DS n° 03 – Initiation à l'algorithmie

- Faire tous les exercices dans un fichier NomPrenom.py à sauvegarder,
- mettre en commentaire l'exercice et la question traités (ex : # Exercice 1),
- ne pas oublier pas de commenter ce qui est fait dans votre code (ex : # Je crée une fonction pour calculer la racine d'un nombre),
- les deux parties sont indépendantes et peuvent être traitées dans l'ordre que vous voulez,
- il faut vérifier avant de partir que le code peut s'exécuter et qu'il affiche les résultats que vous attendez. Les lignes de code qui doivent s'exécuter sont décommentées.

Introduction

Vous avez reçu en cadeau un puzzle numérique, il s'agit d'un ensemble de `pieces` (sous la forme de photos) disponibles dans le dossier `/home/eleve/Ressources/PTSI/boite`.

L'objectif est de recomposer le puzzle à partir de toutes ces `pieces`.

Après avoir codé un algorithme de tri dans la **première partie**, il faudra dans une **seconde** classer la liste des noms des pièces dans l'ordre de leur placement sur le puzzle. Ensuite, dans une **troisième** partie, nous assemblerons ce puzzle. Dans la **quatrième** et dernière partie, nous mettrons le résultat sous la forme d'une image en niveaux de gris.

1 Tri d'une liste

On donne pour commencer ce début de script qui génère une liste `listeint` qui contient les entiers de 0 à 99 dans le désordre.

```
1  # -*- coding: utf-8 -*-
2
3  import random
4  listeint=list(range(100))
5  random.shuffle(listeint)
6  print(listeint)
7
8  def tri(liste):
9          ....
10
11 tri(listeint)
12 print(listeint)
```

Question 1 Compléter le code précédent en codant la fonction `tri` qui permet de classer les entiers de la liste `listeint` dans l'ordre croissant. N'importe quel algorithme de tri pourra être utilisé.

2 Liste des pièces

Le code suivant permet d'afficher, dans l'ordre alphabétique, la liste des pièces du puzzle contenues dans le dossier « /home/eleve/Ressources/PTSI/boite ». Le nom de chaque pièce est constitué de deux lettres puis d'un nombre à 4 chiffre qui permet de classer la position de la pièce dans le tableau.

Ainsi, la pièce dont le nombre est le plus petit se trouve en haut à gauche, la suivante en dessous,... Lorsque la première colonne du puzzle est complète, la pièce suivante se trouve à la première ligne de la deuxième colonne, etc... Le puzzle est constitué de 12 lignes et 12 colonnes.

```
1 from os import listdir
2
3 liste_pieces = listdir('/home/eleve/Ressources/PTSI/boite')
4 liste_pieces.sort()
5 print(liste_pieces)
```

Le premiers éléments de la liste, avant le tri, sont :
`['AO0728.jpg', 'AS0350.jpg', 'BB0287.jpg', 'BE0129.jpg', 'BH0832.jpg',...`

L'information pour le placement de la pièce se trouvant dans le chiffre inclus dans le nom du fichier, on se propose d'extraire cette information.

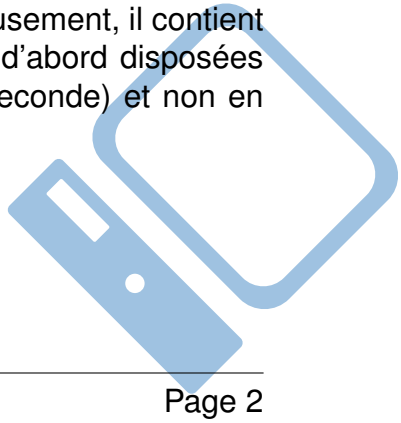
Question 2 Proposer un code permettant d'afficher les 4 caractères du nom, du premier élément de la liste `liste_pieces`, contenant ce chiffre. Le code doit afficher `0728`.

Question 3 A partir du résultat de cette question, proposer une fonction `tri_pieces`, dérivée de la fonction `tri` de la question 1, qui permet de trier les fichiers à partir de ces 4 caractères. Une fois triée, la liste `liste_pieces` commence alors par :
`['TM0010.jpg', 'XU0018.jpg', 'PB0025.jpg', 'TH0030.jpg',...`

3 Résolution du puzzle

Si vous n'avez pas réussi la partie précédente, vous pouvez utiliser les pièces du dossier « /home/eleve/Ressources/PTSI/boite2 ». Dans celui-ci le nom des fichiers est directement trié correctement. Vous pouvez directement utiliser le code au dessus de la question 2. Attention de modifier le code donné dans le sujet en conséquence `boite` devient `boite2`.

