0.1 Introduction

0.1.1 Linux Is Now on Mars

Le 18 février 2021, Le rover *Perseverance* amarsissait avec succès. Il a pour missions de réaliser des prélèvements du sol martien et de roches, d'en faire les premières analyses sur place et de rapporter les échantillons pour des mesures approfondies. Il peut tourner des vidéos de Mars grâce à un drone unique en son genre: *Ingenuity*. C'est un minuscule hélicoptère de la NASA, devenu le premier aéronef motorisé à voler sur une autre planète, Mars.

Cet exploit technique a été réalisé grâce à **Linux**, **système d'exploitation libre** et à d'autres **logiciels libres**.

Figure 1: Perseverance sur Mars

0.1.2 Le programme

Systèmes d'exploitation	Identifier les fonctions d'un système d'exploitation. Utiliser les commandes de base en ligne de commande. Gérer les droits et permissions d'accès aux fichiers.	Les différences entre systèmes d'exploitation libres et propriétaires sont évoquées. Les élèves utilisent un système d'exploitation libre. Il ne s'agit pas d'une étude théorique des systèmes d'exploitation.

Figure 2: Programme-système d'exploitation

Au minimum, les commandes suivantes devront être connues:

- `man` pour se documenter sur une commande.
- `pwd, cd, ls, tree, file, touch, mkdir, cp, mv, rm, chmod, cat, grep`.
- `ip/ifconfig-route, ping, traceroute` pour la partie *réseau*.

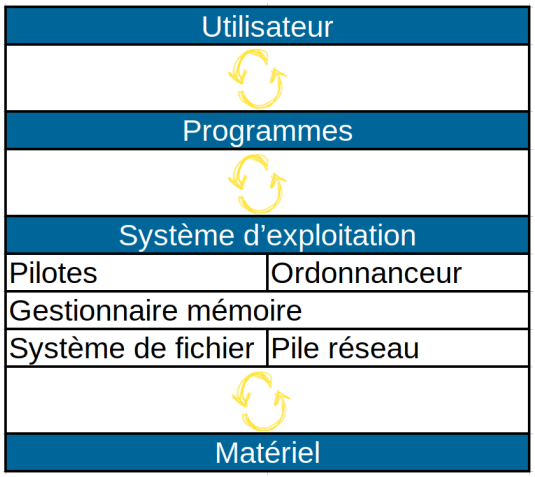
- ps, top, kill pour la gestion des processus (terminale).

0.2 Système d'exploitation

Le système d'exploitation SE (Operating System ou OS en anglais), est responsable de la gestion des ressources matérielles. Le SE se situe donc entre les utilisateurs et le matériel. C'est un ensemble de programme qui permet de gérer et partager les ressources. En voici les principales fonctions extraites d'un article plus complet sur interstices.info:

- Le système assure le stockage de programmes et de données de toutes sortes: textes, images, vidéos, films.
- Le second rôle du système consiste à gérer au mieux les ressources — matérielles et logicielles — qui sont nécessaires pour lancer et suivre l'exécution des applications locales ou distantes. Le système s'occupe ainsi des aspects technologiques et des contraintes d'utilisation des ressources partagées, qu'elles soient attribuées à tour de rôle, comme le sont le processeur et l'imprimante, ou divisées, comme le sont la mémoire et l'écran.
- le système d'exploitation assure — et c'est là son troisième rôle — la sécurité de fonctionnement en cas de panne d'origine interne (matérielle ou logicielle) ou d'agression provenant d'un environnement de plus en plus ouvert (le réseau). I

C'est donc cet OS qui gère la mémoire de votre ordinateur, la répartit entre tous les programmes. Il fait le lien entre votre matériel (carte graphique, mémoire, imprimante) et vos logiciels.



Le **noyau** ou **kernel** est la partie principale d'un système d'exploitation qui assure la communication directe avec les ressources matérielles.

