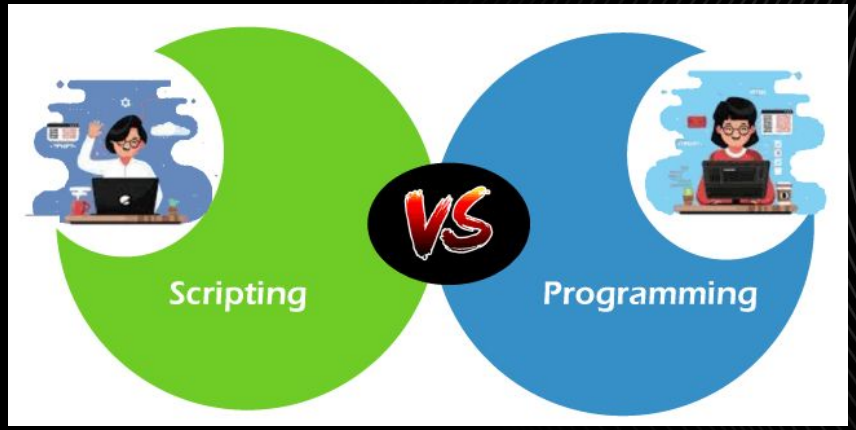
Scripting?

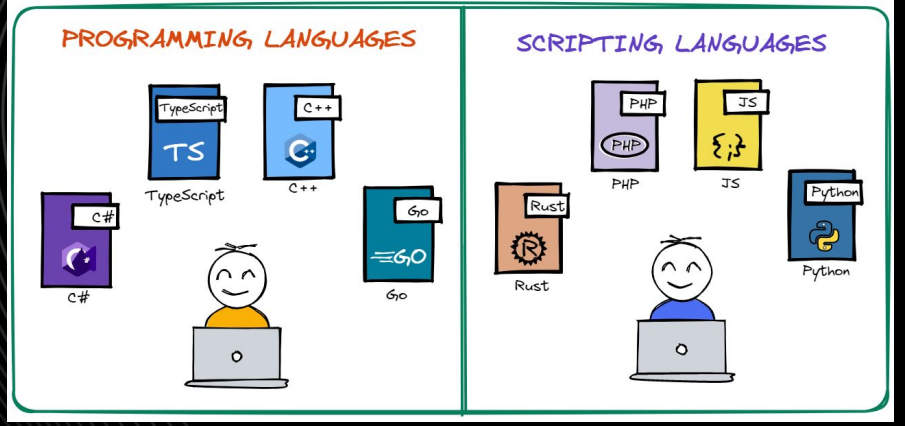
- Langage “haut niveau”
- Langage interprété
- Langage non typé
- Pas de compilation
- Utile pour combiner des composants déjà existants

Scripting?

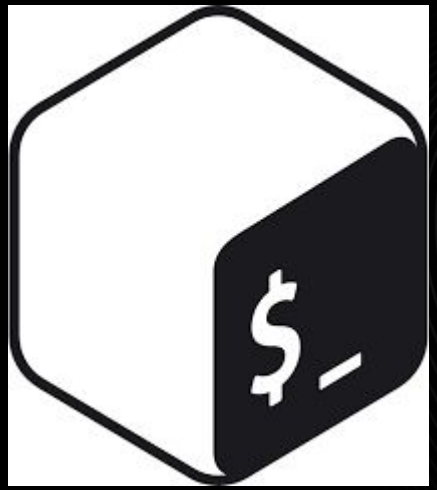


Scripting?


Scripting
VS
Programming

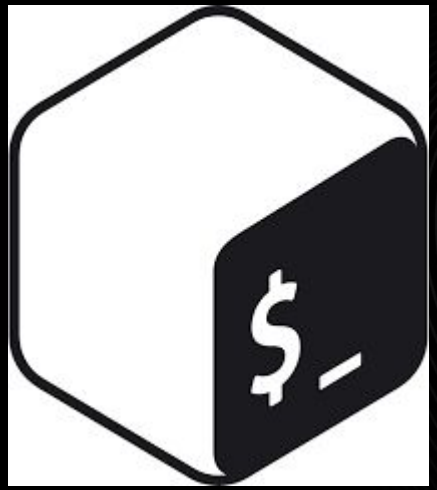

PROGRAMMING LANGUAGES
TypeScript
TS
TypeScript
C++
C++
C#
C#
Go
Go
SCRIPTING LANGUAGES
PHP
PHP
JS
JS
Rust
Rust
Python
Python

Bash?



Bourne again shell

Bash?



Bourne again shell

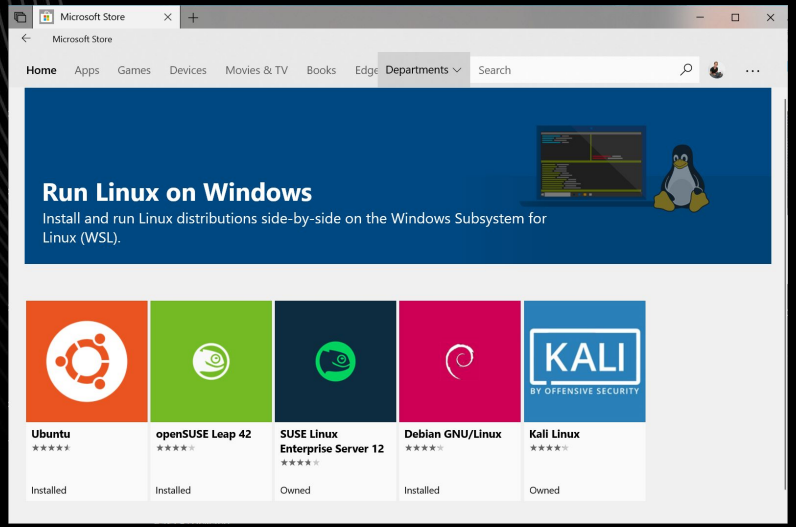
Terminal & langage

Pourquoi Bash?

- Accéder à l’OS de manière performante
- Automatiser des tâches
- Lancer n’importe quel langage de prog
- Utiliser n’importe quel outil (BDD, git, docker…)
- Popularité de bash (serveurs linux, support de bash, communauté)

Comment utiliser Bash?

WSL 2



Comment utiliser Bash?

