BACCALAUREAT

SESSION 2022

Épreuve de l'enseignement de spécialité

NUMERIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°39

DUREE DE L'EPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Le candidat doit traiter les 2 exercices.

EXERCICE 1 (4 points)

Écrire une fonction `moyenne` prenant en paramètres une liste d'entiers et qui renvoie la moyenne des valeurs de cette liste.

Exemple :

```
>>> moyenne([10,20,30,40,60,110])
45.0
```

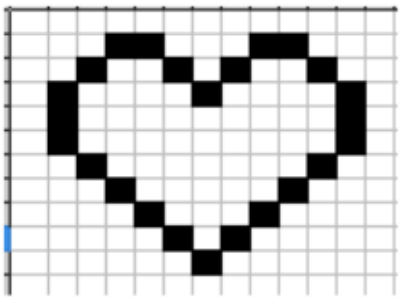
EXERCICE 2 (4 points)

On travaille sur des dessins en noir et blanc obtenu à partir de pixels noirs et blancs :

La figure « cœur » ci-contre va servir d'exemple.

On la représente par une grille de nombres, c'est-à-dire par une liste composée de sous-listes de mêmes longueurs.

Chaque sous-liste représentera donc une ligne du dessin.



Dans le code ci-dessous, la fonction `affiche` permet d'afficher le dessin. Les pixels noirs (1 dans la grille) seront représentés par le caractère `"*"` et les blancs (0 dans la grille) par deux espaces.

La fonction `zoomListe` prend en argument une liste `liste_depart` et un entier `k`. Elle renvoie une liste où chaque élément de `liste_depart` est dupliqué `k` fois.

La fonction `zoomDessin` prend en argument la grille `dessin` et renvoie une grille où toutes les lignes de `dessin` sont zoomées `k` fois et répétées `k` fois.
Soit le code ci-dessous :

```
coeur = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], \
         [0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0], \
         [0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0], \
         [0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0], \
         [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0], \
         [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0], \
         [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0], \
         [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0], \
         [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0], \
         [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0], \
         [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0], \
         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]

def affiche(dessin):
    ''' affichage d'une grille : les 1 sont représentés par
```

```
        des "*" , les 0 par deux espaces "  " '''
    for ligne in dessin:
        for col in ligne:
            if col == 1:
                print(" *",end="")
            else:
                print("  ",end="")
        print()


def zoomListe(liste_depart,k):
    '''renvoie une liste contenant k fois chaque
       élément de liste_depart'''
    liste_zoom = ...
    for elt in ... :
        for i in range(k):
            ...
    return liste_zoom

def zoomDessin(grille,k):
    '''renvoie une grille où les lignes sont zoomées k fois
       ET répétées k fois'''
    grille_zoom=[]
    for elt in grille:
        liste_zoom = ...
        for i in range(k):
            ... .append(...)
    return grille_zoom
```

Résultats à obtenir :

```
>>> affiche(coeur)
```

```
>>> affiche(zoomDessin(coeur,3))
```

```python
"""
Author: Pascal Padilla
Source: correction de l'exercice 1 du sujet 39 des épreuves pratiques NSI 2022

Remarques:
"""


def moyenne(tab: list) -> float:
    """ renvoie la moyenne de tab

    Exemples et tests:
    >>> moyenne([10, 20, 30, 40, 60, 110])
    45.0
    """
    nb_valeurs = len(tab)
    # BOUCLE: parcours du tableau pour calculer la somme des valeurs
    # -> invariant: somme est égal au cumul des valeurs parcourus jusqu'à présent
    somme = 0
    for val in tab:
        # maintient de l'invariant
        # mise à jour de la somme
        somme += val

    # fin BOUCLE: somme contient le cumul de toutes les valeurs du tableau

    # formule de la moyenne
    return somme / nb_valeurs


# vérification avec des assertions
assert moyenne([10, 20, 30, 40, 60, 110]) == 45.0

# vérification avec des affichages
print(moyenne([10, 20, 30, 40, 60, 110]))   # 45.0

# vérification avec doctest
from doctest import testmod
testmod()
```

```
"""
Author: Pascal Padilla
Source: correction de l'exercice 2 du sujet 39 des épreuves pratiques NSI 2022

Remarques:
"""

# un tableau à deux dimensions : un tableau de tableau !
# pour accéder à la 3ème ligne : coeur[2]
# pour accéder à la 5ème colonne de la 3ème ligne : coeur[2][4]
coeur = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], \
         [0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0], \
         [0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0], \
         [0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0], \
         [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0], \
         [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0], \
         [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0], \
         [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0], \
         [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0], \
         [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0], \
         [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0], \
         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]


def affiche(dessin):
    ''' affichage d'une grille : les 1 sont représentés par
        des "*" , les 0 par deux espaces "  " '''
    # parcours le dessin, ligne par ligne
    for ligne in dessin:
        # pour chaque ligne, colonne par colonne
        for col in ligne:
            # cas d'un pixel égal à '1'
            if col == 1:
                # afficher une ' *'
                print(" *",end="")
            else:
                # afficher des espaces
                print("  ",end="")
        print()


def zoomListe(liste_depart,k):
    '''renvoie une liste contenant k fois chaque
       élément de liste_depart'''
    # BOUCLE 1: parcours de la liste en dupliquant k fois chaque pixel
    # -> invariant: liste_zoom contient k fois chaque élément
    #               de la zone parcourue jusqu'à présent
    liste_zoom = []
    # -> condition d'arrêt: après le dernier pixel de la liste_depart
    for elt in liste_depart :
        # BOUCLE 2: duplication du pixel courant
        # -> invariant: le pixel a été dupliqué i fois
        # -> condition d'arrêt: après la boucle i <- k-1
        for i in range(k):
            # ajout d'une copie du pixel
            liste_zoom.append(elt)
        # fin BOUCLE 2
    # fin BOUCLE 1: tous les pixel ont été dupliqués
    return liste_zoom

def zoomDessin(grille,k):
    '''renvoie une grille où les lignes sont zoomées k fois
       ET répétées k fois'''
    # BOUCLE 1: grossit toutes les lignes (en largeur et en hauteur)
    # -> invariant: grille_zoom contient les lignes grossies en largeur
    #               et en hauteur de la zone parcourue de la grille
    grille_zoom=[]
    # -> condition d'arrêt: après la boucle traitant la dernière ligne
    for elt in grille:
        # BOUCLE 2: grossit en largeur et en hauteur la ligne courante
        # -> invariant: grille_zoom possède à la fin i copies du zoom
        #               de la ligne courante
        liste_zoom = zoomListe(elt, k)
```

```
        # -> condition d'arrêt: après le tour de boucle i <- k-1
        for i in range(k):
            # ajout d'une ligne zoomée de la ligne courante supplémentaire
            grille_zoom.append(liste_zoom)

        # fin BOUCLE 2: la ligne courante est complètement ajoutée à grille_zoom

    # fin BOUCLE 1: toutes les lignes sont parcours donc
    # toutes les lignes sont sont toutes ajoutées grossies dans grille_zoom

    # grille_zoom est correct
    return grille_zoom

affiche(coeur)
affiche(zoomDessin(coeur,3))
```