De structure en structure

Bonus 03

Exercice 1

Dans cet exercice, les mots seront supposés être écrits uniquement avec les lettres `a` et `b`.

1. Écrire une fonction `liste_facteurs(mot,n) : (string*int) -> string list` qui renvoie le tableau des facteurs de `mot` de longueur `n` sans doublon.
2. Écrire une fonction `nb_facteurs(mot,n) : (string*int) -> int` qui renvoie le nombre de facteurs distincts de `mot` de longueur `n`.
3. Montrer que `mot` admet au plus 2^n facteurs distincts de longueur `n`.

Exercice 2

En soi, les dictionnaires forment une structure de données qu'on peut parfaitement faire à la main avec des tableaux : c'est l'objectif de l'exercice.
Vous êtes un marchand de meubles suédois, et vous voulez gérer votre catalogue.

1. Version dictionnaires

1. Créer un fichier `version_dictionnaire`.
2. Créer une variable globale `catalogue`, qui est un dictionnaire vide. Une variable globale est une variable qui existe en dehors de toute fonction : toute fonction que vous créerez y aura accès (du moins pourra lire son contenu). En Python, les fonctions ne peuvent pas toujours modifier le contenu d'une variable globale.
3. Écrire une fonction `ajouter_meuble(nom,prix) : (string*int) -> unit` qui rajoute au catalogue un meuble dont vous avez le nom et le prix. La clé est le nom du meuble et la valeur son prix. Par exemple, si vous exécutez `ajouter_meuble("table",250)`, la variable `catalogue` doit être modifiée de la façon suivante :

```
>>> catalogue["table"]
250
```

4. Modifier votre variable globale `catalogue` pour contenir par défaut une table à 250, un lit à 550 et une chaise à 60.
5. Écrire une fonction `moyenne_prix()` qui renvoie la moyenne des prix du magasin. Attention, la fonction ne prend pas `catalogue` en argument.
6. Écrire une fonction `liste_meubles()` qui renvoie le tableau avec les noms des meubles.
7. Écrire une fonction `imprimer_catalogue()` qui affiche chaque meuble et son prix, où chaque ligne de texte a la forme : `chaise coûte 60`
8. Lorsque vous vendez un meuble, il disparaît de vos données : écrire une fonction `vendre(nom)` qui supprime la case correspondant au meuble et affiche un texte de la forme `vendu ! argent gagné :` (à compléter avec l'argent gagné). Pour supprimer une case d'un dictionnaire en Python, la syntaxe est la suivante : `del dict[cle]`.

2. Version « tableaux »

1. Créer un fichier `version_tableaux`.
2. Créer deux variables globales `noms_meuble` et `prix_meuble`, qui sont deux tableaux vides.
3. Écrire une fonction `ajouter_meuble(nom, prix)` qui rajoute au catalogue un meuble dont vous avez le nom et le prix. Attention, vous devez vous assurer que les deux tableaux restent alignés !

4. Écrire une fonction `moyenne_prix()` qui renvoie la moyenne des prix du magasin.
5. Écrire une fonction `liste_meubles()` qui renvoie le tableau avec les noms des meubles.
6. Écrire une fonction `imprimer_catalogue()` qui affiche chaque meuble et son prix, où chaque ligne de texte a la forme : `chaise coûte 60`
7. Lorsque vous vendez un meuble, il disparaît de vos données : écrire une fonction `vendre(nom)` qui « supprime » la case correspondant au meuble et affiche un texte de la forme `vendu ! argent gagné :` (à compléter avec l'argent gagné). Vous ne pouvez pas supprimer une case d'un tableau : il faut donc recopier le contenu du tableau dans un autre sans recopier la valeur à vendre, et écraser les tableaux précédents.

Exercice 3 : limite des dictionnaires

On reprend ici l'exercice sur le Scrabble : on veut créer un « dictionnaire du Scrabble » qui, à chaque mot, associe son score en Python.

1. Créer une variable globale `score`, qui est un dictionnaire vide.
2. Ajouter à `score` les scores des mots `"dauw"` et `"bonbon"`.

Normalement, vous n'avez pas eu de difficultés à le faire.

M. Céargent, votre supérieur hiérarchique, n'aime pas les strings : il veut que les mots soient manipulés comme des tableaux. Ainsi, le mot « dauw » sera stocké sous le format `["d","a","u","w"]`.

3. Créer un nouveau dictionnaire vide `score2`.
4. Y ajouter le score du mot `["d","a","u","w"]` : vous devriez rencontrer un problème. Quelle raison Python donne-t-il pour refuser votre commande ?
5. Dans le cas où les mots sont stockés comme des tuples, Python acceptera-t-il votre commande ? Tester sur machine votre hypothèse.
6. Et si les clés sont des dictionnaires ?