I) Prise en main du système d'exploitation : clarté et efficace

A) Systèmes de fichiers et arborescence

Exercice 1
Visiter votre répertoire « Mes Documents », faire un rapide schéma de l'arborescence présente.

Remarque : La profondeur de l'arbre — c'est-à-dire le nombre de dossiers imbriqués — doit être adaptée :
- au nombre de fichiers prévus ;
- à la façon d'accéder aux fichiers (miniatures de photos, icônes de fichiers, ligne de commande).

Donc il faut *anticiper*. □

Exercice 2
Créer un répertoire **InformatiquePTSI**, avec un sous-répertoire **Python** qui contient un sous-répertoire **TP1 prise en main**

Si vous voulez éviter les ennuis, évitez les accents : normalement il n'y a pas de problème avec les accents. Normalement.

Exercice 3 (droits d'accès)
1) Quels sont les droits d'accès et les propriétaires du répertoire que vous venez de créer ?
2) Même question pour le répertoire CPGE contenu dans le répertoire Dossup (raccourci sur le bureau).
3) Pouvez-vous lire un fichier dans ce répertoire CPGE ? Pouvez-vous y créer un fichier ?

Pour accéder aux propriété d'un répertoire (ou d'un fichier) : clic droit, puis propriété (sous Windows les informations intéressantes sont toujours au fond du menu le plus lointain).

B) Raccourcis clavier : savoir être efficace

Pour le copier/coller, il faut toujours utiliser les raccourcis clavier :

Copier : **CTRL - C**	Coller : **CTRL - V**	Couper : **CTRL - X**

Ne pas hésiter à en user et abuser. Un copier/coller vaut mieux qu'une faute de frappe : il faut toujours éviter de retaper, et privilégier un copier/coller.

a) Comment sélectionner et se déplacer efficacement dans du texte ou un programme : les touches CTRL et Shift :

La combinaison **CTRL + flèche** ou **CTRL + backspace** a le même effet pour un mot que flèche ou backspace pour un caractère : testez ces commandes chez vous sur un fichier texte.
La touche **Shift** sert à sélectionner. Combinée à la touche **CTRL**, elle permet de sélectionner efficacement et rapidement du texte.
Hors d'un éditeur (par exemple sur une page web, ou dans un fichier pdf), vous pouvez utiliser le double clic glissé : au lieu de sélectionner caractère par caractère (et se concentrer pour viser la fin du mot), la souris sélectionne mot par mot.
Quand on recherche un motif (un mot, une phrase, une formule...) dans un texte, il faut toujours utiliser la fonction « rechercher » plutôt que ses yeux : plus rapide et moins fatigant... Idem, pour rechercher/remplacer, il y a toujours une fonction qui le fait systématiquement.

C) Sauvegarder son travail

Par définition, on ne pense à sauvegarder que quand l'ordinateur vient de planter : il faut toujours sauvegarder régulièrement. Les logiciels bien nés sauvegardent automatiquement. Le raccourci est quasi-systématiquement

CTRL-S

Il existe des logiciels qui permettent de garder l'historique de toutes les modifications successives de votre fichier. C'est ce qu'on appelle du « contrôle de version ». C'est hors programme, mais connaître son existence pourra vous servir plus tard.

II) Un environnement de développement : Spyder

Lancer le logiciel Spyder (raccourci sur le bureau).

A) Interpréteur

Prédire puis tester ce que vont faire les commandes suivantes :

```
2 + 2
3*5 #Ceci est un commentaire
5./3
5//3
2**10
1j*2j (Qui est j ?)
a = 3
a+a
```

```
b+1 (savoir lire un message d'erreur : que dit-il ?)
b=0
b=b+1
b=b+1
b=b+1
print(b)
2+*3
x=3.
```

Quelles lignes de code font des erreurs de syntaxe ?

b) Explorateur de variable :

Cliquer sur l'onglet « Explorateur de variable » juste au-dessus de la fenêtre de l'interpréteur. Que constate-t-on ? L'interpreteur sert à tester rapidement des tous petits morceaux de code. Taper

```
bla='Ceci est une phrase'
b=2*b
type(
```

Vous venez de découvrir l'onglet « aide », dont la fenêtre est située elle aussi en haut à droite.

B) Éditeur

Ouvrir un nouvel interpréteur Python : clic droit sur « Python 1 » en bas à droite.
Créer un nouveau fichier (menu fichier ou **CTRL-N**). Sauver immédiatement le fichier : enregistrer sous, puis aller dans « Mes documents », retrouver le répertoire créé en début de TP. Puis sauver le fichier sous un nom évocateur. Ensuite on se servira de **CTRL-S**.
Écrire dans la fenêtre de l'éditeur — à gauche — le programme suivant :

```
print('Bonjour !')
x=42
```

Exécuter le programme à l'aide du menu « Exécution » puis de l'entrée « Exécution » (remarquer que le raccourci est **F5**), en cliquant sur « Exécuter » dans la boîte de dialogue qui vient de s'ouvrir.
Rajouter `print(x)` à la fin du programme, dans l'éditeur, puis sauver et relancer.
Maintenant, exécuter à l'aide de **CTRL-F5**, c'est-à-dire en mode « debug » : pour passer à l'étape suivante, tapez **n** ou cliquez sur le 2ebouton bleu. Étudier l'explorateur de variable pendant ce temps là.

Exercice 4
Taper le programme suivant dans l'éditeur :

```
i=10
while i!= 0
i=1-i
print(i)
```

Normalement, vous devez voir s'afficher à l'écran les lignes suivantes :

```
i=10
while i!= 0:
    i=1-i
    print(i)
```

L'éditeur a rajouté des « : » à la fin de la ligne avec le `while`, puis a décalé les ligne : il a corrigé la syntaxe.
Avant de l'exécuter, regarder comment arrêter un programme devenu fou (le triangle orange en haut à droite de la fenêtre python).
Exécuter le programme. Tenter de l'arrêter (ou fermer Spyder et tout relancer).
L'exécuter pas à pas (avec le debugger). Où est le problème ? On pourra placer un point d'arrêt (double clic dans la marge de gauche de l'éditeur) en face d'une ligne bien choisie du programme.

Exercice 5
Taper dans l'éditeur et analyser le comportement du programme suivant (tester différentes valeurs de a) :

```
a = 1
if a == 1:
    print('a vaut bien 1')
else:
    print('a ne vaut pas 1')
```

III) Pour aller plus loin

Exercice 6
Écrire un programme qui détermine tous les nombres premiers plus petits que 2 millions.