Visual Studio Code

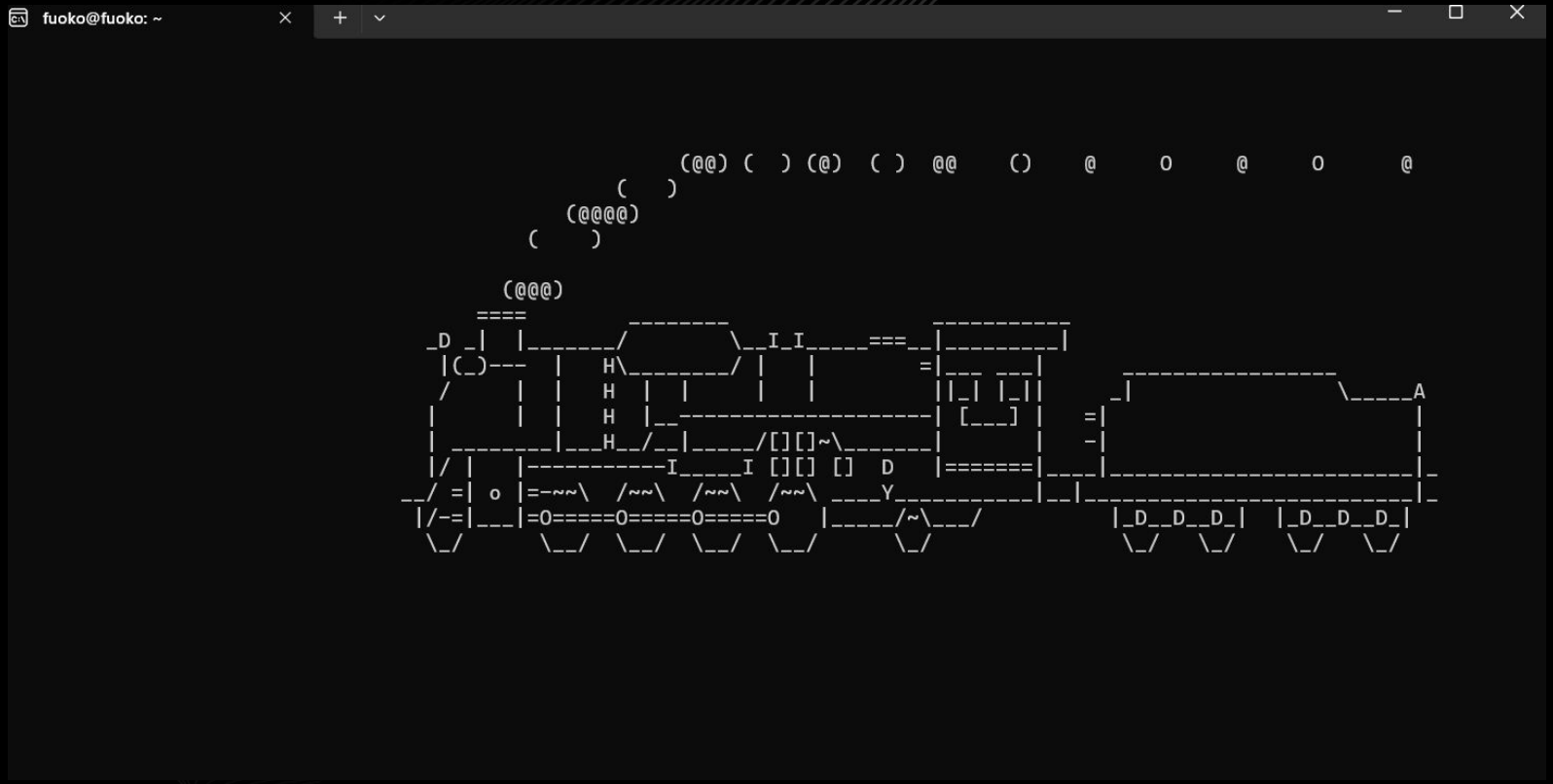
Maîtriser les commandes linux

TOULOUSE
ynov
CAMPUS

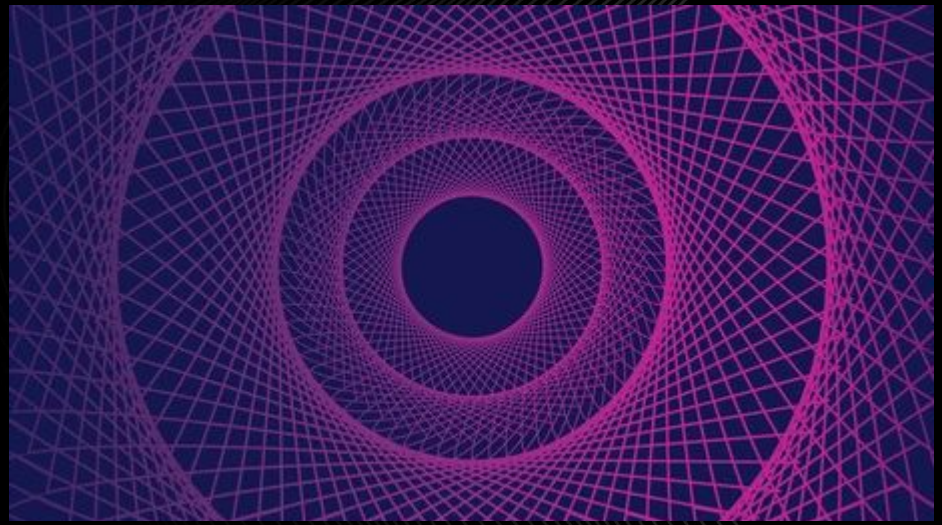
Step 1

```
                .-/+oossssoo+/-.               bajcmartinez@xps-linux
            `:+ssssssssssssssssss+:`           ----------------------
          -+ssssssssssssssssssyyssss+-         OS: Ubuntu 20.04.1 LTS x86_64
        .ossssssssssssssssssdMMMNysssso.       Host: XPS 15 9570
       /ssssssssssshdmmNNmmyNMMMMhssssss/      Kernel: 5.4.0-48-generic
      +ssssssssshmydMMMMMMMNddddyssssssss+     Uptime: 12 days, 21 hours, 48 mins
     /sssssssshNMMMyhhyyyyhmNMMMNhssssssss/    Packages: 2117 (dpkg), 13 (snap)
    .ssssssssdMMMNhsssssssssshNMMMdssssssss.   Shell: zsh 5.8
    +ssssshhhyNMMNyssssssssssyNMMMysssssss+    Resolution: 3840x2160
    ossyNMMMNyMMhsssssssssssshmmmhssssssso     DE: Plasma
    ossyNMMMNyMMhsssssssssssshmmmhssssssso     WM: KWin
    +ssssshhhyNMMNyssssssssssyNMMMysssssss+    Theme: Breeze [Plasma], Breeze [GTK2/3]
    .ssssssssdMMMNhsssssssssshNMMMdssssssss.   Icons: breeze [Plasma], breeze [GTK2/3]
     /sssssssshNMMMyhhyyyyhdNMMMNhssssssss/    Terminal: konsole
      +ssssssssshmydMMMMMMMNddddyssssssss+     CPU: Intel i7-8750H (12) @ 4.100GHz
       /ssssssssssshdmNNNNmyNMMMMhssssss/      GPU: Intel UHD Graphics 630
        .ossssssssssssssssssdMMMNysssso.       GPU: NVIDIA GeForce GTX 1050 Ti Mobile
          -+ssssssssssssssssssyyssss+-                                          3
            `:+ssssssssssssssssss+:`
                .-/+oossssoo+/-.
```