0.3 GNU/ Linux

Linux est un système d'exploitation né du travail d'un étudiant Finnois en 1991: **Linus Torvalds** qui a fourni le noyau manquant au **projet GNU de Richard Stallman**, ce système **libre** de son vrai nom **GNU/Linux** est devenu l'un des plus emblématique de l'informatique moderne.

Comme macOS, c'est un système compatible avec le standard POSIX. Dans ce standard, les fichiers sont organisés selon une **arborescence**, le répertoire / en est la **racine**.

La console dans laquelle s'exécute un *shell* est l'interface historique des système POSIX. Elle permet de saisir des commandes qui vont afficher des résultats, lancer un programme ou modifier le système. Il suffit par exemple d'écrire le nom du programme que vous voulez utiliser pour le lancer.

0.3.1 Les logiciels libres

Vous connaissez Windows, macOS ou Android, ce dernier est lui-même un OS dérivé de Linux. macOS est basé sur UNIX.[1] Cependant, alors que les OS précédents sont des logiciels propriétaires, GNU/Linux est un **logiciel libre**.

[1] En réalité basé sur BSD, lui-même dérivé d'UNIX

Figure 3: La mascotte de Linux - *Antartica Bounds CC-BY-ND*

Liberté 0: Utiliser le programme comme vous voulez, pour n'importe quel usage ;

Liberté 1: Etudier le fonctionnement du programme, et de le modifier pour qu'il effectue vos tâches informatiques comme vous le souhaitez ;

Liberté 2: Distribuer des copies, donc d'aider les autres ;

Liberté 3: Modifier le code, et pourvoir le redistribuer ; en faisant cela, vous donnez à toute la communauté une possibilité de profiter de vos changements.

Libre ou propriétaire

Les logiciels dits **libres** sont sans contrainte d'utilisation, fournis avec leur code source (nécessaire à leur compréhension technique, leur évolution et leur entretien) et la possibilité – légale et technique – de les étudier, les transformer, les adapter, les redistribuer. En faisant cela, vous aidez et donnez à toute la communauté une possibilité de profiter de vos changements ; l'accès au code source est une condition nécessaire.

Au contraire, les logiciels propriétaires sont en général non ouverts, il est donc plus difficile voire illégal de les modifier. Diffusés uniquement sous forme d'applications exécutables, avec des licences extrêmement contraignantes quant à leur usage, et avec interdiction (légale et technique) de procéder à quelque analyse, adaptation ou amélioration que ce soit. L'utilisateur ne sait donc pas ce que contient le code source: impossible pour lui d'obtenir des informations sur les malwares, portes dérobées etc. susceptibles d'y être présents. La communauté du libre les qualifie de **privateurs**.

Maintenance de ces logiciels Les logiciels libres sont souvent maintenus par cette communauté, mais peuvent aussi l'être par des entreprises qui les utilisent et qui ont intérêt à ce qu'ils restent efficaces et utilisés par d'autres, ce qui assure l'existence de développeurs susceptibles de participer à leur maintien. Les logiciels propriétaires quant à eux sont essentiellement développés et mis à jour par l'entreprise qui les possède et qui peut décider d'arrêter de les maintenir. Ce sont deux modèles économiques très différents.

0.3.2 Arborescence du système de fichier

Rappelons les bonnes pratiques de nommage de fichiers:

Autorisés:

- les caractères alphanumériques,
- le tiret bas _ (tiret du 8),
- le trait d'union - (tiret du 6) sauf en début de nom.

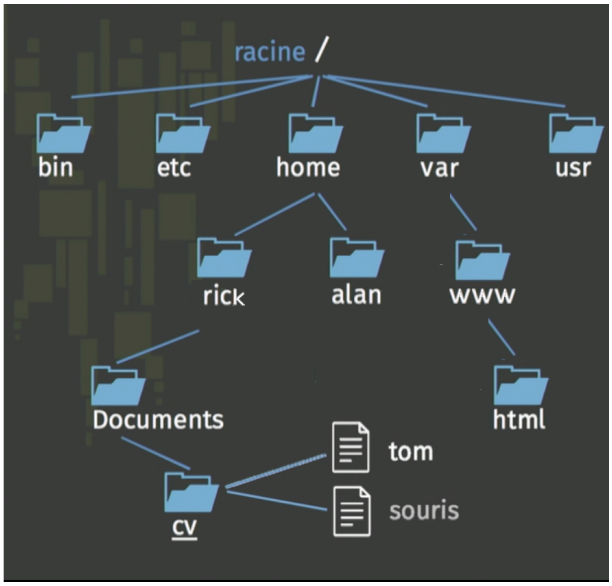
Interdits: Tout le reste !

