Système d'exploitation LINUX

Gabriel Furbringer, Gymnase du Bugnon, le 29.05.2018

Dans ce Jupyter notebook, nous parlerons de :

- du systeme dexploitation LINUX
- l'arborescence Linux
- commandes unix

Sources :

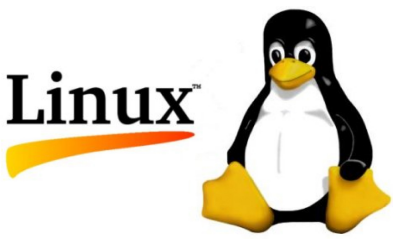
- https://doc.ubuntu-fr.org/arborescence (https://doc.ubuntu-fr.org/arborescence)
- https://fr.wikipedia.org/wiki/Linux (https://fr.wikipedia.org/wiki/Linux)
- https://fr.wikipedia.org/wiki/Noyau_Linux (https://fr.wikipedia.org/wiki/Noyau_Linux)
- https://fr.wikipedia.org/wiki/Noyau_de_syst%C3%A8me_d%27exploitation (https://fr.wikipedia.org/wiki/Noyau_de_syst%C3%A8me_d%27exploitation)
- https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation (https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation)
- https://jakevdp.github.io/PythonDataScienceHandbook/01.05-ipython-and-shell-commands.html#Shell-Commands-in-IPython (https://jakevdp.github.io/PythonDataScienceHandbook/01.05-ipython-and-shell-commands.html#Shell-Commands-in-IPython)

Historique

Le *Linux* est un noyau de système d'exploitation de type *UNIX*. Il est un logiciel libre développé essentiellement en *langage C*. Un noyau de système d'exploitation "*gère les ressources de l'ordinateur et permet aux différents composants — matériels et logiciels — de communiquer entre eux.*"

C'est en 1991 que Linus Torvalds, étudiant finlandais commence le développement d'un noyau de système d'exploitation, autrement appelé *Linux*.

Le mars 1994 sort la version *Linux* 1.0.

