TP n°16 - Travail préparatoire.

Le travail ci-dessous constitue vraiment le minimum à connaître pour l'analyse de données avec Python.

Avant toute chose, je vous invite à vous rafraîchir la mémoire en allant revoir le travail préparatoire du TP n°10.

Le format CSV (*comma-separated values*) est un format très simple permettant de stocker des informations tabulées dans un fichier texte. Un fichier au format CSV est conçu de la manière suivante :

— chaque ligne du fichier correspond à une ligne du tableau

— chaque ligne est formée de différents champs, séparés par un caractère appelé séparateur, généralement une virgule

Ce format est reconnu par la quasi-totalité des logiciels de calcul de type tableurs (`Libre Office Calc`, `Excel`, ...etc)

1. Vous avez normalement déjà installé le gestionnaire de paquets Python `pip3`. Sinon, tapez :

```
> sudo apt-get install python3
```

2. Normalement, la suite logicielle `Libre Office` est installée par défaut sur votre système Linux. Si ce n'est pas le cas, tapez la commande suivante pour l'installer et validez l'installation en tapant `O`.

```
> sudo apt-get install libreoffice
```

   La suite logicielle libre `Libre Office` contient le logiciel tableur `Libre Office Calc`, équivalent du logiciel `Excel` de Microsoft mais en version libre.

3. Visionnez cette vidéo qui explique comment utiliser la bibliothèque `csv` de Python qui permet de lire facilement des fichiers de données pour tableurs.

   https://www.youtube.com/watch?v=qwztvbt8bGk

4. Consultez la documentation du package `csv` :

   https://docs.python.org/fr/3.6/library/csv.html

Normalement, le module `csv` est inclus dans la bibliothèque standard Python sur les nouvelles versions, vous n'avez donc pas besoin d'installer ce module avec le gestionnaire de paquets Python `pip3`. Si toutefois vous commencez le TP et qu'un message indiquant que le module `csv` vous est renvoyé, faites ceci :

1. Vous avez normalement déjà installé le gestionnaire de paquets Python `pip3`. Sinon, tapez :

```
> sudo apt-get install python3-pip
```

2. Installez la bibliothèque `csv` en tapant la ligne de commande suivante dans un Terminal

```
> pip3 install python-csv
```

Exercice 1 (Lecture d'un tableau de valeurs en CSV).
Le fichier `valeurs_fonction.csv` disponible sur Moodle contient les informations correspondant à un tableau de valeurs d'une fonction mathématique, avec deux colonnes : la première correspond au choix de la variable d'entrée et la seconde à l'image associée par la fonction.

1. Ouvrir le fichier `valeurs_fonction.csv` avec `emacs` puis avec `Libre Office Calc` pour comprendre sa structure

2. En utilisant la bibliothèque `csv` de Python, écrire un script Python `trace_fonction_csv.py` qui trace le graphe (approximatif) de la fonction correspondant aux données du fichier `valeurs_fonction.csv`.

Exercice 2 (Lecture d'un fichier de données CSV).
Un exercice très proche de celui-ci a été proposé à la fin du TP11 mais a été codé en C, avec toutes les manipulations techniques sur les chaînes de caractères.
Je vous invite à refaire cet exercice de programmation à l'occasion, qui vous fera réviser les chaînes de caractères et les I/O en C.
L'exercice ici se fait en Python, en utilisant le package CSV.
Le fichier `MPII.csv` disponible sur Moodle, a été extrait de `Pronote` et contient les données relatives à chaque élève (nom, prénom, date de naissance, sexe et régime au lycée) au format CSV. Vous pouvez l'ouvrir dans `emacs` pour en comprendre la structure.

1. Afficher le début du fichier dans le Terminal puis ouvrez le avec `Libre Office Calc` pour comprendre sa structure
2. Écrire un script Python `anniversaire.py` qui prend en entrée un fichier au format CSV comme celui de la classe de MPII et un mois de l'année (sous la forme d'un chiffre entre 1 et 12 pour commencer) et qui affiche à l'écran le nom des personnes dont l'anniversaire a lieu ce mois-là, avec le jour de leur anniversaire. Pour récupérer les arguments de la ligne de commande, vous pourrez utiliser le tableau d'arguments `sys.argv` :

   https://stacklima.com/arguments-de-ligne-de-commande-en-python/

   Vous pourrez également utiliser la fonction `split` sur les chaînes de caractères :

   ma_chaine`.split(`*mon_caractere_separateur*`)`

   qui renvoie la liste des sous-chaînes après découpage de *ma_chaine* selon le caractère séparateur choisi.
3. Tester votre fonction sur les autres fichiers de données mis à disposition : `MPSI1.csv` et `MPSI2.csv`

Vous en avez fait l'expérience : cet exercice est beaucoup plus simple que l'exercice 3 du TP11 : Python nous facilite beaucoup la vie pour le traitement de données.

Morale de l'histoire : si on vous laisse la possibilité de choisir le langage de programmation, n'hésitez pas à utiliser le langage le plus approprié, le plus efficace pour chaque contexte de travail, en faisant communiquer vos codes par l'intermédiaire de fichiers. En général :

Le langage C doit être privilégié pour les codes et algorithmes lourds, demandant une optimisation au plus près de la machine.

Le langage OCaml est très intéressant pour coder des algorithmes proches des formulations mathématiques, en particulier des formulations récursives, ou pour créer un prototype de code utilisant des structures de données complexes.

Le langage Python possède de très nombreuses bibliothèques de fonctions très utiles pour l'analyse et la représentation des données : `numpy`, `scipy`, `csv`, `matplotlib`...etc

La bibliothèque `csv` pourra vous être utile pour vos épreuves orales, en vous permettant par exemple de définir vos propres formats de fichiers (cf colle 4 et corrigé), en vous permettant aussi de représenter les résultats donnés par vos algorithmes codés en C et/ou OCaml...

Vous montrerez également votre maîtrise des 3 langages (C, Python, OCaml) et votre capacité à utiliser chacun d'eux de manière optimale et dans le contexte le plus pertinent.