TP N°1: À la découverte de Python

Lors de cette séance nous allons faire connaissance avec le langage Python. La version utilisée sera Python 3. Nous n'utiliserons pas directement Python ; nous passerons par un environnement de développement intégré (en anglais IDE : interactive development environnement) basique appelé IDLE.
Pour lancer l'« Interpréteur Python » : Menu démarrer , tapez IDLE.
Dans cet interpréteur, le « shell », vous pouvez directement taper apèrs le symbole >>> (le prompt) des opérations ou des instructions, qui seront effectuées immédiatement.

Exercice 1: Utilisation de la console comme calculatrice

1. À l'aide de la console, calculer : $54+23$ $24-7$ 123×87 $34/5$
2. À quoi correspondent les opérations //,% et $**$?
3. Peut-on indiquer plusieurs calculs différents sur une même ligne ? Si oui, quel(s) séparateur(s) mettre ?

Exercice 2: À la découverte des Variables

Une variable est destinée à stocker une valeur en mémoire, elle possède un nom permettant l'accès à cette place en mémoire. La valeur d'une variable peut changer au cours du temps. L'action de donner une valeur à une variable s'appelle « l'affectation », et s'effectue avec le signe = .

1. Créer une variable x de valeur 3. Peut-on écrire l'égalité d'affectation dans le sens qu'on veut ?
2. Créer une variable y égale à $7\times x$, et afficher sa valeur. Modifier la valeur de x. Quel est l'effet de cette modification sur la valeur de y ?
3. À l'aide de la fonction `help`, comprendre ce que fait la fonction `id`, et la tester sur la variable `x`. De même pour la fonction `type`.
4. Rajouter 1 à x. Quel est l'effet sur le type et l'identifiant ? Même question en rajoutant 0.5.
5. Comprendre l'effet sur x des opérations +=, -=, *=, /=.
6. Échanger le contenu de deux variables x et y.

Exercice 3: Comment importer un module

1. Essayer de calculer $sin(1)$.
   Certains outils sont disponibles dans des « modules » spécialisés. C'est le cas d'un certain nombre de fonctions mathématiques usuelles, disponibles dans le module `math`. Il y a 3 façons de faire appel à une fonction définie dans un module. Nous les étudions, de la moins coûteuse à la plus coûteuse.
2. Pour pouvoir faire appel à des fonctions d'un module par exemple `math`, on utilise l'instruction `import math`. On peut alors utiliser la fonction `sinus` par exemple en écrivant `math.sin(x)`.
   - Déterminer $sin(1)$.
   - Calculer $\sqrt{1+ln^2(\pi)}$. (utilisez `help` pour lister les fonctions disponibles dans le module `math`).
3. Si on fait appel un grand nombre de fois à une fonction précise, on peut l'importer, ce qui évite par la suite de devoir faire référence au module `math` à chaque utilisation. Cela se fait (pour le `sinus`) avec l'instruction `from math import sin`.
   - Calculer $sin(1)+sin(2)+sin(3)+sin(4)+sin(5)+sin(6)$.
   - Calculer $\sqrt{1+\sqrt{1\sqrt{1+\sqrt{1+\sqrt{2}}}}}$.
4. Enfin, si on fait appel à un grand nombre de fonctions du module, on peut les importer toutes, en écrivant `from math import *`.
   - Tester l'égalité $cos(2\pi)=cos^2(\pi)+sin^2(\pi)$.

Script Python: Les programmes écrits dans l'interpréteur d'IDLE ne peuvent pas être sauvegardés : ce n'est pas très pratique si vous souhaitez relancer une même série de calculs avec des valeurs différentes pour les variables. Il s'agit donc d'écrire dans un fichier une succession d'instructions qui ne seront effectuées que lorsque nous lancerons l'exécution du programme.
Pour écrire et sauvegarder des programmes, la procédure générale est la suivante :

1. Pour créer un nouveau programme (script) Python, utiliser le menu file / New File.
2. Enregistrer-sous votre script sous le dossier D:/votre_groupe/numéro_tp/numéro_exercice.py
   Exemple : D:/mp1/tp1/ex1.py
   Remarque : lorsque vous saisissez le nom du fichier «ex1.py» , n'oubliez pas de spécifier l'extension du fichier «.py», cela permet à l'éditeur IDLE de mieux visiualiser votre code source.
3. Pour lancer l'exécution de votre programme, utiliser le menu Run / Run Module (F5).

Exercice 4: Premier programme

Nous quittons maintenant la console, pour écrire notre premier programme.
Écrire un programme affichant *Bonjour*.

Exercice 5: La commande `print`

On souhaite rédiger un programme élémentaire qui calcule le prix d'une commande de livres. Les trois valeurs suivantes sont représentées par les variables :

nbr = entier désignant le nombre de livres commandés

prix = prix unitaire d'un livre

reduc = pourcentage de réduction (compris entre 0 et 100) dont bénéficie le client

Le programme devra afficher le montant de la facture.

1. Trouver une expression algébrique représentant le montant de la facture en fonction des variables `nbr`, `prix` et `reduc`.
2. Ouvrir un nouveau fichier que vous sauvegardez sous le nom `factureLivres.py`.
3. Affecter les variables comme indiqué ci-dessus dans le cas d'une commande de 27 livres dont le prix unitaire est de $30,50\,dt$ pour un client bénéficiant de 5% de réduction.
4. Afficher le résultat dans l'interpréteur. Pour afficher un résultat, on peut faire appel à la fonction `print`. Pour afficher une phrase, vous pouvez, par exemple, utiliser la syntaxe : `print('Le montant de la facture est de',m,'dt')`. Vous afficherez la valeur tronquée de m à deux chiffres après la virgule (utilisez `round`). Pour en savoir plus sur la commande round, tapez `help(round)`.
5. Ajouter les commentaires nécessaires pour mieux expliquer le rôle de votre programme.
   Remarque : Pour faire figurer des commentaires dans un script, on les rédige précédés d'un symbole #. Ils apparaissent ainsi dans le script sans être interprétés.

Exercice 6: La commande `input`

La commande input interrompt l'exécution des instructions et attend que l'utilisateur réalise une entrée. Elle récupère les entrées sous forme d'une chaîne de caractères.
Ci-dessous, la commande `input` récupère l'age de l'utilisateur sous la forme d'une chaîne de caractères:

```
age = input('Quel âge avez-vous ?')
type(age) → str
```

Les commandes `int` et `float` convertissent une chaine de caractères respectivement en un entier ou un flottant. Et inversement, la commande `str` convertit un nombre en une chaîne de caractères.

```
age = input('Quel âge avez-vous ?')
age = int(age)
type(age) → int
```

Reprendre l'exercice 5 en utilisant la fonction `input`. Votre programme interroge l'utilisateur au sujet du nombre de livres, puis lui affiche le montant de sa facture.