Rappels des commandes linux

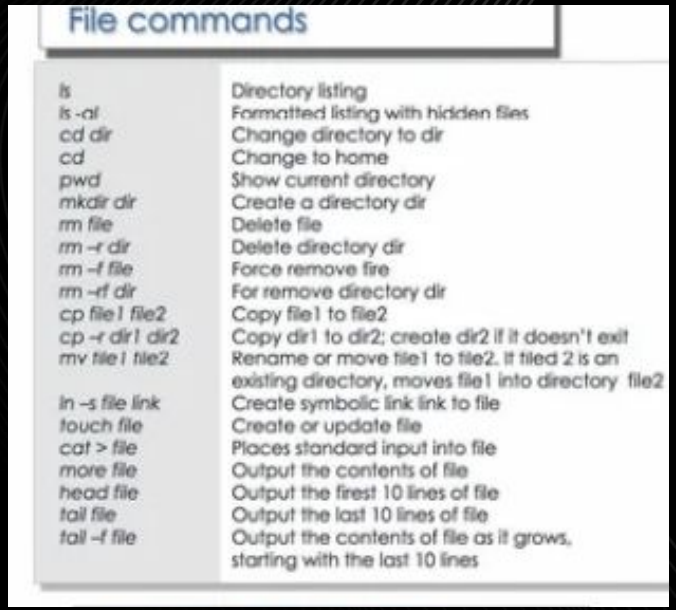


Rappels des commandes linux



echo “hello world!”

Rappels des commandes linux



File commands

ls	Directory listing
ls -al	Formatted listing with hidden files
cd dir	Change directory to dir
cd	Change to home
pwd	Show current directory
mkdir dir	Create a directory dir
rm file	Delete file
rm -r dir	Delete directory dir
rm -f file	Force remove fire
rm -rf dir	For remove directory dir
cp file1 file2	Copy file1 to file2
cp -r dir1 dir2	Copy dir1 to dir2; create dir2 if it doesn't exit
mv file1 file2	Rename or move file1 to file2. If tiled 2 is an existing directory, moves file1 into directory file2
ln -s file link	Create symbolic link link to file
touch file	Create or update file
cat > file	Places standard input into file
more file	Output the contents of file
head file	Output the firest 10 lines of file
tail file	Output the last 10 lines of file
tail -f file	Output the contents of file as it grows, starting with the last 10 lines

Exercice 1

1) Dans les documents, créer un répertoire exo_noob
2) Dans ce répertoire, créer un **répertoire** exo1 et un **fichier** exo1
3) Editer le fichier exo1 avec la commande nano et écrire comme contenu “ezpzlifeisez”
4) Afficher le contenu du fichier exo 1 à l’aide d’une commande
5) Renommer le répertoire exo1 en exo_facile avec une commande
6) Supprimer le répertoire exo_noob avec une commande

Les outils stylés

TOULOUSE
ynov
CAMPUS

Pipe |

Transfère l’output d’une commande à l’input de la suivante

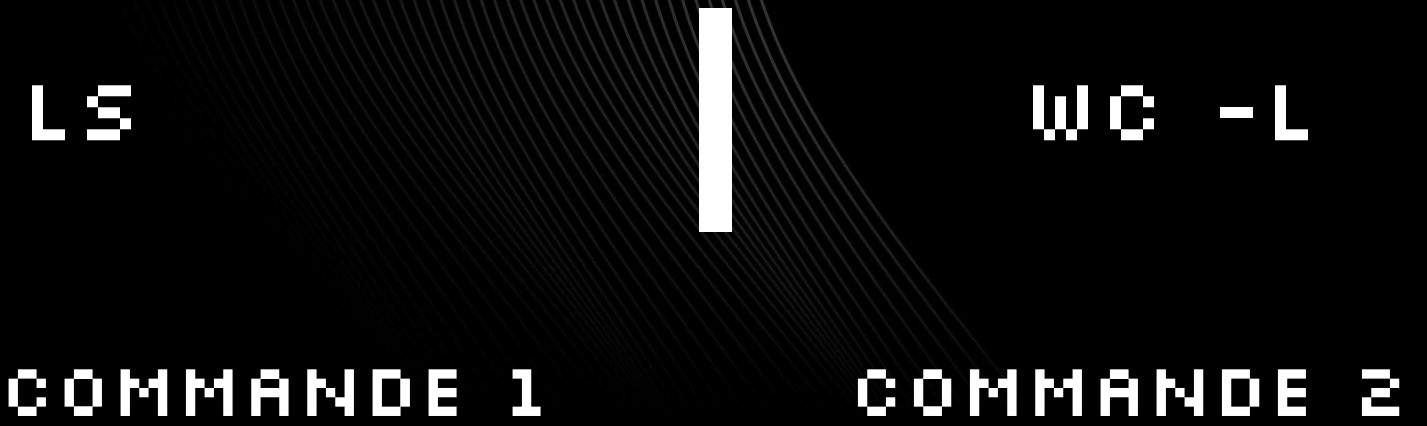
COMMANDE 1

|

COMMANDE 2

Pipe |

Transfère l’output d’une commande à l’input de la suivante



Pipe |

Transfère l’output d’une commande à l’input de la suivante

Redirection sorties > et >>


```
ls > list_files.txt
echo “lut” > sa
```

Opérateur wildcard

*

Opérateur wildcard

*

Remplacé par n’importe quelle chaine de caractère

Opérateur wildcard

*

```
cp * /other/path
rm *
mv *.txt /other/path
```

Opérateur wildcard

?