Le code suivant permet d'assembler les pièces du puzzle. Malheureusement, il contient des erreurs. En effet, vous pouvez constater que les pièces sont tout d'abord disposées en ligne (la première ligne doit être complétée avant de passer à la seconde) et non en colonnes.



```python
from PIL import Image
piece0 = Image.open('/home/eleve/Ressources/PTSI/boite/{}'.format(liste_pieces[0]))
lx,ly=piece0.size
dx,dy=12,12
image = Image.new('RGB', (lx*dx, ly*dy))
i=0
py=0
for y in range(dy):
    px=0
    for x in range(dx):
        nom_piece='/home/eleve/Ressources/PTSI/boite/{}'.format(liste_pieces[i])
        piece = Image.open(nom_piece)
        image.paste(piece, (px, py))
        i+=1
        px+=lx
    py+=ly

image.show()
```

Question 4 Recopier et modifier le code précédent afin d'afficher le puzzle en colonnes afin que l'image qui apparaisse soit correcte.

4 Traitement de l'image

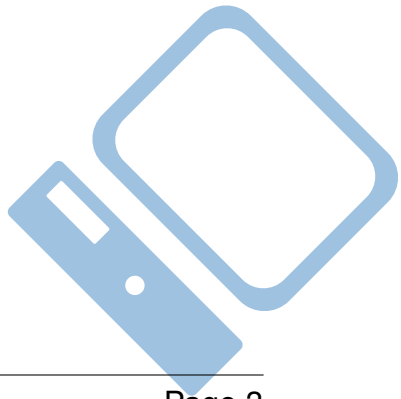
Quelque soit le résultat obtenu, sauvegarder l'image en remplaçant `image.show()` par `image.save('puzzle.png')`. Cela permet de générer le fichier `puzzle.png` à côté de votre script python. Vous utiliserez ce fichier pour la suite de l'exercice.

Si vous n'avez pas réussi à reconstituer le puzzle, vous pouvez pour la suite utiliser un puzzle déjà complet présent dans le fichier `puzzle_annee_derniere.png`.

Question 5 Convertir et afficher cette image en niveaux de gris grâce à la bibliothèque `matplotlib` comme vu en TP. On rappelle les coefficients à utiliser pour le niveau de gris en RGB : `[0.2125,0.7154,0.0721]`.

Remarque : Le dernière page du sujet doit servir de brouillon.

FIN



Question 1

```python
# -*- coding: utf-8 -*-
import random
listeint=list(range(100))
random.shuffle(listeint)
#print(listeint)

#ici on utilise un tri par sélection
def tri(liste):
    for i in range(len(liste)-1):
        min=liste[i]
        jmin=i
        for j in range(i, len(liste)):
            if liste[j]<min:
                jmin=j
                min=liste[j]
        liste[i],liste[jmin]=liste[jmin],liste[i]

tri(listeint)
print(listeint)
```

Question 2

```python
print(liste_pieces[0][2:6])
```

Question 3

```python
from os import listdir
liste_pieces = listdir('/home/eleve/Ressources/PTSI/boite')
print(liste_pieces)

def tri_pieces(liste):
    for i in range(len(liste)-1):
        min=liste[i][2:-4]
        jmin=i
        for j in range(i, len(liste)):
            if liste[j][2:-4]<min:
                jmin=j
                min=liste[j][2:-4]
        liste[i],liste[jmin]=liste[jmin],liste[i]

tri_pieces(liste_pieces)
print(liste_pieces)
```

Question 4

```
piece0 = Image.open('/home/eleve/Ressources/PTSI/boite/{}'.format(liste_pieces[0]))
lx,ly=piece0.size
dx,dy=12,12
image = Image.new('RGB', (lx*dx, ly*dy))
i=0
px=0
for x in range(dx):
    py=0
    for y in range(dy):
        nom_piece='/home/eleve/Ressources/PTSI/boite/{}'.format(liste_pieces[i])
        piece = Image.open(nom_piece)
        image.paste(piece, (px, py))
        i+=1
        py+=ly
    px+=lx

image.show()
```

Question 5

```
import matplotlib.image as img
import matplotlib.pyplot as plt
import numpy as np

image = img.imread('puzzle.png')
image = img.imread('puzzle_annee_derniere.png')
plt.imshow(image)
plt.show()

def convert_nb_evol(image):
    image_out=np.zeros(np.shape(image))
    coefficients=[0.2125,0.7154,0.0721]
    for j in range(3):
        for i in range(3):
            image_out[:,:,j]+=image[:,:,i]*coefficients[i]
    return image_out

plt.imshow(convert_nb_evol(image))
plt.show()
```