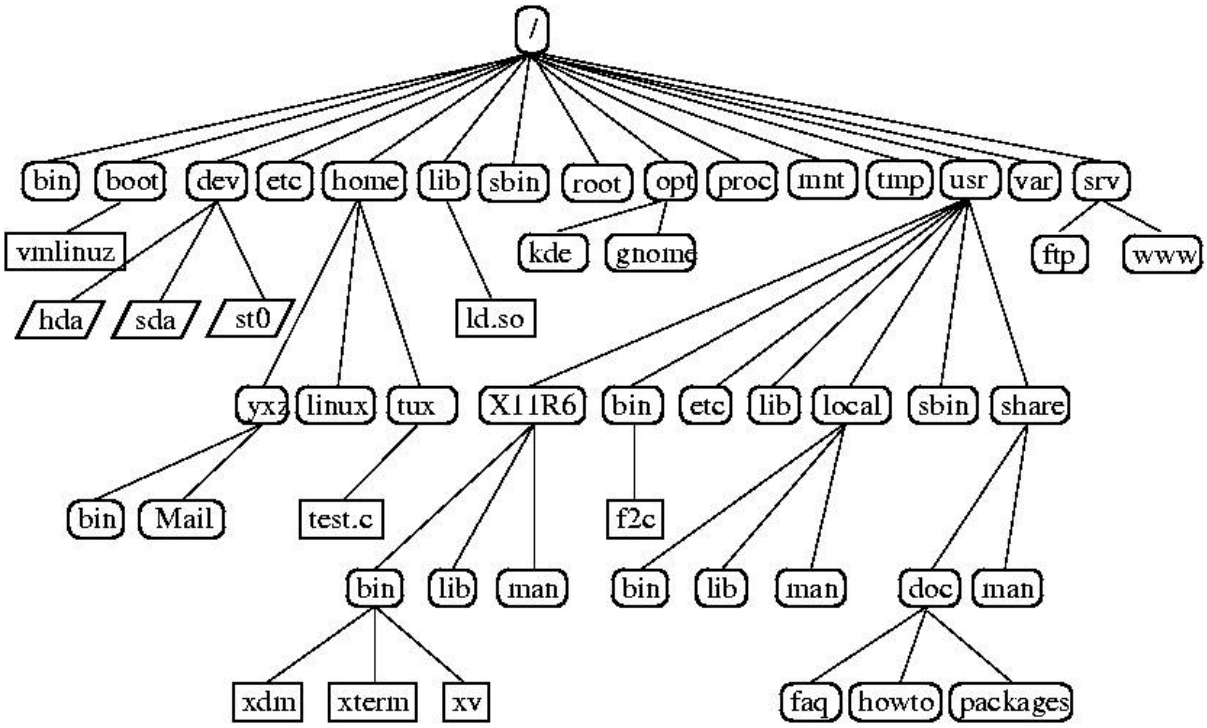


Système d'exploitation

Un système dêxploitation permet à l'utilisateur et aux programmes de l'utilisateur de se lier et utiliser les composants physiques de l'ordinateur.

L'arborescence LINUX

Imaginez vos répertoires et vos fichiers dans ton ordinateur comme un arbre. Il commence par la racine la source et les dossiers important comme ***bin***, les dossiers sont le bois, parfois fin comme des branches, parfois solide comme le tronc; et les fichiers comme les feuilles. Et toutes les informations partant de la racine (indiquée par une barre oblique au départ), ont un chemin bien défini:



Regardez, le dossier **bin**: c'est un dossier qui contient des commandes, nous avons trié et selctionné les élément commencant par m . Il y'a par expemle la commande `mv` que nous verrons plus tard ou alors la commande `mkdir`

In [1]:

```
%ls /bin/m*
```

```
/bin/machinectl*  /bin/mktemp*        /bin/mount*       /bin/mt-gnu*
/bin/mkdir*       /bin/modeline2fb*   /bin/mountpoint*  /bin/mv*
/bin/mknod*       /bin/more*          /bin/mt@
```

Il y'a aussi le dossier home qui contient le dossier d'utilisateur *pi* .

In [3]:

```
%ls /home
```

```
pi/
```

Commandes UNIX dans le terminal

Les systèmes d'exploitation de type UNIX permettent plusieurs commandes interéssantes pour le traitement de dossier, fichiers. Elles sont stockés soit directement dans le shell du terminal ou parfois dans /bin ou dans /usr/bin.

Dans jupyter notebook, nous utilisons `%` ou `!` pour accéder au terminal. `%` *(magic command)* permet d'utiliser certaines commande qui ne marche pas avec `!` *(shell command)*, car `!` exécute les commandes dans un sous-shell temporaire et quand nous voulons nous déplacer dans un autre dossier, nous voudrions y rester. C'est pour cela que nous utiliserons parfois `%` pour que nos déplacements soient définitifs.

Commandes:

In [1]:

```
#Il est préférable de faire ces commandes pour éviter de créer des fichiers
# éparpillés, nous créons donc un dossier qui contiendra tout le processus
#nous nous déplacons dans le dossier "Unix"
!mkdir Unix
%cd Unix
```

```
/home/pi/Documents/oc-2017/exam/Unix
```

Commandes pour se déplacer et comprendre sa position

`pwd` est une commande qui permet d'afficher le chemin de notre position actuelle dans l'arborescence relativement à la source

In [2]:

```
!pwd
# nous sommes dans le dossier Unix qui est dans le dossier oc-2017, etc
```

```
/home/pi/Documents/oc-2017/exam/Unix
```

`cd` est une commande qui permet de se déplacer dans l'arborescence en avant *(par exemple "/Documents/")* et en arrière *(par exemple " /../")*

In [3]:

```
#commencer cd par "/", permet de commencer à la source
%cd /home/pi/Documents/
```

```
/home/pi/Documents
```

In [4]:

```
#nous sommes maintenant dans le dossier Documents
!pwd
```

```
/home/pi/Documents
```

```
`cd` admet l'argument `-` qui permet de revenir à la position précédente
```

