Université de Lille — FACULTÉ DES SCIENCES ET TECHNOLOGIES — Département Informatique

L1 MASS/SESI/PEIP - 2019–2020

Algorithmes et Programmation 1

DS2 - 25/06/2020

durée 2h

Ce devoir contient deux exercices indépendants. Indiquez votre parcours et le numéro de votre groupe, ainsi que votre nom et prénom

— en tête du fichier que vous remettrez si vous rédigez vos réponses sur ordinateur ;

— en tête de chacune des feuilles si vous composez sur papier.

Sauf mention explicite du contraire, il n'est pas attendu de documentation complète pour les fonctions demandées mais uniquement les **conditions d'utilisation (CU)**. En revanche, toute autre fonction non explicitement demandée, mais dont vous estimez avoir besoin, doit être documentée.

En cas de besoin, vous pouvez poser toute question relative à ce sujet par courrier électronique à l'adresse sos-ap1@univ-lille.fr

Exercice 1 *Questions diverses*

Dans cet exercice, il n'est pas demandé de réaliser des fonctions. Vous répondrez à chacune des questions en donnant une séquence d'instructions permettant de réaliser ce qui est demandé (le plus souvent une seule instruction suffit).

Question 1 Étant donnée une chaîne de caractères `s`, construisez la liste des chaînes extraites de `s` en découpant `s` selon le caractère `'.'`.

Question 2 Étant donnée une liste de chaînes de caractères `l_chaines`, construisez la chaîne obtenue en concaténant toutes les chaînes de cette liste en intercalant entre ces chaines le caractère `'/'`.

Question 3 Étant donné un entier `n`, construisez une liste de tous les carrés des entiers impairs compris entre 0 et `n` exclus.

Question 4 Étant donnée une liste d'entiers `l_entiers`, construisez une liste de tous les entiers de cette liste divisibles par 3 ou divisibles par 7.

Question 5 Étant donnée une liste de nombres, `liste`, construisez une liste de tous les couples (x, x^3) avec x éléments de `liste`. Par exemple, avec la liste `[1, 2, 3]` on doit obtenir la liste

`[(1, 1), (2, 8), (3, 27)]`.

Question 6 Étant donnée une liste `liste`, construisez une liste de tous les triplets `(x, y, z)`, `x`, `y`, `z` étant trois éléments consécutifs de `liste`. Par exemple, avec la liste `[1, 2, 3, 4]` on doit obtenir la liste

`[(1, 2, 3), (2, 3, 4)]`.

Question 7 Étant donnée deux listes `liste1` et `liste2`, construisez la liste de tous les couples `(x, y)`, `x` étant un élément quelconque de `liste1` et `y` étant un élément quelconque de `liste2`. Par exemple, avec les listes `liste1=[1, 2, 3]` et `liste2=['a', 'b', 'c']` on doit obtenir la liste

`[(1, 'a'), (1, 'b'), (1, 'c'), (2, 'a'), (2, 'b'), (2, 'c'), (3, 'a'), (3, 'b'), (3, 'c')]`.

Supposons donnée une liste d'informations sur les pays du monde dans laquelle ces informations sont présentées sous la forme (sigle, nom, superficie, population, continent) dont voici le début :

```
liste_infos_pays = [('AD', 'Andorra', 468.0, 84000, 'EU'),
                    ('AE', 'United Arab Emirates', 82880.0, 4975593, 'AS'),
                    ('AF', 'Afghanistan', 647500.0, 29121286, 'AS'),
                    ...
                   ]
```

Question 8 Construisez un dictionnaire dont les clés sont les noms des pays de la liste `liste_infos_pays` et les valeurs associées sont les tuples (sigle, superficie, population, continent) correspondant. Vous nommerez `dico_infos_pays` ce dictionnaire.

Question 9 Construisez l'ensemble des continents qui figurent dans le dictionnaire `dico_infos_pays`.

Question 10 Calculez la superficie totale obtenue en cumulant les superficies de tous les pays de ce dictionnaire.

Question 11 Comptez le nombre de pays dont la superficie est supérieure où égale à 82880, 0.

