TP 1 : prise en main

Tout exercice non marqué d'un ♣ est à terminer pour la semaine prochaine.

Environnement de bureau

Lorsque vous démarrez l'ordinateur, choisissez `Ubuntu` comme système d'exploitation.

Une fois votre nom d'utilisateur et mot de passe vérifiés, apparaît l'environnement de bureau. Il s'agit d'un ensemble de programmes permettant de manipuler l'ordinateur à travers une interface graphique qui fait analogie à un bureau, avec des icônes, des fenêtres et des menus.

Dans nos salles de TP, l'environnement de bureau est Linux Mint, tournant au dessus du système d'exploitation GNU/Linux, le tout étant distribué par Ubuntu.

Le menu principal (bouton en bas à gauche) permet de lancer des applications, d'accéder aux outils de paramétrage du système ou encore de fermer la session ou d'éteindre l'ordinateur (GNU/Linux, comme tout système moderne, doit être arrêté proprement et non pas en éteignant physiquement la machine).

La plupart des programmes ainsi que le menu principal disposent d'une entrée « aide » (*help*) permettant d'accéder à l'aide en ligne. Si vous êtes coincé(e), n'hésitez pas à la consulter.

Manipulation de fichiers et répertoires

Les informations stockées sur un ordinateur sont généralement organisées en un système de *fichiers* qui permet aux utilisateurs d'enregistrer leurs données et leurs programmes. Ces fichiers peuvent correspondre à divers types de données : textes, musique, programmes, etc. Certains fichiers particuliers, appelés *répertoires* (ou *dossiers*), servent de « boîtes » pouvant contenir d'autres fichiers. Du point de vue de l'utilisateur, un répertoire est un ensemble de fichiers et de sous-répertoires, désignés par des noms.

Ainsi, un peu plus tard dans la séance, votre répertoire `Info111` avec ses sous-répertoires et fichiers formeront l'arborescence suivante :

```
Info111
├── essai.txt
├── Semaine1
│   ├── essai2.txt
│   ├── essai3.txt
│   ├── laby-jupyter
│   ├── prise-en-main-jupyter.ipynb
│   ├── TD.pdf
│   └── TP.pdf
├── Semaine2
└── Semaine3
```

Exercice 1 (Gestionnaire de fichiers)**.**

(1) Ouvrez le gestionnaire de fichiers en cliquant sur l'icone « dossier personnel » du bureau.
(2) Quels répertoires et fichiers se trouvent dans votre répertoire personnel ?
(3) Allez dans le répertoire `Documents`.
(4) Quels répertoires et fichiers se trouvent dans votre répertoire Documents ?
(5) Remontez dans votre répertoire personnel.
(6) Créez un répertoire `Essai` dans votre répertoire personnel.
(7) Renommez `Essai` en `Info111`. Attention, respectez bien l'orthographe demandée : pas d'espace entre Info et 111, et une majuscule à Info.
(8) Fermez le gestionnaire de fichiers.

