Rappels d'informatique

1 Linux

Pour vous connecter avec le compte utilisateur ese, le mot de passe est : ese&1234. Ce compte dispose des droits d'administrateurs, soyez vigilents.

1.1 Création d'un compte

Les ordinateurs de la D060 ne sont pas connectés au LDAP. Vous devrez donc créer votre propre compte utilisateur. Pour cela, ouvrez un terminal et tapez la commande suivante :

```
sudo useradd -m -G sys,network,power,uucp,lp,wheel \
                -s /bin/zsh <nom_utilisateur>
sudo passwd <nom_utilisateur>
```

<nom_utilisateur> ne doit pas contenir d'espaces, ni d'accents, ni de caractères spéciaux.

Et comme toujours lorsque l'on présente une commande linux, les chevrons < > indiquent que vous devez remplacer le texte par ce qui est demandé. Ne les mettez pas dans la commande !

Créez un compte par membre du binôme. Vous pouvez vous déconnecter et vous reconnecter avec votre nouveau compte.

1.2 Commandes de base

Ouvrir un terminal. À l'aide de la commande `man`, de votre moteur de recherche favori ou simplement par l'expérimentation, trouver l'utilité des commandes suivantes : mkdir, cd, ls, touch, rm, cat, less, tail, pwd, cp, mv, chmod, chown, chgrp, ps, grep, top.
La plupart des commandes peuvent prendre des options, ce qui peut être nécessaire dans certains cas :

- Copier ou supprimer un dossier
- Afficher les fichiers cachés (sous linux, ce sont les fichiers commençant par un point)
- Mettre en forme la sortie pour différentes commandes

Le compte ese dispose des droits d'administrateur. Pour lancer une commande avec les droits d'administrateur, il faut ajouter `sudo` avant la commande.

Savoir utiliser le terminal linux permet d'aller beaucoup plus vite, mais l'apprentissage peut être long. Mais ça en vaut la peine. Faites-vous une *anti-sèche* papier.

1.3 L'éditeur de texte vim

La plupart du temps vous allez utiliser un éditeur de texte en mode graphique comme kate, ou un IDE comme STM32CubeIDE. Celà dit, il est souvent plus pratique et rapide d'utiliser un éditeur de texte en ligne de commande, comme `vim`. C'est le cas lorsque l'on veut modifier quelques lignes dans un fichier de configuration par exemple.

Parfois, l'interface graphique n'est simplement pas disponible, comme lorsque l'on travaille avec un linux embarqué. Dans ce cas ce genre d'éditeur est nécessaire.

Ouvrir un terminal, lancer l'éditeur de texte `vim`.

Pour taper du texte, il faut se mettre en mode *Insertion*. Pour cela, appuyer sur `i`. Pour sortir du mode insertion, appuyer sur <Echap>.

Voici quelques commandes utiles :

- `:q` puis Entrée : Quitter
- `:w` puis Entrée : Sauvegarde
- `:wq` puis Entrée : Sauvegarde et quitte
- `:q!` puis Entrée : Quitter sans sauvegarder
- `u` : annuler
- `i` : mode insertion
- `o` : ajouter une ligne
- `0` : ajouter une ligne au dessus
- `a` : mode insertion après le caractère courant
- `x` : supprimer un caractère
- `d` : supprimer plusieurs caractères
  - `d$` : supprime le contenu de la ligne
  - `dd` : supprime la ligne
  - `dw` : supprime un mot

Pour accéder au tutoriel officiel de vim, ouvrir un terminal et taper `vimtutor`. N'hésitez pas à y passer un peu de temps.

2 Gestion de versions avec Git

Git est un logiciel de gestion de versions. Son usage est très répandu et savoir s'en servir est quasiment devenu une nécessité. Le site `https://github.com` est un des sites d'hébergement de dépots `git` parmi les plus utilisés.

