TP n° 01 – Manipulation d'un OS

I Mise en route du TP

I.1 Démarrage de l'ordinateur

Les ordinateurs de la salle B 306 sont fixés au dos de l'écran et *ne* comportent *pas* d'interrupteur physique. Pour les démarrer, repérer l'alimentation secteur branchée au mur, la débrancher *sans la retirer complètement* et la rebrancher. Si au moment de rebrancher, la prise résiste, *ne pas* forcer : il y a un détrompeur à insérer.

I.2 Séquence de démarrage

Lorsqu'on démarre un ordinateur, un ensemble de processus sont mis en œuvre successivement et progressivement jusqu'à ce que la personne désirant utiliser l'ordinateur puisse le faire concrètement. Les principales étapes sont les suivantes :

1. Démarrage d'un petit programme, appelé *firmware*[1] et stocké dans une mémoire dite « morte » (mémoire ROM ou EEPROM) et qui permet notamment d'activer les *périphériques* (écran, clavier, éventuellement souris et périphériques audio). Le *firmware* indique également au *processeur* l'emplacement de la *mémoire de masse* (disque dur ou mémoire flash) dans lequel il doit aller lire les informations relatives au *système d'exploitation* (ou *operating system* « *OS* »).
   En TP Vous n'avez pas accès au firmware.
2. Si plusieurs systèmes d'exploitation sont présents, un autre petit programme, le *boot loader* permet parfois à l'utilisateur de choisir lequel il souhaite. Par exemple, certains ordinateurs de la salle informatique peuvent être utilisés sous Ubuntu GNU/Linux et Windows : après le premier écran de démarrage, un menu de systèmes d'exploitation pourrait être proposé à l'utilisateur : il disposerait alors de quelques secondes pour effectuer son choix ; par défaut, si aucune action n'est effectuée, c'est le système Ubuntu GNU/Linux qui est utilisé.
   En TP Cette fonctionnalité a été cachée, vous n'y avez pas accès directement.

I.3 Identification

Écran de connexion : le système d'exploitation choisi propose ensuite un écran de connexion, permettant de s'identifier à l'aide d'un nom d'utilisateur et, parfois, d'un mot de passe. Cette étape est cruciale puisqu'à chaque utilisateur ou groupe d'utilisateurs sont associés différents *droits* (lecture, écriture, exécution, possibilité de modifier certaines parties des paramètres du système d'exploitation, etc.).
En TP Normalement, il n'existe que deux utilisateurs prédéfinis sur les ordinateurs de la salle d'informatique : « professeur » et « élève ». Le compte « élève » s'identifie avec le mot de passe « eleve » : sélectionner ce compte, saisir le mot de passe et taper sur « Entrée ».

II Généralités sur les systèmes d'exploitation / système de fichiers

Une fois un utilisateur identifié la plupart des OS proposent deux types d'*interfaces système*, c'est-à-dire de façon d'utiliser l'ordinateur :

1. une *interface graphique* : c'est celle à laquelle tout le monde est habitué et qui propose des fenêtres, des menus déroulants, des icônes à cliquer, etc. et permettant de lancer des applications (navigateur web, gestionnaire de fichiers, python, etc.) ;
2. une *interface en mode texte* (ou *interprète de commande*) : elle permet de réaliser les mêmes tâches que l'interface graphique (et même plus !) à partir de commandes tapées au clavier. Bien que non ergonomique et d'apparence un peu désuète et il s'agit d'une méthode extrêmement robuste et efficace d'utiliser un ordinateur.

1. Il s'agit souvent de BIOS ou de UEFI.

III Interface en mode texte

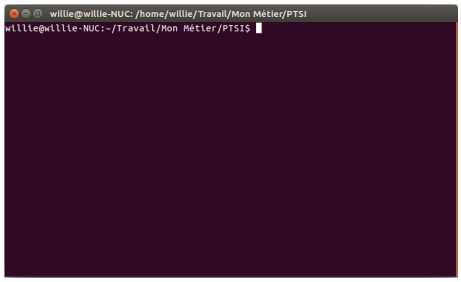


FIGURE 1 – Terminal

Sous Ubuntu GNU/Linux, vous pouvez accéder à l'émulation d'une interface en mode texte à l'aide d'un terminal (voir figure 1).
En TP Taper au clavier la combinaison « Ctrl + Alt + T ». Pour accéder à un terminal. En début de ligne, on voit le nom d'utilisateur connecté et le nom de l'ordinateur. Après le caractère « $ », il est possible de taper des commandes puis de taper sur « Entrée ». Essayer successivement les commandes suivantes :

```
date
whoami
```