Opérateur wildcard

?

Remplacé par n’importe quel caractère unique

Opérateur wildcard

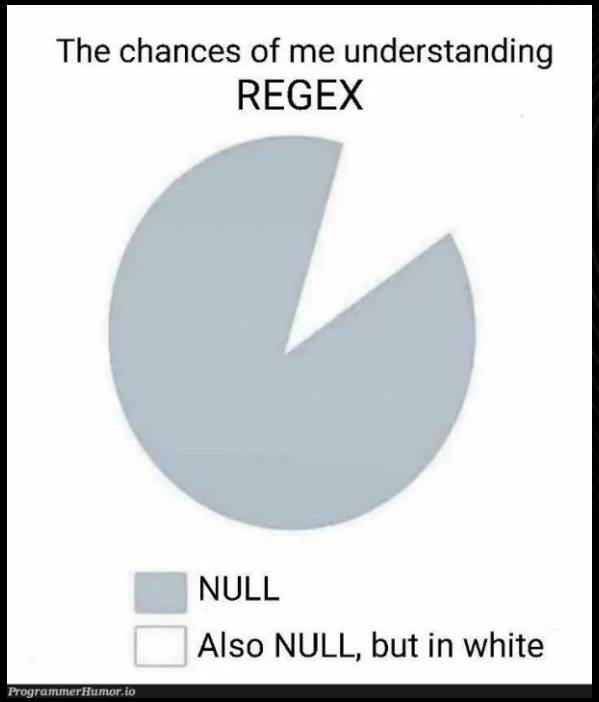
?

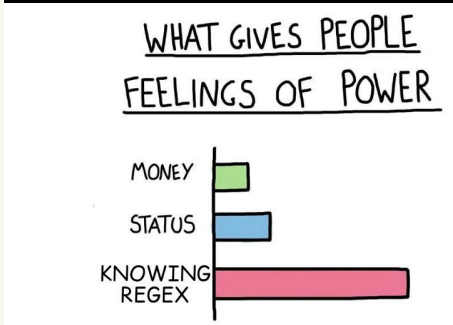
```
cp ??? /other/path
```

Exercice 2

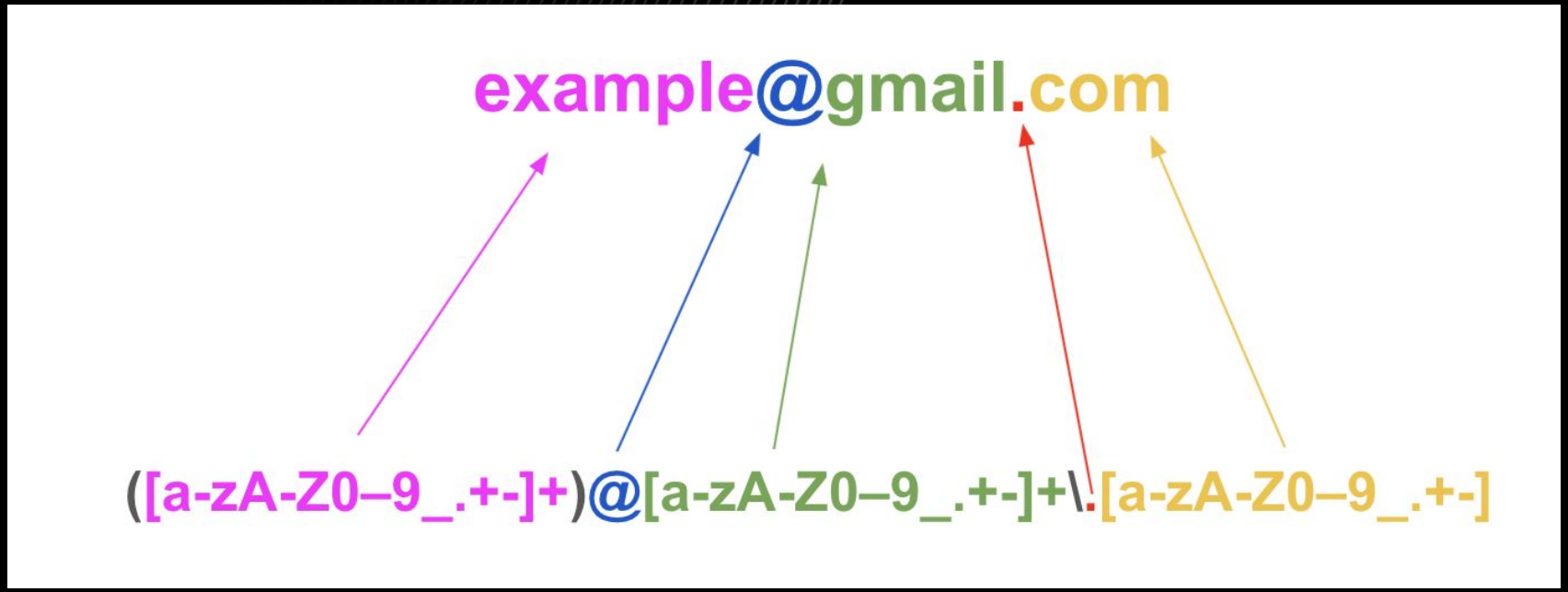
1) Dans dossier, créer 5 fichiers
   - one (contenu 1)
   - two (contenu 2)
   - three (contenu 3)
   - four (contenu 4)
   - five (contenu 5)
2) Afficher le contenu des fichiers qui ont un nom de trois lettres avec le wildcard ?
3) Afficher le contenu des fichiers qui commencent par la lettre t avec le wilcard *
4) Ecrire le contenu des cinq fichiers dans un nouveau fichier fusion.txt (redirection)
5) Ecrire une commande qui affiche le nombre de fichiers dans ce répertoire

Expressions régulières





Expressions régulières



Expressions régulières

Applications :

Extractions de données

Search & replace

Scraping

Versatile

...