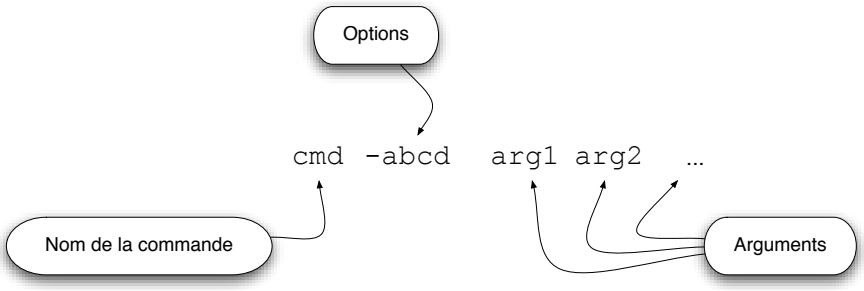
Exercice 2 (Éditeur de texte)**.**

(1) Lancez un éditeur de texte, par exemple `gedit`. Pour cela utilisez le menu principal (bouton en bas à gauche), où `gedit` se trouve dans la section `Accessoires`.
(2) Écrivez un court texte et enregistrez-le sous le nom `essai.txt` en faisant bien attention à l'endroit où vous l'enregistrez : on veut qu'il se trouve dans le répertoire `Info111` que vous avez créé dans votre répertoire personnel.
(3) Fermez `gedit`.
(4) Ouvrez le gestionnaire de fichiers en cliquant sur l'icone « répertoire personnel » du bureau.
(5) Allez dans le répertoire `Info111` et vérifiez que le nouveau fichier `essai.txt` s'y trouve bien.
(6) Ouvrir `essai.txt` en cliquant sur son icone et vérifiez le texte qu'il contient.
(7) Fermez toutes les fenêtres ouvertes (éditeur de texte et gestionnaire de fichiers).

Le terminal

Une autre façon d'interagir avec le système est d'utiliser un *terminal* dans lequel vous pouvez taper une à une des commandes (qui ne sont rien d'autre que des noms de programmes). Le programme avec lequel vous interagissez pour exécuter les commandes s'appelle le *shell*. À chaque fois, il vous indique qu'il est prêt en affichant en début de ligne une invite (ou *prompt*), ici le caractère dollar ($). Vous lui indiquez que vous avez fini de taper la commande suivante en appuyant sur la touche `Entrée` (`Enter`).

Pour affiner le comportement d'une commande, on peut lui adjoindre ce que l'on appelle des *options*, représentées par un tiret suivie d'une ou plusieurs lettre(s). On peut également utiliser certaines commandes avec des *arguments*. La forme générale d'une ligne de commande est souvent (mais pas toujours) la suivante :



On peut distinguer diverses sortes de commandes :
— des commandes « simples » qui affichent leur résultat directement dans le terminal (`ls`, `cd`, `cp`, ...). Ces commandes, puisqu'elles utilisent le terminal pour leurs interactions, ne peuvent pas être exécutées en dehors de celui-ci.
— des commandes qui lancent des applications (`firefox`, `gedit`, etc.). Ces programmes peuvent également être exécutés à l'aide d'une icône de l'interface graphique.

L'utilisation du terminal peut paraître rebutante au premier abord ; pourtant elle n'a rien d'obsolète, bien au contraire. Cela demande un apprentissage supplémentaire par rapport à une interface graphique, mais à la longue, c'est l'une des façons les plus productives de faire exécuter des tâches à un ordinateur, car elle permet d'automatiser simplement de nombreuses actions. Pour l'utilisateur régulier du système que vous êtes amenés à devenir, c'est un outil indispensable.

Exercice 3 (Manipulation de fichiers et répertoires)**.**
Vous allez maintenant apprendre quelques commandes de base qui vous permettront d'effectuer toutes les opérations utiles sur le système de fichiers (parcours, copie, déplacement, etc.) par l'intermédiaire du terminal. Vous constaterez que cela se révèle souvent bien plus rapide qu'utiliser l'interface graphique.

(1) Lancez un terminal. Pour cela vous pouvez utiliser le menu principal (bouton en bas à gauche) et cliquer sur « terminal ».

(2) Exécutez la commande `pwd` et observez le résultat.

> `pwd` (*print working directory*) : La commande `pwd` indique dans quel répertoire vous vous trouvez actuellement, autrement dit le répertoire courant.

À l'ouverture d'un nouveau terminal, le répertoire courant est toujours votre répertoire personnel.

(3) Affichez le contenu de votre répertoire personnel en utilisant `ls` (voir ci-dessous. Attention, le premier caractère de `ls` est une lettre, pas un chiffre).

`ls` (*list directory*) : La commande `ls` affiche le contenu du répertoire courant : ses fichiers et ses sous-répertoires.

(4) Allez dans le répertoire `Documents` en utilisant `cd` et le bon argument. Vous pouvez vous inspirer d'une ligne de l'exemple ci-dessous en l'adaptant à la question posée.