Exercice 2 *Calcul de racines carrées entières*

Le but de cet exercice est d'étudier deux algorithmes de calcul de la *racine carrée entière* d'un nombre entier naturel n, c'est-à-dire l'unique entier naturel r tel que

$$r^2 \leq n < (r+1)^2,$$

autrement dit l'entier $r = \lfloor \sqrt{n} \rfloor$. La table 1 donne la racine carrée entière de quelques nombres.

n	0	1	2	3	4	9	10	50	100	200
r	0	1	1	1	2	3	3	7	10	14

TABLE 1 – Quelques racines carrée entières

En PYTHON, le calcul d'une telle racine peut aisément être obtenu en utilisant les fonctions `sqrt` et `floor` du module `math`.

```
>>> from math import floor, sqrt
>>> floor(sqrt(200))
14
```

Mais la fonction `sqrt` renvoie un nombre flottant qui n'est qu'une valeur approchée de la racine carrée du nombre voulu. Et combinée avec la fonction `floor`, il est possible d'obtenir des résultats faux comme le montre l'exemple suivant.

```
>>> x = 2**53 + 1
>>> n = x**2
>>> r = floor(sqrt(n))
>>> n - r**2, (r+1)**2 - n
(18014398509481985, 0)
```

Question 1 Expliquez en quoi la valeur obtenue pour la dernière expression de la session ci-dessus montre bien que le calcul de r à l'aide des deux fonctions prédéfinies `floor` et `sqrt` est faux.

Dans la suite, on s'interdit donc l'usage de ces deux fonctions, et plus généralement tout calcul avec des nombres flottants. Vous veillerez donc à ce que toutes vos expressions produisent bien des nombres entiers.

Il s'agit donc de programmer une fonction nommée `racine_carree` dont voici la spécification :

```
def racine_carree(n):
    '''
    :param n: (int) l'entier dont on cherche la racine carrée
    :return: (int) la partie entière de la racine carrée de n
             autrement dit l'entier r tel que r**2 <= n < (r + 1)**2
    :CU: n >= 0
    '''
```

Question 2 Ajoutez **deux** doctests à cette spécification :
— la première qui reproduit les exemples de la table 1 ;
— et la seconde qui permet de garantir en cas de succès de la doctest que la fonction est correcte pour les 1000 premiers nombres entiers naturels.

On va considérer deux algorithmes pour programmer cette fonction. Les deux algorithmes reposent principalement sur l'idée qu'il s'agit de chercher l'unique entier positif ou nul r tel que $r^2 \leq n < (r+1)^2$, en sachant d'une part que la fonction carré est strictement croissante sur $\mathbb{R}_+$, et d'autre part que cet entier est compris entre 0 et n, $0 \leq r \leq n$.

Méthode séquentielle : Sachant que pour tout entier naturel n, sa racine carrée entière r est comprise entre 0 et n, on peut chercher séquentiellement r en considérant les entiers successifs croissants à partir de 0, ou bien en parcourant les entiers successifs décroissants à partir de n.

Question 3 Donnez un argument en faveur de l'une ou l'autre de ces deux options.

Question 4 Programmez une première version de la fonction `racine_carree` qui effectue une recherche séquentielle avec l'option que vous avez choisie dans la question précédente. On rappelle que l'usage de la fonction `sqrt` n'est pas autorisé, mais que l'élévation au carré l'est.

Question 5 Exprimez en fonction de n le nombre d'élévations au carré que votre fonction effectue.

Méthode dichotomique : Cette méthode consiste à découper l'intervalle de recherche $[a, b[$, initialement égal à $[0, n+1[$ en deux intervalles de longueurs égales $[a, m[$ et $[m, b[$, et à poursuivre la recherche dans l'un ou l'autre de ces deux intervalles selon la situation de r par rapport à m.

Question 6 Programmez une seconde version de la fonction `racine_carree` qui effectue une recherche par la méthode dichotomique.

Question 7 Exprimez en fonction de n le nombre d'élévations au carré que votre seconde version fonction effectue.