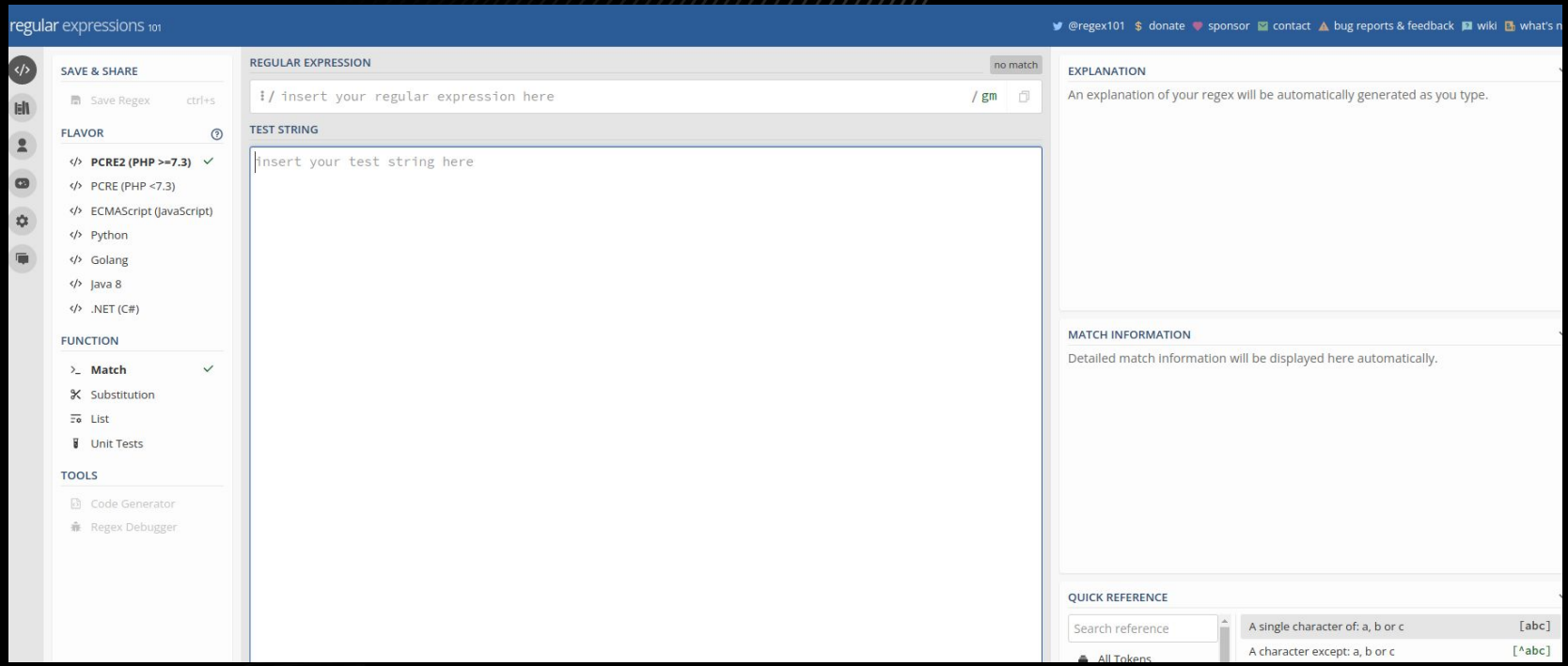
Découverte de grep et des expressions régulières

Character Classes	
`[abc]`	A single character of: a, b or c
`[^abc]`	A character except: a, b or c
`[a-z]`	A character in the range: a-z
`[^a-z]`	A character not in the range: a-z
`[0-9]`	A digit in the range: 0-9
`[a-zA-Z]`	A character in the range: a-z or A-Z
`[a-zA-Z0-9]`	A character in the range: a-z, A-Z or 0-9

Quantifiers	
`a?`	Zero or one of a
`a*`	Zero or more of a
`a+`	One or more of a
`[0-9]+`	One or more of 0-9
`a{3}`	Exactly 3 of a
`a{3,}`	3 or more of a
`a{3,6}`	Between 3 and 6 of a
`a*`	Greedy quantifier
`a*?`	Lazy quantifier
`a*+`	Possessive quantifier

Anchors	
`\G`	Start of match
`^`	Start of string
`$`	End of string
`\A`	Start of string
`\Z`	End of string
`\z`	Absolute end of string
`\b`	A word boundary
`\B`	Non-word boundary

Regex101.com, site pour tester ses expressions



Découverte de grep et des expressions régulières

```
grep bonjour bonjour.txt

grep -E <regex> bonjour.txt

grep -oE <regex> bonjour.txt
(-o garde seulement l’expression matchée)
```

Découverte de grep et des expressions régulières

MOT QUI COMMENCENT PAR COR

CORDE
CORSE
CORNE

CONTE
COMME
COLIBRI
ACCORD

Découverte de grep et des expressions régulières

MOTS QUI COMMENCENT PAR BA ET FINISSENT PAR E

BASSE
BACTERIE
BAIGNOIRE

BONJOUR
BILLE
EBAHI
BASSIN

Découverte de grep et des expressions régulières

MOTS DE CINQ LETTRES

BOULE
BONTE
ALLEZ

ALLO
INCROYABLE
MAISON

Découverte de grep et des expressions régulières

MOTS QUI CONTIENNENT QUE DES VOYELLES

EAU
OUI
OU

AILE
SOU

Découverte de grep et des expressions régulières

MOTS QUI ALTERNENT SUCCESSIVEMENT CONSONNES ET VOYELLES

PATATE
SOPALIN
MONTAGE

EAU
BONJOUR
COULEUR

Exercice Regex

1) Créer un répertoire exo2 dans le dossier personnel
2) Télécharger le zip du dico à cette adresse (wget)
   http://www.3zsoftware.com/listes/liste_francais.zip
3) Installer le paquet unzip avec apt
4) Utiliser la commande unzip pour extraire le fichier du zip
5) supprimer liste_francais.zip avec rm
6) renommer liste_francais.txt en dico.txt avec mv
7) Ecrire tous les mots qui commencent par z et qui finissent par e dans un fichier z-e.txt
8) Afficher le nombre de mots dans ce fichier z-e.txt avec une commande

   BONUS :
   Afficher le nombre de commandes grep que vous avez utilisé depuis le début !