III.1 Commandes de base

À l'aide de lignes de commande on peut également *naviguer dans le système de fichiers*, c'est-à-dire parcourir l'arborescence des différents dossiers qui structurent les données enregistrées dans la mémoire de masse. Les principales commandes sont : « `pwd` » pour savoir dans quel dossier le terminal travaille actuellement ; « `ls` » pour lister les dossiers et fichiers accessibles depuis le dossier actuellement ; « `ls -l` » utilise une option permettant de voir de façon ordonnée l'ensemble des informations sur les dossiers et fichiers (les dix premiers caractères regroupent les informations sur le type (dossier, lien, fichier) et sur les droits de lecture, d'écriture, d'exécution ; viennent ensuite : utilisateur propriétaire, groupe propriétaire, horodatage de la dernière modification, nom) ;
Si le premier caractère d'un élément affiché par `ls -l` est `d` alors il s'agit d'un sous-dossier du dossier actuel : on peut y faire travailler le terminal en utilisant la commande `cd` suivie du nom du sous-dossier.
En TP : se familiariser avec la navigation en mode texte dans le système de fichiers en :

1. repérant le nom d'un sous-dossier et taper la commande adéquate pour y faire travailler le terminal ;
2. tapant « `cd ..` » pour revenir au dossier immédiatement supérieur ;
3. tapant « `pwd` » pour connaître le dossier courant.

III.2 Python dans un terminal

Les terminaux permettent également de lancer une session interactive de programmation en python.
En TP : taper `python` dans votre terminal.
Le terminal affiche un certains nombres d'informations sur python puis un nouveau type de ligne de commande débutée par « > > > ». Vous êtes alors dans une session python, et seules les commandes connues de python sont alors actives et correctement interprétées. À tout moment, pour quitter cette session python, on peut taper `quit()` puis « Entrée » ou la combinaison `Ctrl + D`.
La programmation en python sera apprise tout au long de l'année. Dans un premier temps, nous nous servirons de python comme calculatrice.

En TP : taper les instructions suivantes toujours suivie de « Entrée » et observer les résultats...

```
a=1
type(a)
a
b=2
type(b)
b
a+b
a/b
a=1.
type(a)
a
b=2.
type(b)
b
a+b
a/b
```

On comprend assez vite qu'un langage de programmation obéit à des règles strictes qui ne correspondent pas toujours à l'intuition même pour une opération aussi simple qu'une division.

En TP : taper les instructions suivantes toujours suivie de « Entrée » et observer les résultats...

```
1-3*1/3.
1-1/3.
1-1/3.-1/3.
1-1/3.-1/3.-1/3.
```

Visiblement l'affichage du résultat des opérations précédentes « cache » quelque chose... Pour en être sûr, taper :

```
from decimal import Decimal
Decimal(1/3.)
```

Il va falloir apprendre et comprendre le cours d'Informatique pour interpréter ce comportement un peu particulier de la « calculatrice python [2] ».

Avant de passer à la partie suivante, quitter python en tapant `Ctrl + D` puis fermer le fenêtre du terminal.

IV Environnement intégré de développement : IDLE

Le logiciel IDLE propose les mêmes fonctionnalités que python dans un terminal et aussi d'autres fonctionnalités, comme un éditeur de script.

IV.1 Le shell

Dans la barre des tâches en haut de l'écran, chercher l'icône IDLE et cliquer dessus. Cela ouvre une fenêtre, nommée « shell », qui propose un interpréteur interactif.

Exercice 1 (Shell)**.** Reprendre l'exercice de la section III.2 et taper quelques lignes de commandes dans la fenêtre du shell. Vous devez obtenir les mêmes résultats qu'avec python dans un terminal.

IV.2 L'éditeur de script

Dans un interpréteur interactif, Python est interprété ligne par ligne. Pour exécuter plusieurs lignes de commandes, il faut utiliser un éditeur de fichier. IDLE propose aussi cette fonctionnalité.

Ouvrir un éditeur de script :

— en sélectionnant « New file » dans le menu « File » ;

— ou en tapant CTRL+N.

Une nouvelle fenêtre (qui est un éditeur de script) apparaît, nommée « Untitled », ce qui indique que le fichier dans lequel le script sera enregistré n'a pas de nom ou un nom générique, ce qui n'est pas une bonne pratique.

2. De la plupart des calculatrices numériques informatisées en réalité. Essayez « 1-1/3-1/3-1/3 » sur votre calculatrice pour en être sûr.

IV.3 Règles de sauvegarde d'un fichier

Avant même de commencer à rédiger un script, il faut le sauvegarder dans un fichier. Quelques règles indispensables :

1. Toujours vérifier **l'emplacement** de votre sauvegarde. Lors des TP d'informatique, vous devrez sauvegarder vos programmes sur votre clé USB ou éventuellement dans un dossier à votre nom dans « /home/eleve/Dossiers personnels »
   Si votre dossier personnel n'existe pas, créez-le : sur le bureau, repérez l'icône « Dossiers personnels », cliquez dessus ; naviguez jusque dans le répertoire « PTSI », faites un clic-droit puis choisissez « Créer un nouveau... dossier ». Entrez votre NOM suivi de votre Prénom, par exemple : `MARTIN Jean-Alexandre`.
2. Nommer de façon **pertinente** vos fichiers. Le nom du fichier doit contenir trois informations :
   (a) le nom du programme,
   (b) la version du programme (pour garder une trace des versions successives),
   (c) l'extension du fichier, séparée du nom du fichier par un « . » : elle indique le type de fichier (par exemple fichier texte, feuille de calcul de tableur, etc.) ; pour un script python, l'extension est « .py ».
3. Pas de ponctuation, d'espace dans le nom d'un fichier. Seul le underscore « _ » est accepté.
4. Le nom d'un fichier sera donc du type :

   NomProgramme_Version.py Ex : TP1_JoliDessin_V1.py

Enregistrer votre fichier avec un nom pertinent.