- espaces, accents,
- symboles de ponctuation,
- = + * / & # % £ $ ...

La hiérarchie du système de gestion de fichier de type UNIX peut être représentée par un arbre au sens informatique du terme, dont les nœuds internes seraient les répertoires et les feuilles les fichiers réguliers (et les répertoires vides). Il vous faut connaitre l'arborescence du système de fichier pour pouvoir vous y déplacer. Une partie de cette arborescence est représentée ci-contre. **On lit de haut en bas**.

- Le répertoire "le plus haut" du système de fichier est le répertoire **racine** est représenté par la barre oblique ou *slash* notée `/` .
- Le répertoire courant est noté avec un point `.` .
- Le répertoire "au-dessus" appelé "parent" est noté avec deux points `..` . `home` est le répertoire parent de `alan`. Par convention, le répertoire père de la racine est la racine elle-même.
- Vous pouvez indiquer un chemin dans cette arborescence en séparant les dossiers par le caractère `/`. Le chemin du répertoire racine `/` au répertoire `Documents` de rick s'écrit:

  /home/rick/Documents



Un chemin partant de la racine est appelé **chemin absolu**. S'il part du répertoire courant, il est appelé chemin **relatif**.

Voici une description de quelques répertoires systèmes:

/ :racine, répertoire racine entrée du système de fichier

/bin : binaires, exécutables des commandes essentielles disponibles pour tous les utilisateurs (ex: cd, cat, ls...)

/dev : périphérique, fichiers spéciaux des périphériques

/etc : configuration, fichiers de configuration au format textuel de plusieurs programmes et services du système

/home : maison, répertoires personnels des utilisateurs

/lib : bibliothèques, bibliothèques partagées essentielles et modules du noyau

/media : media, contient les points de montages pour les médias amovibles

/root :root, répertoire personnel du super-utilisateur

/tmp :temporaire, fichiers temporaires des applications

/usr :ressources système, données en lecture seule par les utilisateurs.

/var :variable, données variables et diverses

0.3.3 Le Shell

Motivation

Le shell n'est pas un outil spécifique à GNU/Linux mais c'est le shell appelé *shell bash* du projet **GNU** que nous utiliserons dans ce cours. Il est devenu le shell par défaut des sytème *Unix* et on le trouve également sous *Windows* depuis sa version 10. Le fait d'utiliser un shell revient à effectuer une interaction avec l'ordinateur par un terminal: l'utilisateur entre des commandes à l'aide du clavier dans ce terminal (ou console).

De nos jours, où tous les ordinateurs ont la puissance nécessaire pour offrir une interface graphique, cela peut sembler archaïque d'utiliser un terminal[2]. En fait il n'en est rien, car l'interaction par un langage textuel offre une richesse et une flexibilité sans limite pour décrire des actions.

La gestion du système peut bien sûr être faite en mode graphique, mais la puissance de la ligne de commande est telle qu'il est nécessaire que vous en maitrisiez les bases, en outre il peut arriver de travailler sur une machine sans interface graphique.

Utilisation

L'invite de commande (ou *prompt*) dans un terminal se présente sous la forme:

```
rick@PC137:/home/rick/images$cp fichier1 fichier2
```

A gauche du $ se situent les informations d'identification et de localisation. A droite sont tapées les commandes *Shell*.