In [5]:

```
%cd -
```

```
/home/pi/Documents/oc-2017/exam/Unix
```

`ls` est une commande qui permet d'afficher la liste des fichiers d'un fichier, en ajoutant `..` nous pouvons regarder la liste des fichiers qui contient le dossier dans lequel nous sommes

In [6]:

```
!ls ..
# regardons le dossier des résumés de vos examens
```

```
Benoit_OpenCV.ipynb           img_jonas
Benoit_OpenCV.pdf             Jonas_Pythonintroduction.ipynb
Blanc_Git_GitHub.ipynb        Jonas_Pythonintroduction.pdf
Blanc_Git_GitHub.pdf          README.md
Fleischer_LaTeX-NumPy.ipynb   résumé Albert Guedj POO.pdf
Fleischer_LaTeX-NumPy.pdf     Résumé_Gue_POO.ipynb
Furbringer_Linux_Unix.ipynb   Samuel_PythonStructuresDonnees.ipynb
img_furb                      trex.png
img_gabriel                   Unix
```

`ls` admet différents arguments tel que : `-l` ,

In [7]:

```
!ls .. -l
```

```
total 8700
-rw-r--r-- 1 pi pi 1686626 mai 29 22:02 Benoit_OpenCV.ipynb
-rw-r--r-- 1 pi pi 1644166 mai 29 22:02 Benoit_OpenCV.pdf
-rw-r--r-- 1 pi pi   35749 mai 29 22:02 Blanc_Git_GitHub.ipynb
-rw-r--r-- 1 pi pi  658226 mai 29 22:02 Blanc_Git_GitHub.pdf
-rw-r--r-- 1 pi pi  139530 mai 29 22:02 Fleischer_LaTeX-NumPy.ipynb
-rw-r--r-- 1 pi pi  279345 mai 29 22:02 Fleischer_LaTeX-NumPy.pdf
-rw-r--r-- 1 pi pi   33623 mai 31 12:01 Furbringer_Linux_Unix.ipynb
drwxr-xr-x 3 pi pi    4096 mai 29 20:42 img_furb
drwxr-xr-x 2 pi pi    4096 mai 29 22:02 img_gabriel
drwxr-xr-x 2 pi pi    4096 mai 29 22:02 img_jonas
-rw-r--r-- 1 pi pi   40641 mai 29 22:02 Jonas_Pythonintroduction.ipynb
-rw-r--r-- 1 pi pi  328664 mai 29 22:02 Jonas_Pythonintroduction.pdf
-rw-r--r-- 1 pi pi     535 mai 29 22:02 README.md
-rw-r--r-- 1 pi pi 3876897 mai 29 22:02 résumé Albert Guedj POO.pdf
-rw-r--r-- 1 pi pi   53239 mai 29 22:02 Résumé_Gue_POO.ipynb
-rw-r--r-- 1 pi pi   12958 mai 29 22:02 Samuel_PythonStructuresDonnees.i
pynb
-rw-r--r-- 1 pi pi   71544 mai 29 22:02 trex.png
drwxr-xr-x 2 pi pi    4096 mai 31 12:02 Unix
```

ce qui permet d'avoir une liste des éléments plus précise,

`ls` admet aussi `-a` , qui permet de voir des éléments qui seraient cachés

In [8]:

```
!ls .. -a
# si nous comparons nous voyons que de
#nouveaeu fichier sont présents tel que "." ou ".."
```