`cd` (*change directory*) : La commande `cd repertoireX` change le répertoire courant en le répertoire `repertoireX`. Elle permet ainsi de se déplacer parmi vos répertoires.

Conventions utiles :
— `..` : le répertoire parent
— `~` : le répertoire personnel
— `Images/Islande` : le sous-répertoire `Islande` du répertoire `Images`.

Exemple :

```
~ $ ls
Bureau Images Documents Musique Téléchargements Vidéo
~ $ cd Musique
~/Musique $ ls
~/Musique $ cd ..
~ $ cd Images
~/Images $ cd ../Vidéos
~/Vidéos $
```

(5) Exécutez la commande `pwd` et observez le résultat.

(6) Affichez dans le terminal le contenu du répertoire `Documents`.

(7) Retournez dans votre répertoire personnel (toujours avec le terminal !).

(8) Allez dans le répertoire `Info111`.

Astuce : Complétion automatique : pour aller plus vite, au lieu de taper le nom d'un fichier ou répertoire en entier, vous pouvez indiquer le début du nom et essayer de le compléter avec la touche `Tab` (tabulation, à gauche de la touche A). Si plusieurs fins sont possibles elles vous seront proposées. Ici, pour `Info111`, tapez `In` puis la touche `Tab`.

(9) Exécutez la commande `pwd` et observez le résultat.

(10) Affichez le contenu du répertoire `Info111`.

(11) Dans le répertoire `Info111`, utilisez la commande `mkdir` pour créer un sous-répertoire `Semaine1`, puis des sous-répertoires `Semaine2` et `Semaine3`.

`mkdir` (*make directory*) : La commande `mkdir répertoireX` crée un répertoire vide nommé `répertoireX`. Si un répertoire de même nom existe déjà à cet emplacement, vous obtiendrez un message d'erreur.

Astuce : en utilisant les flèches de déplacement du clavier, vous pouvez récupérer (flèche du haut, flèche du bas) et modifier (flèche de gauche, flèche de droite) les commandes précédentes.

(12) Dans le répertoire `Semaine1`, créez un sous-répertoire `laby-jupyter`.

(13) À l'aide de `ls` (et `cd` si besoin), vérifiez que les répertoires ont été créés au bon endroit.

(14) Affichez dans le terminal le contenu du fichier `essai.txt` en utilisant `less`.

`less` : La commande `less fichierX` affiche le contenu du fichier `fichierX`. Pour quitter, taper la touche Q.

Exercice 4 (Lancement d'applications et interruption de programme)**.**
Pour lancer une application, il suffit la plupart du temps de taper son nom dans le terminal.

(1) Allez dans le répertoire `Info111/Semaine1`.

(2) Lancez l'éditeur de texte `gedit` en tapant dans le terminal la ligne de commande : `gedit`

(3) Avec `gedit`, créez un nouveau fichier texte `essai2.txt` et enregistrez-le dans le répertoire `Semaine1` (qui se trouve dans le répertoire `Info111`).

(4) Fermez `gedit`.

(5) En utilisant `ls` (et `cd` si besoin), vérifiez que le fichier a bien été créé.

(6) Ouvrez à nouveau votre fichier texte `essai2.txt` en tapant dans le terminal la ligne de commande : `gedit essai2.txt`
Pour gagner du temps, pensez à utiliser la complétion automatique (tapez juste le début du nom du fichier et utilisez la touche `Tab` pour compléter, comme expliqué plus haut).

(7) Fermez `gedit`.

> `gedit` : La commande `gedit fichierX` ouvre à l'aide de `gedit` le fichier `fichierX` si celui-ci existe déjà, ou crée un nouveau fichier vide nommé `fichierX` et l'ouvre avec `gedit` si ce fichier n'existe pas encore.

(8) En tapant une seule ligne de commande, créez un fichier vide nommé `essai3.txt` et ouvrez-le avec `gedit`.

(9) Une fois l'éditeur ouvert, revenez sur le terminal (en laissant `gedit` ouvert) et tapez une commande (par exemple `ls`). Que se passe-t-il ?