Une commande a la forme suivante où les mots sont séparés par un espace:

```
commande [-options] [argument1 argument2 etc.]
```

Dans l'exemple, l'utilisateur rick utilise le PC137, il est dans le répertoire `/home/rick/images`, ses droits sont normaux. Il a tapé la commande `cp fichier1 /travail/fichier2`. Un argument précédé d'un tiret est appelé option. Divers caractères sont utiles: `*` permet de remplacer n'importe quelle chaîne de caractère, `?` remplace n'importe quel caractère.

0.3.4 Commandes à connaitre:

`cd, mv, rm, mkdir, cp, rmdir, touch, pwd, ls, echo, cat, less, wc.`

`sudo` (substitute user do) donne les droits *administrateur*. Cela peut devenir pénible si votre session admin doit être importante: utilisée avec l'option -s, elle permet d'ouvrir un SHELL avec l'utilisateur root.

Exemples

1. Un terminal avec l'utilisateur *root*

   ```
   rick@Hal:~sudo -s
   [sudo] Mot de passe de rick :*****
   root@Hal:~#
   ```

   Observez le changement: le $ est devenu un #.

[2] ou console, c'est l'interface où s'exécute le *shell*

2. Un cas type où les droits root sont nécessaires est l'installation de logiciels. Si vous voulez par exemple installer un serveur LAMP (Linux Apache Mysql Php), vous aurez besoin d'installer Apache, MySql, Php. Pour rappel, Apache est un serveur Web, Mysql est un logiciel de base de donnée et PHP est un langage de programmation côté serveur. son installation s'effectue simplement en une ligne de commande sur un sytème basé sur Debian:

   ```
   rick@Hal:~#sudo apt install apache2 php libapache2-mod-php mysql-server php-mysql
   ```

   ou avec le système de base de données alternatif MariaDB :

   ```
   rick@Hal:~#sudo apt install apache2 php libapache2-mod-php mariadb-server php-mysql
   ```

3. Voici une commande (`cd` pour *change directory*) qui vous amènera dans le répertoire **html** tel qu'il est situé ci-dessus:

   ```
   rick@Hal:~$cd /var/www/html
   rick@Hal:$/var/www/html$
   ```

   Le tilde a été remplacé par le nouveau répertoire courant.

Exemple 2 Utilisation de la commande `man` pour obtenir la documentation de `ls` (affichage partiel):

```
rick@Hal:~$man ls
LS(1)                          Commandes                          LS(1)

NOM
       ls - Afficher le contenu de répertoires

SYNOPSIS
       ls [OPTION] ... [FICHIER] ...

DESCRIPTION
       Afficher les informations des FICHIERs (du répertoire courant par
       défaut).[...] Les entrées sont triées alphabétiquement.

       -a, --all
              inclure les entrées débutant par .
```

On comprend que cette commande liste les fichiers dans le répertoire courant. L'option `a` affiche tous les fichiers, y compris ceux cachés.

Exercice On utilise l'arborescence de fichier donnée plus haut.

1. Précisez le nom d'utilisateur, de la machine utilisée.
2. Recopier et modifier l'arborescence après l'exécution de chaque commande.
3. Pourquoi utilise-t-on `sudo` dans l'une des commandes?
4. Quel chemin absolu aurait-on pu écrire à la place du chemin relatif `../../alan/fichier2` ?
5. Que fait la commande `ls` est utilisée avec l'option `al` et l'argument `dossier1` ?

```
rick@Hal:~$rm Documents/cv/tom
rick@Hal:~$mkdir Travail
rick@Hal:~$touch Travail/fichier1
rick@Hal:~$cd Travail
```

```
rick@Hal:~/Travail$mkdir dossier1
rick@Hal:~/Travail$mv fichier1 dossier1
rick@Hal:~/Travail$sudo mv fichier1 ../../alan/fichier2
rick@Hal:~/Travail$ls -al dossier1
```

Attention: la commande `rm` peut causer de gros dégats! A manipuler avec précaution, et le moins possible avec le compte `root`.