CURL <URL>

Récupérer avec une requête curl seulement les pays qui commencent par “Al”

Utiliser ce site : https://www.atlas-monde.net/tous-les-pays/

Exercice 2

1) Créer un répertoire exo2 dans le dossier personnel
2) Télécharger le zip du dico à cette adresse (wget)
   http://www.3zsoftware.com/listes/liste_francais.zip
3) Installer le paquet unzip avec apt
4) Utiliser la commande unzip pour extraire le fichier du zip
5) supprimer liste_francais.zip avec rm
6) renommer liste_francais.txt en dico.txt avec mv
7) Compter le nombre de mots qui commencent par z dans ce dictionnaire
8) Enregistrer le résultat dans un fichier nb_z_words.txt (avec la redirection >

Scripting

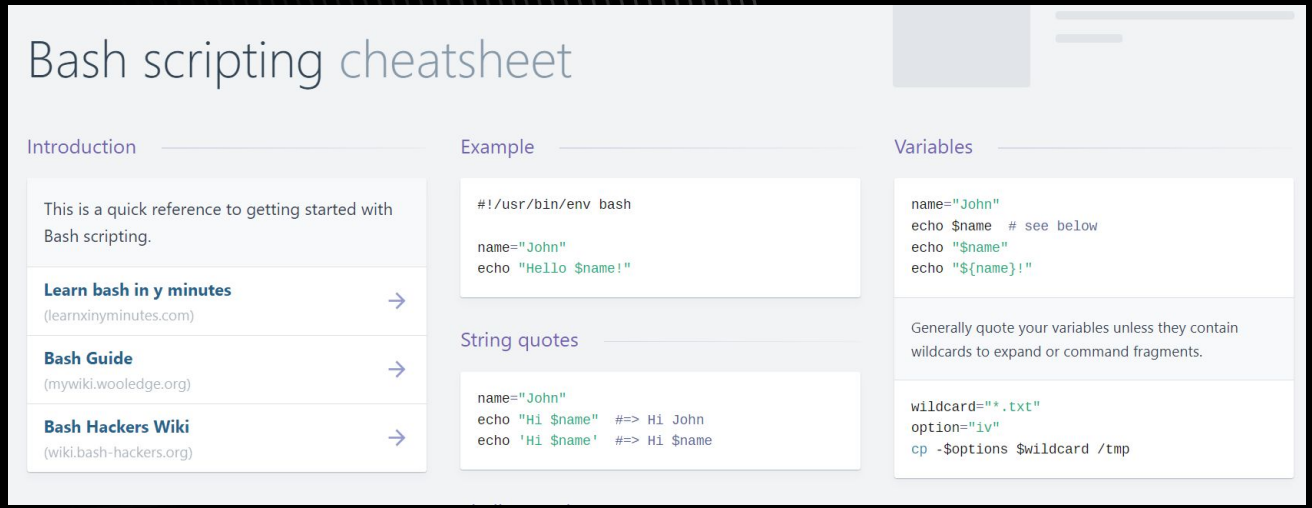

$_


bash saying my script has errors
me who can't even write a single line of bash without consulting google


MY CODE
LUCK
OLD REPOSITORIES
INDIAN GUY ON YOUTUBE
STACKOVERFLOW
GOOGLE

Cheat sheets

https://devhints.io/bash
https://devhints.io/regexp



Bash scripting cheatsheet

Introduction

This is a quick reference to getting started with Bash scripting.

Learn bash in y minutes
(learnxinyminutes.com)

Bash Guide
(mywiki.wooledge.org)

Bash Hackers Wiki
(wiki.bash-hackers.org)

Example

```
#!/usr/bin/env bash

name="John"
echo "Hello $name!"
```

String quotes

```
name="John"
echo "Hi $name"  #=> Hi John
echo 'Hi $name'  #=> Hi $name
```

Variables

```
name="John"
echo $name  # see below
echo "$name"
echo "${name}!"
```

Generally quote your variables unless they contain wildcards to expand or command fragments.

```
wildcard="*.txt"
option="iv"
cp -$options $wildcard /tmp
```

SCRIPTS BASH

```
  GNU nano 2.9.3                                    script.sh

#!/bin/bash
mkdir ./dossier1
cd ./dossier1
touch fichier1
touch fichier2
```

SCRIPTS BASH

```
  GNU nano 2.9.3                                    script.sh

#!/bin/bash
mkdir ./dossier1
cd ./dossier1
touch fichier1
touch fichier2
```

```
chmod 700 script.sh
bash script.sh
```

Exercice

EASY

Créer un script bash qui crée un dossier test_bash, et qui crée trois fichiers à l’intérieur 1.txt 2.txt 3.txt puis qui copie le dossier test_bash vers un autre dossier test_bash_2

Créer une commande

un alias permet de créer des commandes personnalisées

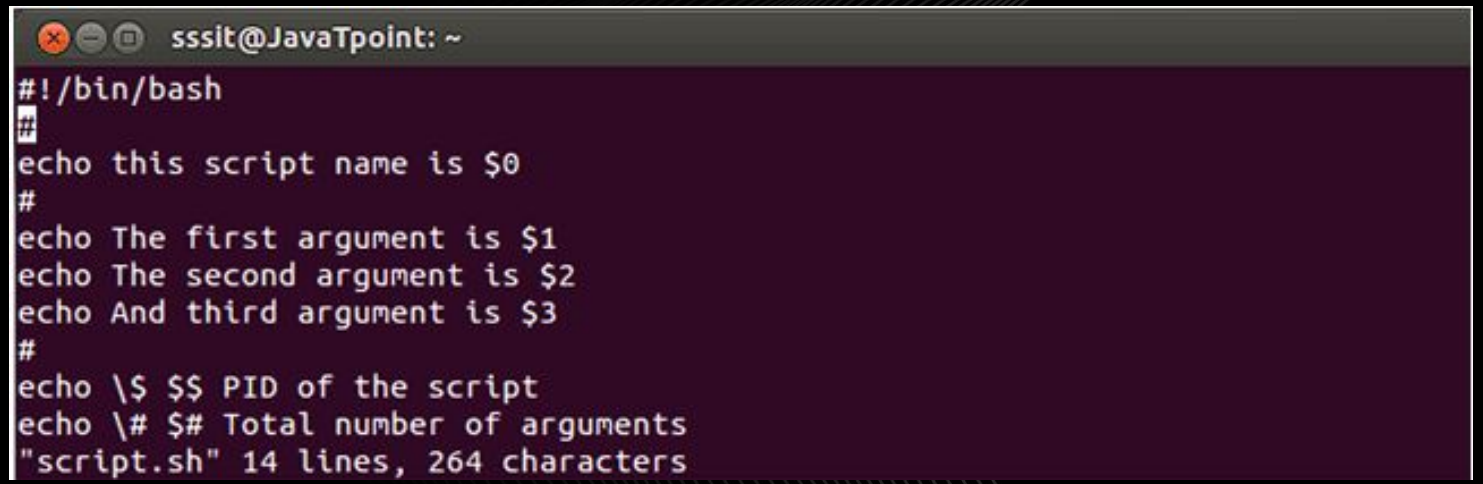
alias [nom_alias]=[cmd]