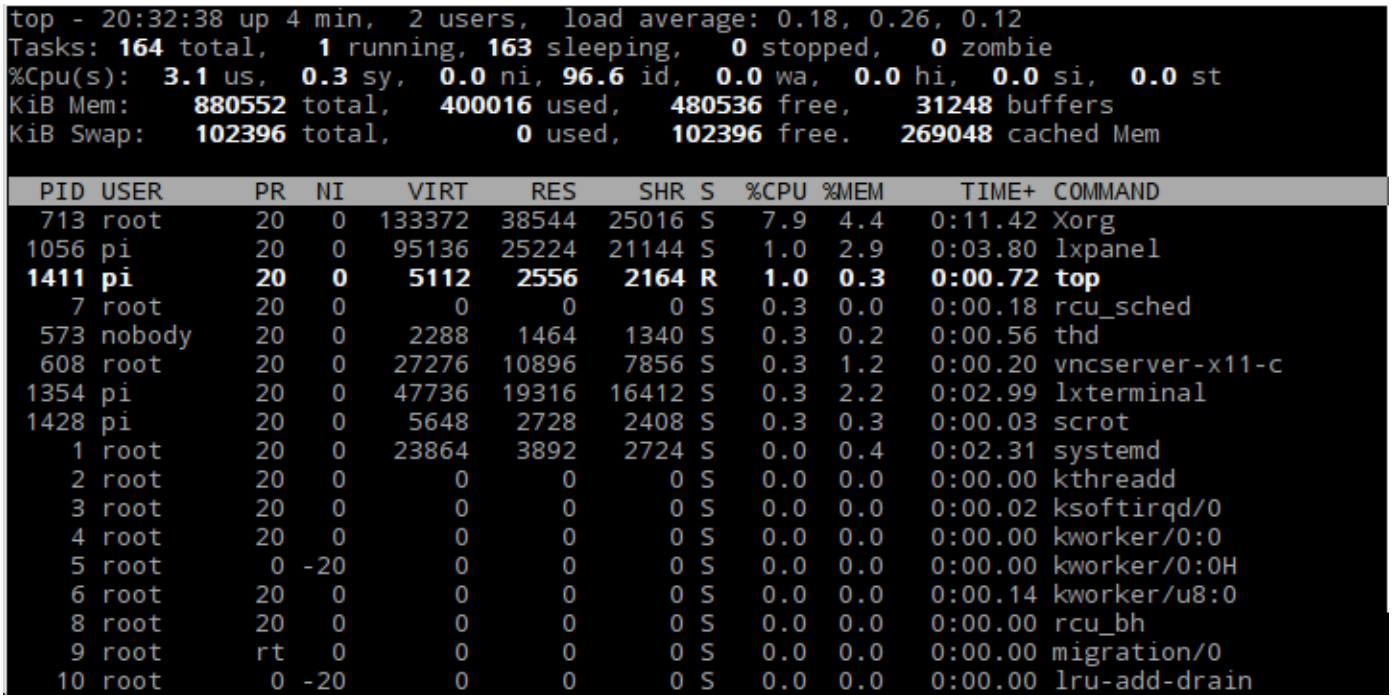
```
.                             img_jonas
..                            .ipynb_checkpoints
Benoit_OpenCV.ipynb           Jonas_Pythonintroduction.ipynb
Benoit_OpenCV.pdf             Jonas_Pythonintroduction.pdf
Blanc_Git_GitHub.ipynb        README.md
Blanc_Git_GitHub.pdf          résumé Albert Guedj POO.pdf
Fleischer_LaTeX-NumPy.ipynb   Résumé_Gue_POO.ipynb
Fleischer_LaTeX-NumPy.pdf     Samuel_PythonStructuresDonnees.ipynb
Furbringer_Linux_Unix.ipynb   trex.png
img_furb                      Unix
img_gabriel
```

Commande top

top est une commande qui permet de voir les logiciels qui sont entrain d'agir affichant entre autres leurs vrais noms et leurs *PID*