0.3.5 Gestion des droits

Le système doit pouvoir distinguer chaque utilisateur, gérer la propriété des fichiers et partager les ressources. Chaque utilisateur aura donc un identifiant et un mot de passe de connexion, ainsi qu'un point d'entrée dans l'arborescence. Pour des raisons de sécurité et d'administration, tous le sutilisteurs ne peuvent pas faire la même chose. On distingue le super-utilisateur identifié par `root` qui peut tout faire et les utilisateurs normaux qui ne peuvent modifier que les fichiers de leur répertoire `/home/utilisateur` et à qui on peut attribuer un groupe pour le partage de ressource.

Représentation des droits

Pour un fichier, on distingue les droits d'accès pour le propriétaire, son groupe et le reste des utilisateurs.

Exemple: Ces droits sont visibles lorsqu'on utilise `ls -l`:

```
rick@Hal:~$ls -l
total 6
-rw-rw-r--  1 rick rick    2031506 janv.  5 23:35  algebre_boole.pdf
-rw-rw-r--  1 rick rick    6722945 mars  13 22:46  Algorithmique_1.mp4
-rw-rw-r--  1 rick rick       3820 avril 24  2020  all_simple_paths.py
drwxrwxr-x  3 rick rick       4096 mai    3  2020  apps
-rw-rw-r--  1 rick rick        626 juin  12  2019  arbo.txt
```

Les droits sont donnés dans cet ordre: en lecture: `r` pour *read*, `w` pour *write* ou `x` pour *execute*.

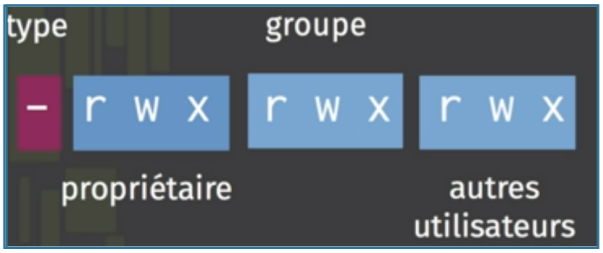


Figure 4: Droits sur un fichier

Le tout premier symbole indique le type: tiret bas `-` pour un fichier ou `d` pour répertoire.

Changer les droits

La commande `chmod` pour *change mode* permet de modifier les droits en utilisant la représentation suivante: Dans l'ordre on indique:

- l'utilisateur: `u` pour le propriétaire, `g` pour les utilisateurs du groupe, `o` pour other, `a` pour toutes les catégories précédentes.
- `+` pour ajout ou `-` pour suppression des droits,
- r, w ou x.

Exemple:

```
rick@Hal:$/Lycee/NSI/Premiere/SHELL/Travail$ ls -l
total 0
-rw-rw-r-- 1 rick rick 0 avril 19 14:06 correction.pdf
rick@Hal:$/Lycee/NSI/Premiere/SHELL/Travail$ chmod o-r correction.pdf
rick@Hal:$/Lycee/NSI/Premiere/SHELL/Travail$ ls -l
total 0
-rw-rw---- 1 rick rick 0 avril 19 14:06 correction.pdf
rick@Hal:$/Lycee/NSI/Premiere/SHELL/Travail$
```

Le droit de lecture a été retiré aux utilisateurs autre que le propriétaire et son groupe. Pour ajouter les droits d'exécution au propriétaire et enlever le droit d'écriture au groupe:

```
rick@Hal:/Lycee/NSI/Premiere/SHELL/Travail$ chmod u+x, g-w correction.pdf
```

Version numérique Une version numérique des droits simplifie leur écriture et leur lecture, il suffit d'additionner les valeurs associées aux droits:

- r $\longrightarrow$ 4.
- w $\longrightarrow$ 2.
- x $\longrightarrow$ pour valeur 1.

			fichier régulier	répertoire
lecture	r	4	regarder le contenu	lister le contenu
écriture	w	2	modifier le contenu	ajouter ou supprimer un élément
exécution	x	1	exécuter	traverser

Figure 5: Droits sur fichiers et répertoires

Exemple: La commande `chmod 754 fichier.sh` donne tous les droits au propriétaire du fichier: la valeur 7 (soit 111_2)[3] donne tous les droits au propriétaire, la valeur 5 (101_2) donne les droits d'écriture et d'execution au groupe etc. Le tableau ci-dessous donne la correspondance.