Exercice : créer la commande cree_deux_dossiers qui lance le script de l’exercice précédent

Les variables


```
#!/bin/bash
var1="one"
var2="piece"
echo $var1 $var2
```

passage d’arguments



$1 - $9 : arguments de la commande

./run_script.sh arg1 arg2 arg3

Exercice

EASY

Créer un script qui prend un chemin en premier argument et qui renvoie le nombre de mots cumulé de tous les fichiers qui composent le dossier

Opérations

```
#!/bin/bash

var=$((3+9))
echo $var
```

OPÉRATEUR	USAGE
+	ajout
-	soustraction
*	multiplication
/	division
**	exponentiation
%	module

Exercice

MEDIUM

Créer un script bash qui prend 3 arguments : un nombre de victoires, d’égalité et de défaites d’une équipe de foot et qui renvoie son score : une victoire compte pour trois points, une égalité compte pour 1 point, une défaite enlève 1 point

Enregistrer une commande dans une variable

MEDIUM

```
var=$(cmd)
```

EXERCICE

MEDIUM

CRÉER UN SCRIPT QUI PREND UN URL DE PAGE WIKIPÉDIA EN PREMIER PARAMETRE, QUI PREND UN MOT EN SECOND PARAMETRE ET RENVOIE EN MESSAGE LE NOMBRE DE FOIS QUE LE MOT EST APPARU DANS LA PAGE WEB

EXERCICE

MEDIUM

```
#!/bin/bash
content_html=$(curl $1 | grep -o $2)
number_words=$(echo $content_html | wc -w)
echo "Le nombre de mot $2 que j'ai trouvé dans le code de la page est : $number_words"
```

CONDITIONS BASH

OPÉRATION	SYNTAXE	EXPLICATION
Égalité	num1 -eq num2	est num1 égal à num2
Supérieur à égal à	num1 -ge num2	est num1 supérieur à égal à num2
Plus grand que	num1 -gt num2	est num1 supérieur à num2
Moins qu'égal à	num1 -le num2	est num1 inférieur à égal à num2
Moins que	num1 -lt num2	est num1 inférieur à num2
Pas égal à	num1 -ne num2	est num1 différent de num2

```
read x
read y

if [ $x -gt $y ]
then
echo X is greater than Y
elif [ $x -lt $y ]
then
echo X is less than Y
elif [ $x -eq $y ]
then
echo X is equal to Y
fi
```

CONDITIONS BASH

EASY

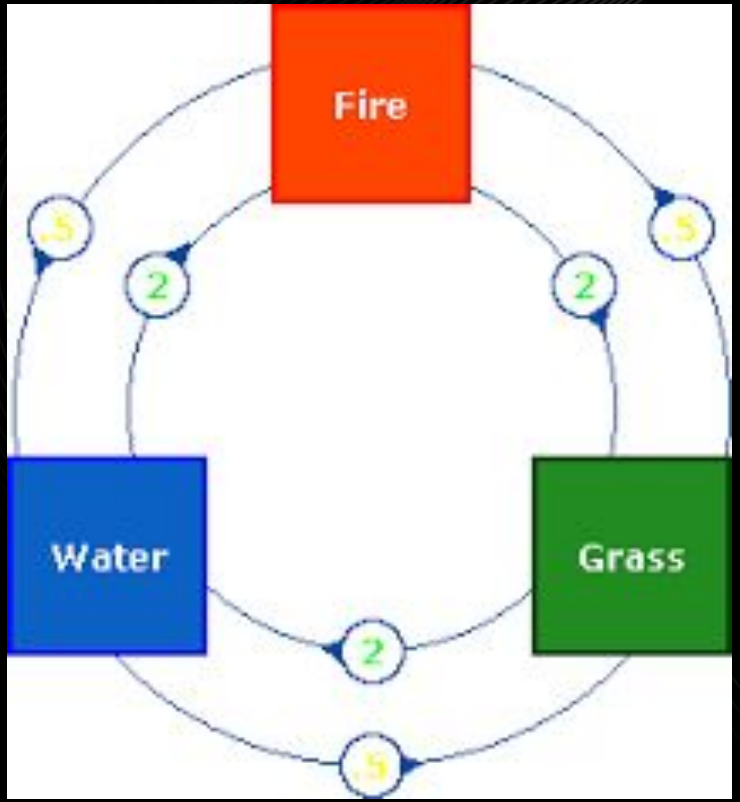
CRÉER UN SCRIPT QUI PREND DEUX NOMBRES EN ARGUMENT, ET QUI RENVOIE LE MESSAGE "LESS_THAN_100" SI LA SOMME DES DEUX NOMBRES EST INFÉRIEURE À 100, "MORE_OR_EQUAL_THAN_100" SINON

CONDITIONS BASH

MEDIUM

CRÉER UN SCRIPT QUI PREND DEUX ARGUMENTS : LE TYPE DU POKEMON 1 ET LE TYPE DU POKÉMON 2 ET QUI RENVOIE EN MESSAGE L'EFFICACITÉ DE L'ATTAQUE
(3 TYPES POSSIBLES : FEU, EAU, PLANTE)
(EFFICACITÉ :
- SUPER EFFICACE !
- CE N'EST PAS TRÈS EFFICACE !

