Algo & Prog, avec Python
(L1-Sciences)
TP n° 5, Automne 2016

Tuples

Exercice 5.1 a) Soient a et b deux variables contenant des valeurs quelconques. Montrez que les *tuples* permettent d'échanger facilement les valeurs de a et b.

b) Donnez la manière standard d'échanger les valeurs de a et b sans utiliser de *tuple* (pensez à l'échange d'un verre de vin et d'un verre d'eau, il faut un troisième verre !).

c) *Optionnel astucieux*. Supposons que les valeurs de a et b soient des entiers. Donnez une troisième manière d'échanger les valeurs de a et b, sans tuples et sans variable supplémentaire.

Chaînes

Exercice 5.2 a) Programmez une fonction `masque(s,L)` prenant une liste de nombres `L` et une chaîne `s`, et retournant une nouvelle chaîne contenant les caractères dont les positions dans `s` figurent dans la liste `L`. Exemple :

```
masque('CAGCTACCTA',[2,5,3,8]) → 'GACT'
```

b) Programmez une fonction `choice(s)` retournant un caractère aléatoire de la chaîne `s`. En réalité, `choice` est une primitive Python du module `random` et qui fonctionne sur n'importe quelle *séquence*...

Listes

Exercice 5.3 a) Programmer avec une boucle une fonction `maxliste(L)` prenant une liste `L` de nombres réels, et retournant le couple contenant le plus grand élément et sa position :

```
maxliste([6,3,9,5,8,4]) → (9, 2)
```

b) Comment feriez-vous cela avec les primitives Python `max` et `index` ?

Exercice 5.4 a) Programmez la fonction `imin(L,i)` prenant une liste de nombres `L` et retournant l'indice du minimum de `L` à partir de l'indice `i` inclus. Exemple :

```
imin([3,2,4,8,5,3,1,9,7],3) → 6
```

b) Utilisez-la pour exécuter le *tri par insertion* d'une liste de nombres (cours page 20).

c) En vous inspirant du chronomètre utilisé dans le cours page 19, vérifiez que le comportement de ce tri par insertion d'une liste de longueur `n` est mauvais, en $O(n^2)$...

Exercice 5.5 Reprogrammez la fonction `diviseurs(n)` du cours 5 page 4 en une ligne, avec une *compréhension de listes* (sur le modèle de la fonction `premiers` page 17).

Exercices complémentaires

Exercice 5.6 Programmez une fonction `compacter(L)` prenant une liste d'entiers `L` et remplaçant toute suite d'éléments consécutifs `x,x,…,x` par le seul élément `x`. La liste `L` ne sera pas modifiée. Exemple : `compacter([3,3,3,8,5,5,5,7,7,5]) --> [3,8,5,7,5]`

N.B. La liste L ne sera pas modifiée *: c'est en général une assez mauvaise idée de procéder à une mutation sur un paramètre de fonction ! On fait plutôt muter les variables locales aux fonctions, ou les variables du toplevel. Imaginez qu'on vous prête le tableau de la Joconde et que vous écriviez dessus...*

Exercice 5.7 a) Modifiez la fonction `compacter` de l'exercice 5.6 de sorte qu'elle indique en plus la longueur de chaque sous-suite d'éléments consécutifs :

`compacter([3,3,3,8,5,5,5,7,7]) --> [[3,3],[8,1],[5,3],[7,2]]`

b) Modifiez la fonction `compacter` de l'exercice 5.6 de sorte qu'elle travaille **sur place**, sans construire une nouvelle liste, mais en supprimant les éléments répétés dans `L`, par *mutation* [utilisez la méthode `pop`]. Aucun résultat.

```
> L = [3,3,3,8,5,5,5,7,7]
> compacter(L)
> L
[3,8,5,7]
```

Exercice 5.8 *Nous poursuivons l'exercice complémentaire du TP3, dont nous supposons que vous avez programmé la fonction count_substring_match(s1,s2) dont vous allez vous inspirer.* Programmez une fonction `substring_match_exact(s1,s2)` retournant le **tuple** conternant les positions successives de l'apparition de `s2` dans `s1`. Par exemple :

`substring_match_exact("atgacatgcacaagtatgcat","atgc") → (5, 15)`

Exercice 5.9 Programmez une fonction `max2(L)` prenant une liste `L` de nombres entiers de longueur ≥ 2, et retournant le couple [sous forme de *tuple*] des deux plus grands éléments de `L`, le maximum étant en première position. Exemple : `max2([6,3,7,1,5,8,2]) → (8,7)`. Attention : on ne veut pas que la liste `L` soit modifiée après avoir évalué `max2(L)`.

Algo & Prog, avec Python
(L1-Sciences)
TD n° 5, Automne 2016

Exercice 5.0 Programmez une fonction `echanger(a,b)` sans résultat, de telle sorte que les valeurs de a et b soient échangées après exécution de cette fonction :

```
> a = 2 ; b = 3
> echanger(a,b)
> print('a =',a,' et b =',b)
a = 3 et b = 2
```

Exercice 5.1 **L'algorithme d'Euclide** pour calculer le PGCD de deux entiers a et b ≥ 0 consiste à appliquer les deux règles suivantes :

- si b = 0, le PGCD de a et de b est a
- sinon, le PGCD de a et b est le même que celui de b et du reste de la division de a par b

a) Calculez le PGCD de 8 et 12 par cette méthode.
b) Programmez par récurrence la fonction `pgcd(a,b)`.
c) Programmez cette fonction de manière itérative.

Exercice 5.2 **L'algorithme de Bezout** prolonge l'algorithme d'Euclide. Il dit qu'il est possible d'écrire le PGCD g de a et b comme combinaison linéaire de a et b à coefficients entiers : il existe u et v (non uniques) tels que g = a*u + b*v. Par exemple 4 = 8*(-1) + 12*1.
a) On se propose de programmer une fonction `bezout(a,b)` retournant un triplet `(g,u,v)`. Montrez que si l'on sait calculer `bezout(b,a % b)` alors on peut en déduire `bezout(a,b)`. Programmez par récurrence la fonction `bezout(a,b)`. Testez-la sur `bezout(8,12)`.
b) Programmez une fonction `aff_bezout(a,b)` affichant la décomposition g = a*u + b*v.

Exercice 5.3 En parcourant la documentation en ligne de Python 3, illustrez rapidement l'utilisation de `max`, `min`, `index`, `count`, `sum`. Comment pourrait-on simuler par des fonctions `max`, `index` et `count` (sur les listes par exemple) si elles n'existaient pas ?

Listes

Exercice 5.4 Programmez une fonction `nomul(k,a,b)` retournant la liste des entiers non multiples de `k` dans l'intervalle `[a,b]`. Ex : `nomul(3,1,10)` → `[1,2,4,5,7,8,10]`.

Exercice 5.5 a) Programmez la fonction `iota(n,x,s)` retournant la liste de longueur `n` contenant les nombres débutant par `x` et espacés de `s`. Par exemple :

```
> iota(5,2,3)          # j'en veux 5 à partir de 2 espacés de 3
[2,5,8,11,14]
```

b) Modifiez la même fonction pour que x et s soient **optionnels** : x vaudra 0 et s vaudra 1 par défaut. Par exemple : `iota(5)` → `[0,1,2,3,4]`, `iota(5,3)` → `[3,4,5,6,7]`. Cherchez sur Internet[1] ou demandez à votre enseignant comment faire, ceci est un complément de cours !

[1] Demandez par exemple à Google : « arguments optionnels python 3 ».