- Si ce n'est pas déjà fait, créez un compte (par étudiant!) sur le site `https://github.com`
- Une fois connecté, créez un nouveau *Repository*
  - Donnez-lui un nom et une description
  - Vous pouvez le passer en privé.
  - Ajoutez le fichier README
- Sur la page du *Repository*, cliquez sur `Settings > Collaborators`. Ajoutez votre binôme.
- Retournez dans l'onglet `Code`, puis cliquez sur le bouton vert `Code`, sélectionnez SSH et copiez l'adresse.

La suite s’effectue dans le terminal.

— Naviguez jusqu’à votre répertoire de travail (par exemple `~/Documents/votre_nom`)
— Clonez le projet :

```
git clone adresse_copiée_précédemment
```

— Entrez dans le dossier créé, et vérifiez le contenu du fichier README.md (par exemple avec vim)
— Ajoutez du texte dans ce fichier
— On peut observer les fichiers modifiés depuis le clone :

```
git status
```

— Pour pousser les modifications, il faut d’abord les ajouter au *commit*

```
git add README.md
```

  — Qu’est-ce qu’un *commit* ?
  — On peut aussi ajouter tout un dossier de la manière suivante :

```
git add .
```

— Testez avec `git status`
— Commitez les modifications :

```
git commit
```

  — Trouvez un moyen de vous en rappeler, il faudra le refaire pour chaque projet.
— Oh non ! vim s’ouvre ! Ajoutez un message. C’est important.
  — Sauvegargez/quittez (<Esc> :wq)
— Testez avec `git status`
— Pour l’instant les modifications ne sont sauvegardées qu’en local. Pour les pousser sur le serveur, il suffit de taper la commande suivante :

```
git push
```

— J’ai menti, ça génère une erreur. Il faut enregistre la clé SSH. Là je pars en réunion, débrouillez-vous (ou appelez-moi).

Ça fait beaucoup d’informations, mais c’est important. Prenez le temps de pratiquer.

3 Rappels de C

3.1 Boucles, algorithmes simples

1. Créez un tableau d'entiers dont la taille sera facilement configurable (avec un `#define` par exemple).
2. Remplir ce tableau avec des nombres aléatoire entre 0 et 99.
3. Écrire une fonction permettant de trouver le nombre maximal de cette liste, ainsi que son indice dans le tableau.
4. Même question pour trouver le nombre minimal.
5. Écrire une fonction permettant de trier le tableau du nombre le plus petit au nombre le plus grand.
6. Où ce situe le tableau dans la mémoire ? Où sont les différentes variables ?
7. Montrez votre travail à l'encadrant.

3.2 Manipulation de pointeurs

```c
#include <stdio.h>

int main (void) {
    int a;
    int * p;

    /* Parmi les deux lignes suivantes, laquelle est la bonne? Pourquoi?
     * Commentez la ligne fausse.
     */
    *p = a;
    p = &a;

    /* Que représente &p? Que représente p? Et *p? */

    /* Expliquer le comportement des lignes suivantes */
    printf("%d %d\n", a, *p);
    *p = 5;
    printf("%d %d\n", a, *p);
    a = 12;
    printf("%d %d\n", a, *p);

    /* Faites vérifier par l'encadrant */

    return 0;
}
```

3.3 Allocation de mémoire

Reprendre l'exercice 1.

1. Permettre à l'utilisateur, lors de l'exécution du programme, d'entrer la taille du tableau.
2. Où ce situe le tableau dans la mémoire ? Où sont les différentes variables ?
3. Montrez votre travail à l'encadrant.

3.4 Manipulation de fichiers

Créez (avec `vim`, par exemple) un fichier contenant une dizaine de nombres désordonnés.

1. Modifier le programme de l'exercice précédent pour qu'il aille lire les nombres dans le fichier au lieu de les générer aléatoirement.
2. Écrire le résultat dans un autre fichier
3. Montrez votre travail à l'encadrant.