Quelques scripts

Utiliser Python comme dans le TP précédent ne permet que de faire des opérations simples. De plus, il est impossible de corriger une erreur faite plus tôt.

On va donc écrire des scripts, qui se comportent au moment de l'édition comme un simple éditeur de texte, permettant de modifier n'importe quelle partie du code.

L'édition de script se fait dans la fenêtre de gauche. On peut enregister ces scripts, en faire un nouveau, *etc.*

On a ensuite plusieurs choix pour interpréter le code :

- La commande Alt-Enter exécute la partie sélectionnée du code, ou à défaut la ligne actuelle.
- La commande Ctrl-Enter permet d'exécuter une cellule, c'est-à-dire tout le code compris entre deux lignes commençant par ##. La commande Ctrl-Shift-Enter exécute une cellule, et avance le curseur à la suivante.
- La commande Ctrl-E exécute le fichier entier.

Attention, la mémoire de Python n'est pas réinitialisée après ces commandes. Pour effacer cette mémoire, on peut taper la commande Ctrl-K.

On peut aussi exécuter le fichier complet avec Ctrl-Shit-E, qui peut être vu comme un raccourci de Ctrl-K Ctrl-E.

Il est très important de commenter le code, c'est-à-dire d'ajouter des informations qui ne seront pas lues par l'interpréteur, expliquant ce que fait le code tapé. On écrit un commentaire d'une ligne en commençant cette ligne par #. On écrit un paragraphe de commentaire en l'encadrant de """.

Coder les scripts suivants :

Exercice 1. Écrire un script qui demande un entier positif à l'uilisateur, et qui renvoie True s'il est pair, False sinon.

Exercice 2. Python peut faire des graphiques avec une tortue. Il faut importer le module `turtle`, dont les commandes principales sont :

Commande	Résultat
reset()	Effacer le graphique
goto(x,y)	Aller au point de coordonnées (x, y)
forward(d)	Avancer de d pixels
backward(d)	Reculer de d pixels
up()	Lève le crayon
down()	Baisse le crayon
color(c)	Dessine en couleurs (c='red','blue',...)
left(α)	Tourne à gauche d'un angle de mesure α (en degré)
right(α)	Tourne à droite d'un angle de mesure α (en degré)

Dessiner un carré de côté 100, un triangle équilatéral de côté 100, puis les lettres B,C,P,S,T.

Exercice 3. On veut découper un disque de rayon R dans une plaque de métal, dans laquelle on veut enlever un disque de rayon $r < R$.

Écrire un script qui demande les nombres réels r et R, et qui renvoie la surface de la pièce obtenue (on ne vérifiera pas que $r < R$).

Exercice 4. Écrire un script qui demande les longueurs des deux côtés adjacents à l'angle rectangle d'un triangle rectangle, et qui renvoie la longueur de l'hypothénuse.

Écrire un script qui demande la longueur de l'hypothénuse d'un triangle rectangle, une mesure d'angle et la longueur du côté opposé à l'angle, et qui calcule la longueur du dernier côté.

Exercice 5. La formule pour calculer le montant à rembourser par an dans le cas d'un prêt de capital c, de taux t sur n ans est la suivante :

$$m = \frac{tc(1+t)^n}{(1+t)^n - 1}.$$

Écrire un script qui demande les valeurs de c, t et n et calcule la valeur de m.

Afficher aussi le remboursement mensuel, et le coût du crédit.

Exercice 6. On peut vouloir sauvegarder des choses en Python. Nous allons donc voir comment écrire dans un fichier auxiliaire. Commencer par créer un fichier texte dans le répertoire de travail.

Pour ouvrir un fichier, on utilise la commande suivante :

fichier = `open("nom_du_fichier.txt",'a')`

Le paramètre a peut être changé par

- a : pour ajouter des lignes de textes à la fin du fichier
- r : pour seulement lire le fichier
- w : pour effacer le fichier puis écrire dedans

Pour lire le fichier, on utilise la commande *`fichier`*`.read()`. C'est une variable qui contient la chaîne de caractères qui est dans le fichier, on pourra donc l'afficher avec `print`.

Pour écrire dans le fichier, on utilise la commande *`fichier`*`.write("`*`ce qu'on veut`*`")` (on note que le caractère $\backslash n$ dans une chaîne de caractère permet de faire un saut de ligne).

On n'oubliera pas de fermer le fichier à la fin avec la commande *`fichier`*`.close()`.

Reprendre le programme précédent, mais en sauvegardant les différentes valeurs de m simulées, associées aux données fournies par l'utilisateur.

On pourra tester notre programme en affichant le fichier après plusieurs essais.