(10) Toujours sur le terminal, gardez la touche `Ctrl` enfoncée puis pressez et relâchez la touche `C`. Que se passe-t-il ?

(11) Répétez les opérations en ajoutant cette fois le caractère `&` en fin de commande, c'est-à-dire ouvrez votre fichier texte `essai3.txt` en tapant dans le terminal la ligne de commande `gedit essai3.txt &`
Une fois l'éditeur ouvert, revenez sur le terminal et tapez une commande (par exemple `ls`). Voyez-vous la différence ?

Exercice 5 (Téléchargement d'archive et la commande info-111)**.**
Chaque semaine, nous fournissons une archive contenant les sujets de TD et TP accompagnés de fichiers de travail. En semaine 1, ce sera l'archive `Semaine1.zip` dont le contenu ira dans le répertoire `~/Info111/Semaine1`. Pour simplifier de telles opérations que nous effectuerons souvent dans le cadre de ce cours, mais aussi pour accéder à des logiciels installés de manière spécifique, nous avons mis à votre disposition une commande `info-111`.

(1) Exécutez la commande `info-111` et observez le résultat.

(2) Vous venez de lire la documentation de la commande `info-111`. Téléchargez l'archive de la semaine 1 en suivant les instructions données à ce sujet par cette documentation.

(3) Observez ce qui est affiché ; notez que `info-111` a fait appel aux commandes `wget` pour télécharger l'archive, `unzip` pour l'extraire, et `rm` pour la supprimer.

(4) Consultez le nouveau contenu de votre répertoire `Info111` ainsi que celui de ses sous-dossiers. Il devrait maintenant correspondre à celui représenté par une arborescence à la première page de l'énoncé.

Exercice 6 (Aller un peu plus loin ♣)**.**
Pour gagner du temps, pensez à utiliser la complétion automatique autant que possible !

(1) Renommez le fichier `essai.txt` en `essai1.txt` en utilisant `mv`.

> `mv` (*move*) : La commande `mv fichierX fichierY` déplace le fichier de chemin `fichierX` pour lui attribuer le chemin `fichierY`.
> Par exemple si `fichierX` est le nom d'un fichier du répertoire courant et `fichierY` est un nouveau nom de fichier, la commande change le nom du fichier. Si `fichierX` est le nom d'un fichier du répertoire courant et `fichierY` est le nom d'un sous-répertoire du répertoire courant, la commande déplace le fichier et le met dans le sous-répertoire.

(2) Déplacez le fichier `essai1.txt` pour le mettre dans le répertoire `Semaine1`.

(3) Copiez `essai1.txt` en un nouveau fichier `essai0.txt` en utilisant `cp`.

> `cp` (*copy*) : La commande `cp fic1 fic2` duplique le fichier de nom `fic1` en un nouveau fichier de nom `fic2`. Après exécution de la commande, `fic1` et `fic2` sont deux fichiers de même contenu mais complètement autonomes.

(4) Modifiez `essai0.txt` (en l'ouvrant avec `gedit` depuis le terminal).

(5) Vérifiez avec `less` que les deux fichiers sont maintenant différents.

(6) Effacez le fichier `essai3.txt` en utilisant `rm`.

> `rm` (*remove*) : La commande `rm fic` efface le fichier `fic`.

(7) Selon la configuration du système, `rm` ne demande pas de confirmation lorsque vous tentez de supprimer un fichier. Ceci peut se révéler assez dangereux. Trouvez l'option qui permet de demander confirmation.

> `man` (*manual*) : Le manuel en ligne pour toutes les commandes accessibles depuis le terminal. Il suffit de taper `man cmd` pour accéder à la description complète de la commande `cmd`.

Exercice 7 (Premier pas avec Jupyter)**.**
Lors des premières semaines de cours, nous allons travailler dans l'application web `Jupyter` qui permet de programmer interactivement dans de nombreux langages (`Python`, `C++`, ...), un peu comme une super calculatrice, et de rédiger des documents interactifs.