IV.4 Exécution d'un script

Exercice 2 (Exécution d'un script).

1. Dans l'éditeur de script ouvert précédemment, recopier le script suivant. On sera attentif à la ponctuation et à l'indentation.

```
1  from mpl_toolkits.mplot3d import Axes3D
2  from matplotlib import cm
3  from array import array
4  import matplotlib.pyplot as plt
5  import numpy as np
6  import csv
7  X1 = []
8  Y1 = []
9  Z1 = []
10 cr = csv.reader(open("FICHIER"))
11 for row in cr:
12     X1.append(float(row[0]))
13     Y1.append(float(row[1]))
14     Z1.append(float(row[2]))
15 X = np.asarray(X1)
16 Y = np.asarray(Y1)
17 Z = np.asarray(Z1)
18 fig = plt.figure()
19 ax = fig.gca(projection='3d')
20 surf = ax.scatter(X, Y, Z)
21 plt.show()
```

2. À la ligne **10**, le programme va récupérer les données présentes dans le fichier *data.csv*. Il faut indiquer à Python l'emplacement de ce fichier. Pour obtenir le chemin d'accès à un fichier, il faut parcourir l'arborescence du disque. Sous la plupart des systèmes d'exploitation, cela peut être fait en ouvrant un dossier quelconque puis en naviguant dans les différents dossiers et sous-dossiers. Sous Ubuntu, l'icône dédiée se trouve dans la barre latérale et prend l'aspect d'un petit dossier appelé « Dossier personnel ». Le fichier que nous utiliserons est nommé *data.csv* et est situé dans le dossier */home/eleve/Ressources/PTSI/TP/TP01*. Se placer dans ce dossier. Faire un clic-droit sur le fichier. Choisir « Propriétés » . Sélectionner et copier le chemin d'accès qui apparaît après « Emplacement ».

3. Dans votre script, à la place de « FICHIER », coller le chemin d'accès suivi de «/data.csv ».
4. Exécuter le script par les deux méthodes suivantes :
   (a) en cliquant sur « Run module » dans le menu « Run »,
   (b) ou en utilisant la touche fonction F5.

V Analyse d'un algorithme

La suite va consister en l'analyse de l'algorithme d'un test de personnalité. Le fichier contenant le script est disponible dans le dossier */home/eleve/Ressources/PTSI/TP/TP01*. Copier-le dans votre répertoire personnel.
Ouvrir le fichier dans un éditeur de script IDLE en :
— sélectionnant « Open » dans le le menu « File », puis en choisissant le fichier ;
— en tapant « CTRL+O », puis en choisissant le fichier.
La nouvelle fenêtre d'éditeur de script qui s'ouvre doit contenir :

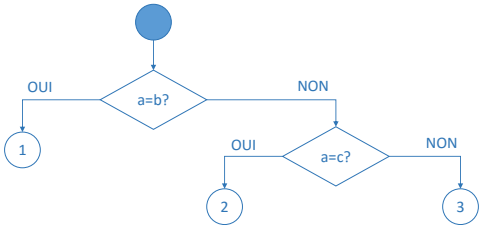
```
#!/usr/bin/python
#Test_personalite.py

questions1 = [("Votre talent tient dans vos", "super-pouvoirs", "ressources"),
("Un animal est pour vous un", "symbole", "truc pas pratique lorsqu'on part en vacances")]
result=''

for question in questions1:
    question_string = "%s:\n\ta. %s\n\tb. %s\n[a/b]:  " % (question[0], question[1], question[2])
    answer = input(question_string).lower()
    while answer not in ("a", "b"):
        print("Please choose A or B")
        answer = input(question_string).lower()
    result = result + answer

if result == 'aa':
    print('Vous etes Spiderman !')
elif  result == 'ab':
    print('Vous etes Superman !')
elif  result == 'ba':
    print('Vous etes Batman !')
else:
    print('Vous etes Iron man !')
```

Exercice 3 (Exécution d'un fichier).

1. Exécuter le script plusieurs fois en modifiant vos réponses afin de déterminer comment ces réponses influent sur le résultat.

Un organigramme permet de décrire le comportement d'un algorithme. Un exemple simple est présenté ci-dessous.

```
if a == b
   do 1
elif a == c
   do 2
else do 3
```



2. Décrire à l'aide de cette représentation le comportement du script précédent.
3. Dans le cas où le résultat est `'Vous etes Batman !'` quelle est la valeur de la variable `answer` ?
4. Compléter le questionnaire en ajoutant une question. On commencera par déterminer le nombre de nouveaux super-héros que cela fait apparaitre.
   Compléter ensuite le questionnaire en ajoutant une proposition « c » pour chaque question afin de faire apparaître de nouveaux super-héros (féminins si possible !).