CONDITIONS BASH - EFFICACITE



SCRIPTS BASH - BOUCLES

Loops

Basic for loop

```
for i in /etc/rc.*; do
  echo "$i"
done
```

C-like for loop

```
for ((i = 0 ; i < 100 ; i++)); do
  echo "$i"
done
```

Ranges

```
for i in {1..5}; do
    echo "Welcome $i"
done
```

With step size

```
for i in {5..50..5}; do
    echo "Welcome $i"
done
```

Reading lines

```
while read -r line; do
  echo "$line"
done <file.txt
```

Forever

```
while true; do
  ...
done
```

BOUCLES EN BASH

```
#!/bin/bash
for i in {1..5}; do
  touch "${i}.txt"
done
```

SCRIPTS BASH - BOUCLES

```
#!/bin/bash
for i in {1..5}; do
  touch "${i}.txt"
done
```

RUN.SH

```
(base) eol@opFenice:~/Documents/travail/regexp/test$ ls
run.sh
(base) eol@opFenice:~/Documents/travail/regexp/test$ ./run.sh
(base) eol@opFenice:~/Documents/travail/regexp/test$ ls
1.txt  2.txt  3.txt  4.txt  5.txt  run.sh
(base) eol@opFenice:~/Documents/travail/regexp/test$
```

SCRIPTS BASH - BOUCLES

HARDCORE

SCRIPTS BASH - BOUCLES

HARDCORE

MISE EN SITUATION :

- 50% DE LA CLASSE VA SE FAIRE ASPIRER L'ÂME
- POUR DÉTERMINER QUI VIT ET QUI MEURT :
- VOUS AVEZ 2 DÉS 20, VOUS POUVEZ JUSTE DÉCIDER UN NOMBRE DE LANCERS
- A CHAQUE LANCER DES DEUX DÉS, ON ADDITIONNE LA SOMME DES DEUX NOMBRES ET ON AJOUTE CETTE VALEUR À UN SCORE TOTAL
- LES 50% DE LA CLASSE AVEC LES PLUS HAUTS SCORE VIVENT
- SI UN DOUBLE APPARAÎT, LE SCORE **FINAL** DEVIENT 0

Plus de commandes !

```
tree

find [path] -name [motif]

find ./ -name fichier

find ./ -name fich*
```

Planifier des tâches avec cron

- Lancer des scripts de facon réguliere
- Choix de l’intervalle et de la date
- usermod -aG crontab (username)
- su username

```
fuokov2@opFenice:~$ crontab -e
```

Planifier des tâches avec cron

```
  GNU nano 6.2                /tmp/crontab.r5p8dW/crontab

# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

* * * * * mkdir /home/fuoko/test.cron
```

PLANIFIER DES TÂCHES AVEC CRON

crontab guru

The quick and simple editor for cron schedule expressions by Cronitor

"At every minute."

next at 2022-12-12 11:08:00

random

```
* * * * *
```

minute	hour	day (month)	month	day (week)

*	any value
,	value list separator
-	range of values
/	step values
@yearly	(non-standard)
@annually	(non-standard)
@monthly	(non-standard)
@weekly	(non-standard)
@daily	(non-standard)
@hourly	(non-standard)
@reboot	(non-standard)

Planifier des tâches avec cron

- Une fois par mois
- Une fois tous les deux jours à 10 heures
- Le 6eme, 13eme et 24eme jour du mois à midi
- Toutes les trente minutes

Planifier des tâches avec cron

Créer une tâche cron mensuelle à 10 heures du matin qui fait une copie du contenu du dossier Documents vers un nouveau dossier “backup” dans le dossier personnel de l’utilisateur

Planifier des tâches avec cron

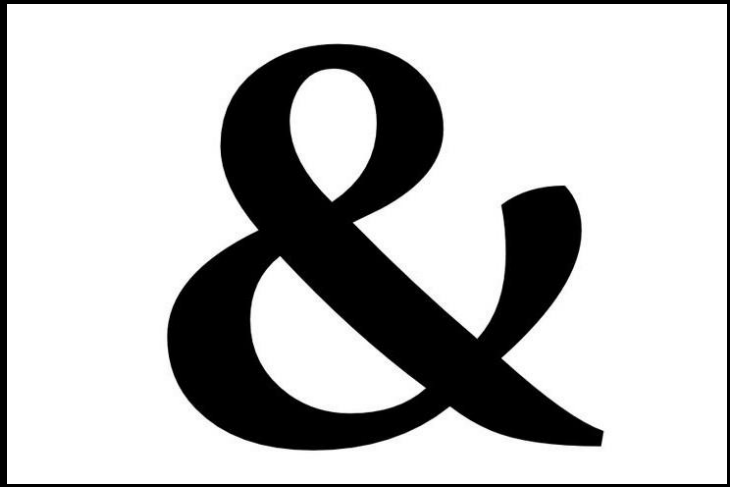
Créer une tâche cron qui lance un scrip
bash toutes les minutes.

Le script est le suivant :

- Si le fichier les_minutes_passent.tx
  n’existe pas dans le répertoire
  personnel, il est créé avec comme
  contenu le chiffre 0.
- Sinon, ce fichier est remplacé en
  incrémentant le nombre présent dans
  l’ancien fichier de 1.

LANCER UN PROGRAMME EN TÂCHE DE FOND

COMMANDE &



Variables d’environnement

Pour afficher les variables d’env :

```
printenv
```

Pour créer une variable d’env :

```
export env_var=10
```

```
echo $env_var
```

On peut aussi Modifier le fichier ~/.bashrc

Exercice variable d’env #1 EZ PZ LIFE IS EZ

- Créer une variable appelée chemin qui pointe vers le répertoire en cours

- Afficher son contenu. Fermer le terminal et le réouvrir. La variable existe-t-elle toujours?

- Editer le fichier ~/.bashrc et ajouter la variable d’env. Ouvrir un terminal : la var est-elle enregistrée?
- Afficher seulement la ligne de printenv qui contient la variable “chemin”

Exercice variable d’env #1 EZ PZ LIFE IS EZ

- Créer une variable EXTENSION_base qui a comme contenu .sh

- Dans un répertoire, créer 4 fichiers : exo1.sh exo2.sh exo3.sh EXO1.fake

- Avec une commande et en utilisant la variable EXTENSION_base, afficher les fichiers qui se terminent par cette dernière.