Aujourd'hui, nous allons prendre en main l'application, avant de l'utiliser pour jouer en programmant.

(1) Allez dans votre dossier Info111 : `cd ~/Info111`

(2) Lancez l'application Jupyter avec la commande suivante :
`info-111 jupyter notebook`

(3) Cette application ouvre une fenêtre dans votre navigateur web. Naviguez jusqu'au dossier `Semaine1` et sélectionnez la feuille de travail `prise-en-main-jupyter.ipynb`.

(4) Suivez les instructions qu'elle contient.

(5) Ne fermez pas le terminal !

Exercice 8 (Premiers programmes en jouant)**.**
Le jeu `laby` [1] propose plusieurs défis successifs ; pour chacun d'entre eux, le but est de guider pas à pas une fourmi hors d'un labyrinthe vers la sortie à l'aide. Chacun de ces défi sera l'occasion de découvrir ou manipuler un concept de programmation.

À titre expérimental, cette année nous allons utiliser `laby-jupyter` [2] une réimplantation de laby dans Jupyter.

(1) Relancez Jupyter si vous l'avez fermé (voir exercice précédent) ;
(2) Dans Jupyter, naviguer jusqu'à `Semaine1/laby-jupyter/notebooks` ;
(3) Chaque feuille de travail (hors `0a-prise-en-main-jupyter.ipynb`) correspond à un défi. Ouvrez les tour à tour dans l'ordre, en commençant par `0b.ipynb`. Suivre les instructions incluses.
(4) ♣ Reprendre tous les niveaux avec un autre langage de programmation (par exemple `Python`). `laby-jupyter` ne permettant pour le moment que `C++`, vous utiliser le programme `laby` original, en suivant les instructions de la page suivante.

Exercice 9 (À faire pour la semaine prochaine)**.**
Deux heures supplémentaires de `Laby` en salle libre-service (salle 215), ou depuis chez vous (instructions à venir).

1. `https://sgimenez.github.io/laby/`
2. `https://github.com/nthiery/laby-jupyter/`

Exercice 10 (Instructions pour le programme laby d'origine)**.**
Si vous souhaitez essayer d'autres langages que C++, ou s'il y a des difficultés techniques, vous pourrez être amenés à utiliser le programme laby d'origine. Voici les instructions :

(1) Lancez le jeu depuis le terminal avec la commande `laby`.

(2) Sélectionnez le langage de programmation « c ».

(3) Suivez les instructions (en haut à gauche) pour exécuter pas à pas la démonstration.

(4) Résolvez le maximum de niveaux en une heure ! Vous pouvez vous aider de l'API ci-dessous. Faites attention aux astuces données en bas à gauche pour résoudre les niveaux de façon intelligente. Enregistrez dans le répertoire `Semaine1` votre solution à chaque niveau en copiant-collant votre programme dans des fichiers nommés `niveau1a.cpp`, `niveau1b.cpp`, ...

API du jeu laby

```
////////////////////////////////////////////////////////////////////////
// Instructions
////////////////////////////////////////////////////////////////////////

avance();       // Fait un pas vers l'avant
droite();       // Fait pivoter la fourmi vers la droite
gauche();       // ou la gauche
prend();        // Prend le caillou situé sur la case devant la fourmi
pose();         // Pose le caillou devant la fourmi
dit("bonjour"); // Dit bonjour
regarde();      // Renvoie Vide, Caillou, Mur, Toile, Sortie, ou Inconnu
                // selon ce qui se trouve sur la case devant la fourmi

ouvre();        // Ouvre la porte située devant la fourmi (fin du niveau)

////////////////////////////////////////////////////////////////////////
// Constructions
////////////////////////////////////////////////////////////////////////

// Tant que la condition est respectée, répète les instructions
while(condition) {
    instructions;
}

// Si condition est vrai, exécute les instructions 1 sinon les instructions 2
if (condition) {
    instructions1;
} else {
    instructions2;
}

// Défini une fonction f
void f() {
    instructions;
}
```