-	r	w	x	r	-	x	r	-	-
	1	1	1	1	0	1	1	0	0
	7			5			4		
	4	2	1	4	2	1	4	2	

Remarque: la commande `chown utilisateur chemin_vers_fichier` permet de changer le propriétaire d'un fichier et éventuellement son groupe: `chown utilisateur:group chemin_vers_fichier`.[4]

0.4 Enchaîner les commandes

Un mécanisme simple permet d'enchainer les commandes: l'utilisation d'un *pipe* ou *tube* permet de rediriger la sortie de la dernière commande pour l'utiliser avec la seconde. Son symbole est | (Alt Gr+6).

[3] Réflechissez au masque binaire sur la représentation `rwx`
[4] Il existe aussi une commande `chgroup`

Exemple: La commande `grep` permet de **filtrer** les lignes d'un fichier ou d'une *sortie* à partir d'un mot clé. Le fichier `fstab` se situe dans `/etc/` , il permet de connaitre la liste des disques montés au démarrage du système, ainsi que les droits et les systèmes de fichiers qui y sont installés. Affichez ce fichier avec `cat`. Que réalise la commande suivante ?

```
pi@raspberry:~ $cat /etc/fstab | grep ext4
```

0.5 Exemple concret: installation d'un serveur LAMP

Pour tester vos pages HTML, vous avez simplement ouvert le fichier *html* avec un navigateur. Pour tester les scripts PHP, il faut un interpréteur PHP, *de préférence* intégré à un serveur web: nous utiliserons Apache. Vous pourrez ensuite dans un navigateur saisir l'adresse (l'URL) correspondant à l'emplacement de ce fichier sur le serveur pour en obtenir le résultat. N'utilisez que la ligne de commande pour les questions qui suivent:

Préparation de l'espace de travail

1. Créer le répertoire *Travail* dans votre dossier utilisateur. Vous y déplacer.
2. A l'intérieur de votre répertoire *Travail*, créer le répertoire *TP_Web*. Y créer un fichier (vide) *formulaire.php* et *resultat.php*.
3. Vérifier que votre serveur *Apache* fonctionne en tapant `http://localhost` dans le navigateur. En principe, vous avez effectué l'installation avec M. Beel, La page *Apache 2* doit s'afficher si tout fonctionne correctement.
4. S'il ne fonctionne pas, utiliser la commande donnée en exemple plus haut.

   Le répertoire par défaut du serveur web apache2 est `/var/www/html`. C'est à dire que vous devriez déposer vos fichier PHP dans ce répertoire pour que le serveur les interprète. Ce n'est pas très confortable puisqu' écrire dans ce répertoire nécessite les droits root. On pourrait changer les droits de ce répertoire avec la commande `chmod`, mais ce n'est pas très sécurisé. Nous allons donc créer un lien symbolique entre le répertoire *Travail* que vous avez créé et ce dossier:
5. Créez ce lien avec la commande suivante:

   ```
   rick@Hal:~$ln -s /home/rick/Travail/   /var/www/html/
   ```

   Faites attention au nom d'utilisateur.
6. Déplacez vous avec `cd` dans le répertoire var/www/html/ et affichez son contenu:

   ```
   rick@Hal:/var/www/html$ls -a
   .  ..  Maths-code Travail
   ```

   Vous devriez voir apparaitre votre lien dans le répertoire `/var/www/html` grâce à ce lien symbolique. Sur mon PC, on peut y voir le répertoire de `maths-code.fr`. Remarquez le point et les deux points qui symbolisent ce répertoire et le répertoire "au-dessus" de celui-ci: Observez le changement si vous tapez `cd ..`:

   ```
   rick@Hal:/var/www/html$ cd ..
   rick@Hal:/var/www$
   ```

   Vous pouvez maintenant travailler avec votre répertoire *Travail* dans le dossier `home`. Il vous suffit d'ouvrir l'adresse `localhost/Travail/` dans votre navigateur pour avoir accès à vos fichiers par le **serveur Apache**.

0.6 Structures algorithmiques(hors programme)

Le *Shell bash* est un langage interprété. Les structures algorithmiques de bases permettent de manipuler plus efficacement le système.

0.6.1 La boucle pour

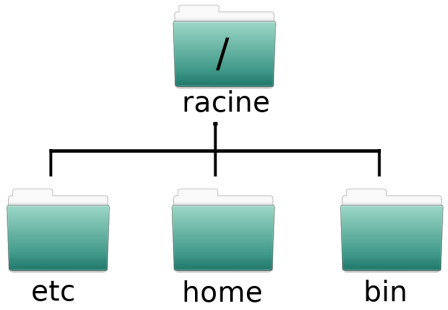
```
for i in 1 2 3 ;
do
    echo "$i";
done
1
2
3
```

On peut générer la liste de valeurs sur laquelle itérer.

0.7 Exercices

Exercice 1 L'utilisateur est . Reproduire l'arborescence ci-dessous et la compléter, puis taper ces commandes sur votre système.

1. `cd ~`
2. `mkdir Travail NSI`
3. `touch NSI/projet1.py`
4. `mkdir Travail/Python`
5. `cd Travail/Python`
6. `touch ex1.py ex2.py`
7. `cp ex1.py ../ex1_correction.py`
8. Avec la commande `adduser`, ajouter un utilisateur `alan`.
9. Copier le repertoire `Travail` dans le répertoire `alan`.
10. `mv NSI/projet1.py Travail/projet.py`
11. De votre répertoire courant, donner le chemin absolu et le chemin relatif pour se rendre dans le rendre répertoire `Travail` de l'utilisateur `alan`.



Exercice 2 On suppose que le répertoire utilisateur est vide. Décrire l'effet de chacune des commandes suivantes:

1. `cd ~`
2. `mkdir NSI`
3. `mkdir NSI/TP_SHELL`
4. `cd NSI/TP_SHELL`
5. `cd ..`
6. `ls`
7. `chmod u+rwx, g-rwx,o-rwx TP_SHELL`
8. Donner la version numérique pour les droits précédents.

Exercice 2b: Droits Dans cet exercice, on considère les répertoires `Travail` et `Travail/Projet`. Quelle commande permet d'obtenir les permissions ci-dessous ? Testez-les.

1. Le répertoire personnel possède tous les droits pour son utilisateur et seul le droit en exécution pour le groupe et les autres.
2. `Travail` et `Travail/Projet` possèdent tous les droits pour l'utilisateur, lecture et exécution pour le groupe et les autres.

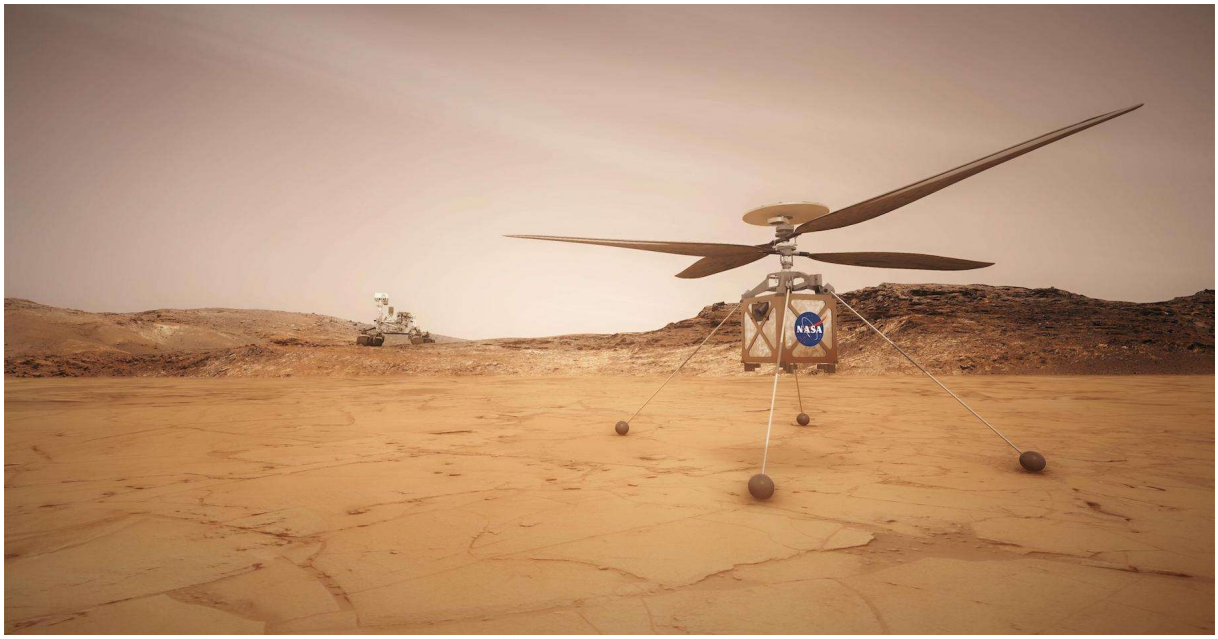


Figure 6: Ingenuity

3. Le fichier `Mandelbrot.py` du répertoire `Travail/Projet` possède les droits lecture et écriture pour l'utilisateur et lecture pour le groupe et les autres.
4. Le fichier `Fractales.py` du répertoire `Travail/Projet` possède les droits lecture et écriture pour l'utilisateur et aucun droit pour le groupe et les autres.

Exercice 3: Autres commandes en vrac

1. Utiliser la commande `man` pour obtenir le manuel de `grep, cat, find, seq, wc`.
2. Faire une recherche sur les opérateur | (pipe ou redirection) et > et >> (redirection de la sortie standard). Quelle commande ou combinaison de commande faut-il taper pour:
3. Afficher le **nombre** de dossier et répertoire contenus dans un dossier, par exemple votre répertoire courant.
4. Créer en une seule fois 5 repertoires numérotés de 1 à 5 ?
5. Obtenir la liste des groupes auquel l'utilisateur `pi` appartient ? **Note:** Le fichier contenant les groupes est le fichier `/etc/group`.
6. Enregistrer la liste précédente dans un fichier `liste_group` dans le répertoire courant ?
7. Obtenir le nombre de groupe auquel appartient l'utilisateur pi ?
8. Comment ne lister que les fichiers et dossiers créés en décembre du répertoire courant?

Exercice 4: Motif glob Les arguments d'une commande peuvent contenir des caractères spéciaux comme * et ? ou des ensembles de caractères entre crochets.

- * désigne n'importe quelle séquence de caractères.
- ? remplace un caractère quelconque.
- [^abcd] désigne n'importe quel caractère sauf a,b,c,d.

- [abcd] ou [a-d] désigne n'importe quelle lettre a,b,c,d.
  Ces motifs sont appelés motifs *glob* pour *global*.

Donner une suite de caractère de longueur au moins un reconnue par le motif *glob* suivant:

1. `*.jpg`
2. `[0-9]*[a-z]`
3. `*.???`
4. `*[^A-Z]*`
5. `?????`
6. `?*?`
7. `*[^.]???`

8. Créer des fichiers avec la commande `touch` et vérifier vos réponses. Si le fichier n'existe pas, vous obtenez une erreur.

   Exemple:

```
eliott@mrRobot:~$touch projet.py
eliott@mrRobot:~$ls *py
projet.py
eliott@mrRobot:~$ls *[a].jpg
ls: impossible d'accéder à *[a].jpg: Aucun fichier ou dossier de ce type
```

9. Quelle commande taper pour renommer tous les fichiers `.jpg` d'un dossier en `.JPG` ?
10. Alan a stocké ses photos dans le repértoire `Photos` de son dossier courant ∼. Quelle commande taper pour écrire dans un fichier `liste_photos` la liste de toutes les photos d'extension `.jpg` que contient ce dossier ?