

TD-TP n° 5

Cette feuille d'exercices suit le cours 5 et reprend aussi des notions des cours précédents.

Selon les indications données, les exercices sont à traiter

- en mode TD (papier, crayon),
- en mode TP (Thonny, jupyter-notebook)

Exercice 1. (TP) Retour sur les pièces d'or, argent, bronze et nickel

Reprenez le contexte de l'exercice 2 du TDTP 2.

1. Écrivez une fonction **nickel(O, A, B, N)** qui prend 4 paramètres pour représenter les quantités de pièces d'or, argent, bronze et nickel, et qui retourne le nombre de pièces de nickel équivalent.
2. Écrivez une fonction **decomposition(P)** qui prend en paramètre un nombre P (représentant une quantité de pièces de nickel) et qui retourne un tuple (O, A, B, N) qui traduit la quantité P en maximisant le nombre de pièces d'or, puis d'argent, puis de bronze.

Exercice 2. (TP) Retour les motifs géométriques - fonctions d'affichage sans return

Reprenez le contexte de l'exercice 8 du TDTP 3.

Toutes les fonctions que vous écrirez seront à tester dans un programme principal.

1. Écrivez une fonction **ligneEtoiles(n)** qui prend en paramètre un entier **n** et qui affiche une ligne de **n** caractères **"*"**.
2. Écrivez une fonction **ligne(n, c)** qui prend en paramètres un entier **n** et un caractère nommé **c** et qui affiche une ligne de **n** caractères semblables à **c**.
3. Écrivez une fonction **rectanglePlein(n, p, c)** qui prend en paramètres deux entiers **n** et **p**, et un caractère nommé **c** et qui affiche **p** lignes de **n** caractères semblables à **c**.
4. Écrivez une fonction **rectangleCreux(n, p, c)** qui prend en paramètres deux entiers **n** et **p**, et un caractère nommé **c** et qui affiche un rectangle de **p** lignes et **n** colonnes d'intérieur "creux" (ou encore, on n'affiche que le contour du rectangle comme ce-dessous) :

N'hésitez pas à appeler une fonction dans une autre fonction.

Exercice 3. (TP) Retour les motifs géométriques - fonctions avec return

On reprend les mêmes questions qu'à l'exercice précédent, mais cette fois, les fonctions n'affichent pas un motif, mais retournent une chaîne de caractères.

Vous pourrez vous servir du caractère de retour à la ligne représenté par **"\n"**.

Toutes les fonctions que vous écrirez seront à tester dans un programme principal.

Comment modifier ce programme principal ?

Exercice 4. (TP) voyelles et consonnes

1. Écrivez deux fonctions `voyelles(mot)` et `consonnes(mot)` qui retournent respectivement la liste des voyelles (des consonnes) du mot.
2. Écrivez une fonction **`composition(mot)`** qui prend en paramètre un mot (une chaîne de caractères) et qui retourne une liste composée de deux sous-listes : la première contient les voyelles du mot, la deuxième les consonnes.

Exercice 5. (TP) comparaison de croissances

Deux arbres sont plantés en même temps. Leur taille initiale est de 50 cm. Le premier arbre pousse de 15 cm par an. Le deuxième arbre voit sa taille augmenter de 10 % chaque année.

1. Écrivez une fonction **`taille1(n)`** qui prend en paramètre un nombre entier (représentant le nombre d'années écoulées depuis qu'on a planté l'arbre) et qui retourne la taille du premier arbre après ces **`n`** années.
2. Même question pour le deuxième arbre (fonction **`taille2`**).
3. Testez ces fonctions dans un programme principal pour savoir qui est le plus grand année après année. Vous pourrez utiliser la fonction native **`round`** pour avoir un arrondi (trouvez vous-même comment elle fonctionne). Voici un exemple d'exécution :

```
n = 0 : arbre 1 = 50 cm, arbre 2 = 50.0 cm
n = 1 : arbre 1 = 65 cm, arbre 2 = 55.0 cm
n = 2 : arbre 1 = 80 cm, arbre 2 = 60.5 cm
n = 3 : arbre 1 = 95 cm, arbre 2 = 66.55 cm
n = 4 : arbre 1 = 110 cm, arbre 2 = 73.21 cm
n = 5 : arbre 1 = 125 cm, arbre 2 = 80.53 cm
n = 6 : arbre 1 = 140 cm, arbre 2 = 88.58 cm
n = 7 : arbre 1 = 155 cm, arbre 2 = 97.44 cm
n = 8 : arbre 1 = 170 cm, arbre 2 = 107.18 cm
n = 9 : arbre 1 = 185 cm, arbre 2 = 117.9 cm
```

Exercice 6. (TP) comparaison de croissances v2

On reprend le thème de l'exercice précédent, mais cette fois, rien n'est fixe. La taille initiale des deux arbres est variable, ainsi que les valeurs qui caractérisent leur croissance respective.

1. Réécrivez la fonction **`taille1(n, t0, p)`** qui prend en paramètre un nombre entier **`n`** (les années de pousse), une taille initiale **`t0`** et une pousse annuelle **`p`** et qui retourne la taille du premier arbre après ces **`n`** années.
2. Même question pour le deuxième arbre : adaptez **`taille2`** pour qu'elle prenne en charge le nombre **`n`** d'années de pousse, la taille initiale **`t0`** et le taux de croissance **`tx`** (par exemple, pour 10%, on passera en argument la valeur 10).
3. Testez ces fonctions dans un programme principal pour savoir qui est le plus grand année après année, comme dans l'exercice précédent.
4. Écrivez une fonction **`plusGrand(n, t01, p, t02, tx)`** qui prend en charge le nombre d'années de pousse **`n`**, la taille **`t01`** et la pousse annuelle **`p`** de l'arbre 1, la taille initiale **`t02`** et le taux de croissance annuelle **`tx`** de l'arbre 2, et qui retourne un entier :

- 0 si les deux arbres ont la même taille après ces n années,
- 1 si c'est l'arbre 1 qui est le plus grand après ces n années,
- 2 si c'est l'arbre 2.

Voici un exemple d'exécution :

```
arbre 1 : t01 = 45 cm, p = 15 cm
arbre 2 : t02 = 50 cm, tx = 20 %

n = 0 : arbre 1 = 45 cm, arbre 2 = 50.0 cm : le plus grand est l'arbre 2
n = 1 : arbre 1 = 60 cm, arbre 2 = 60.0 cm : les deux arbres ont la même taille
n = 2 : arbre 1 = 75 cm, arbre 2 = 72.0 cm : le plus grand est l'arbre 1
n = 3 : arbre 1 = 90 cm, arbre 2 = 86.4 cm : le plus grand est l'arbre 1
n = 4 : arbre 1 = 105 cm, arbre 2 = 103.68 cm : le plus grand est l'arbre 1
n = 5 : arbre 1 = 120 cm, arbre 2 = 124.42 cm : le plus grand est l'arbre 2
n = 6 : arbre 1 = 135 cm, arbre 2 = 149.3 cm : le plus grand est l'arbre 2
n = 7 : arbre 1 = 150 cm, arbre 2 = 179.16 cm : le plus grand est l'arbre 2
n = 8 : arbre 1 = 165 cm, arbre 2 = 214.99 cm : le plus grand est l'arbre 2
n = 9 : arbre 1 = 180 cm, arbre 2 = 257.99 cm : le plus grand est l'arbre 2
```