In [9]:

```
#!top
#nous ne le lancerons pas,car cela prendrait trop de place me voyez une image
#mais essayez sur votre terminal et vous comprendrez mieux
```



```
top - 20:32:38 up 4 min,  2 users,  load average: 0.18, 0.26, 0.12
Tasks: 164 total,   1 running, 163 sleeping,   0 stopped,   0 zombie
%Cpu(s):  3.1 us,  0.3 sy,  0.0 ni, 96.6 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:    880552 total,   400016 used,   480536 free,    31248 buffers
KiB Swap:   102396 total,        0 used,   102396 free.   269048 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  713 root      20   0  133372  38544  25016 S   7.9  4.4   0:11.42 Xorg
 1056 pi        20   0   95136  25224  21144 S   1.0  2.9   0:03.80 lxpanel
 1411 pi        20   0    5112   2556   2164 R   1.0  0.3   0:00.72 top
    7 root      20   0       0      0      0 S   0.3  0.0   0:00.18 rcu_sched
  573 nobody    20   0    2288   1464   1340 S   0.3  0.2   0:00.56 thd
  608 root      20   0   27276  10896   7856 S   0.3  1.2   0:00.20 vncserver-x11-c
 1354 pi        20   0   47736  19316  16412 S   0.3  2.2   0:02.99 lxterminal
 1428 pi        20   0    5648   2728   2408 S   0.3  0.3   0:00.03 scrot
    1 root      20   0   23864   3892   2724 S   0.0  0.4   0:02.31 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.02 ksoftirqd/0
    4 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 S   0.0  0.0   0:00.14 kworker/u8:0
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 migration/0
   10 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 lru-add-drain
```

Commandes pour gérer des dossiers

`mkdir` est une commande qui permet de créer un dossier vide

In [10]:

```
!ls
```

In [11]:

```
!mkdir dossierA
!mkdir dossierB
!ls
# il n'yavait rien et à présent il y'a deux dossiers
```

```
dossierA  dossierB
```

`rmdir` est une commande qui permet de supprimer un dossier vide

In [12]:

```
!rmdir dossierA
!ls
#il n'y a plus que le B
```

```
dossierB
```

`mv` est une commande qui permet de déplacer un dossier dans un autre dossier

In [13]:

```
!ls
!mkdir Graal
!mkdir Arthur
!mv Arthur Graal
```

```
dossierB
```

In [14]:

```
!ls Graal
```

```
Arthur
```

mais `mv` permet aussi le nom d'un dossier/fichier

In [15]:

```
%cd Graal
!mv Arthur King
!ls
```

```
/home/pi/Documents/oc-2017/exam/Unix/Graal
King
```

Commandes utiles pour la vie quotidienne

`date` est une commande qui permet d'indiquer la date d'aujourd'hui et l'heure actuelle

In [16]:

```
!date
#Est-ce juste?
```

```
jeudi 31 mai 2018, 12:03:35 (UTC+0000)
```

`cal` est une commande qui permet d'afficher le calendrier du mois actuel

In [17]:

```
!cal
```

```
      Mai 2018
di lu ma me je ve sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

`cal 2017` permet d'afficher le calendrier entier de 2017

In [18]:

```
!cal 2017
#essayez avec une année dans le futur pour savoir quel sera
#le jour de la semaine où tombera votre date de fête
```

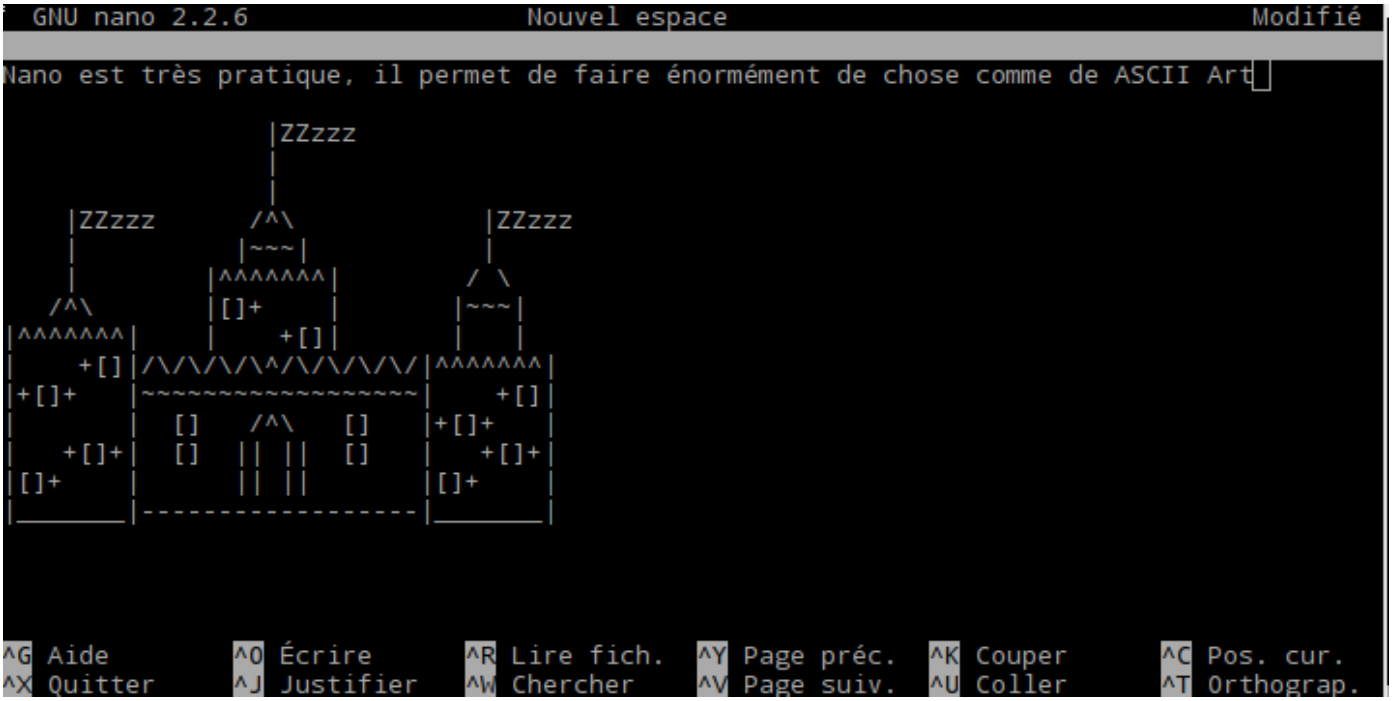
```
                               2017
      Janvier               Février                Mars
di lu ma me je ve sa  di lu ma me je ve sa  di lu ma me je ve sa
 1  2  3  4  5  6  7              1  2  3  4            1  2  3  4
 8  9 10 11 12 13 14   5  6  7  8  9 10 11   5  6  7  8  9 10 11
15 16 17 18 19 20 21  12 13 14 15 16 17 18  12 13 14 15 16 17 18
22 23 24 25 26 27 28  19 20 21 22 23 24 25  19 20 21 22 23 24 25
29 30 31              26 27 28              26 27 28 29 30 31


       Avril                  Mai                   Juin
di lu ma me je ve sa  di lu ma me je ve sa  di lu ma me je ve sa
                   1      1  2  3  4  5  6               1  2  3
 2  3  4  5  6  7  8   7  8  9 10 11 12 13   4  5  6  7  8  9 10
 9 10 11 12 13 14 15  14 15 16 17 18 19 20  11 12 13 14 15 16 17
16 17 18 19 20 21 22  21 22 23 24 25 26 27  18 19 20 21 22 23 24
23 24 25 26 27 28 29  28 29 30 31           25 26 27 28 29 30
30

      Juillet                Août                Septembre
di lu ma me je ve sa  di lu ma me je ve sa  di lu ma me je ve sa
                   1         1  2  3  4  5                  1  2
 2  3  4  5  6  7  8   6  7  8  9 10 11 12   3  4  5  6  7  8  9
 9 10 11 12 13 14 15  13 14 15 16 17 18 19  10 11 12 13 14 15 16
16 17 18 19 20 21 22  20 21 22 23 24 25 26  17 18 19 20 21 22 23
23 24 25 26 27 28 29  27 28 29 30 31        24 25 26 27 28 29 30
30 31

      Octobre               Novembre              Décembre
di lu ma me je ve sa  di lu ma me je ve sa  di lu ma me je ve sa
 1  2  3  4  5  6  7              1  2  3  4                  1  2
 8  9 10 11 12 13 14   5  6  7  8  9 10 11   3  4  5  6  7  8  9
15 16 17 18 19 20 21  12 13 14 15 16 17 18  10 11 12 13 14 15 16
22 23 24 25 26 27 28  19 20 21 22 23 24 25  17 18 19 20 21 22 23
29 30 31              26 27 28 29 30        24 25 26 27 28 29 30
                                            31
```

Commandes pour gérer des fichiers txt

nano est une commande qui ouvre une fenêtre pour faire du traitement de texte

In [19]:

```
#!nano
#Jupyter notebook n'arrive pas à afficher nano donc voici une image,
# mais essayez dans votre terminal
```



Pour ajouter le calendrier dans un fichier il suffit de faire cela: `cal 100> cal100.txt`

In [20]:

```
%cd ..
!cal 100 > cal100.txt
```

```
/home/pi/Documents/oc-2017/exam/Unix
```

In [45]:

```
#nano cal100.txt
#faites cette commande dans votre terminal,
#quand vous êtes dans le dossier Unix
#regardez ce qu'il se passe
```

Commandes relatives aux utilisateurs

`users` , `who` et `w` sont des commandes qui permettent de savoir qui sont les utilisateurs connectés sur la machine

In [22]:

```
!who
```

```
pi       :0           2018-05-29 22:32 (:0)
pi       tty1         2018-05-29 22:32
```

In [23]:

```
!users
```

```
pi pi
```

In [24]:

```
!w
```

```
 12:03:47 up 34 min,  2 users,  load average: 0.57, 0.48, 0.37
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
pi       :0       :0               mar22   ?xdm?  11:44   0.29s /usr/bi
n/lxsess
pi       tty1                      mar22   37:31m  0.37s  0.35s -bash
```

`write` est une commande qui permet de parler avec la personne choisit

In [25]:

```
! write pi
#Mais ici ca ne marche pas, car vous êtes
#normalement seul sur votre ordinateur
```

```
write: pi has messages disabled
```

Commandes diverses

`echo` est une commande qui permet d'écrire des caractères dans le terminal

In [26]:

```
!echo ni ni ni it it it
```

```
ni ni ni it it it
```

`cp` est une commande qui permet de copier et coller un fichier

In [27]:

```
!echo ni ni ni>knight.txt
#nous avons créé un fichier texte qui va
#être observé pour la commande cp
```

In [28]:

```
!cp knight.txt Graal
!cp cal100.txt Graal
```

In [29]:

```
!ls Graal
#dans le dossier Graal il y a les deux fichiers txt
```

```
cal100.txt  King  knight.txt
```

In [31]:

```
!ls
#et dans le dossier qui contient Graal il y'a les deux fichiers txt
#identique à ceux à l'intérieur de Graal
```

```
cal100.txt  dossierB  Graal  knight.txt
```

`sudo` est un ajout au début de la commande pour s'accorder les droits super user juste pour une commande

Les exemples de `sudo` seront après pour `open` et `pkill`

`which` est une commande qui donne le chemin relativement à la source d'un fichier

In [32]:

```
!which minecraft-pi
```

```
/usr/bin/minecraft-pi
```

`open` est une commande qui permet d'ouvrir un fichier en donnant le chemin

In [33]:

```
! sudo open /usr/bin/minecraft-pi
#Une fenêtre vient de s'ouvrir
```

`kill` est une commande qui permet d'arrêter le processus en cours d'un logiciel en précisant son PID que vous pouvez trouver grace à la commande `top`

`pkill` est une commande qui permet d'arrêter le processus en cours d'un logiciel en précisant son vrai nom que vous pouvez trouver grace à la commande `top`

In [34]:

```
!sudo pkill minecraft-pi
#Une fenêtre vient de se fermer
```

`rm -r` est une commande qui efface un fichier et un dossier qu'il soit vide ou pas

In [36]:

```
%cd Unix
!rm -r Graal
!ls
```

```
/home/pi/Documents/oc-2017/exam/Unix
cal100.txt  dossierB  knight.txt
```

`w3m` est une commande pour ouvrir une internet en texte

In [39]:

```
#!w3m google.com
#essayez et naviguez sur votre terminal
```

Pour télécharger w3m, il faut l'installer ainsi: