Concours National Commun (CNC), Maroc

Session 2019, filière MP

Informatique

Ce que traite le sujet

- Exercice : médiane d'une liste de nombres
- Partie I : bases de données et langage SQL
- Partie II : grille binaire, calcul du déterminant
- Partie II : cases, chemins et zones d'une grille binaire

Sujet reproduit sans retouche à partir de la page suivante.

Résumé des parties, chapitres de cours liés et autres années :
prepahelp.com/annales/cnc/mp/2019/info

Prepa Help ne revendique aucun droit sur le sujet, propriété de l'organisateur du concours.

Épreuve d'Informatique – Session 2019 – Filière MP

Les candidats sont informés que la précision des raisonnements algorithmiques ainsi que le soin apporté à la rédaction et à la présentation des copies seront des éléments pris en compte dans la notation. Il convient en particulier de rappeler avec précision les références des questions abordées. Si, au cours de l'épreuve, un candidat repère ce qui peut lui sembler être une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives qu'il est amené à prendre.

Remarques générales :

- Cette épreuve est composée d'un exercice et de deux parties tous indépendants ;
- Toutes les instructions et les fonctions demandées seront écrites en Python ;
- Les questions non traitées peuvent être admises pour aborder les questions ultérieures ;
- Toute fonction peut être décomposée, si nécessaire, en plusieurs fonctions.

≈ ≈

Exercice : (4 points)

Médian d'une liste de nombres

1 pt **Q 1-** Écrire la fonction **`grands(L,x)`** qui reçoit en paramètres une liste de nombres **L**, et un élément x de **L**. La fonction renvoie le nombre d'éléments de **L** qui sont supérieurs strictement à x.

1 pt **Q 2-** Déterminer la complexité de la fonction **grands (L, x)**, et justifier votre réponse.

0.5 pt **Q 3-** Écrire la fonction **`petits(L,x)`** qui reçoit en paramètres une liste de nombres **L**, et un élément x de **L**. La fonction renvoie le nombre d'éléments de **L** qui sont inférieurs strictement à x.

L est une liste de taille **n** qui contient des nombres, et **m** un élément de **L**.

L'élément **m** est un **médian** de **L**, si les deux conditions suivantes sont vérifiées :

- Le nombre d'éléments de **L**, qui sont supérieurs strictement à **m**, est inférieur ou égale à **n/2**
- Le nombre d'éléments de **L**, qui sont inférieurs strictement à **m**, est inférieur ou égale à **n/2**

Exemple : On considère la liste **L = [** 25 , **12** , 6 , 17, 3 , 10 , 20 , **12** , 15 , 38 **]**, de taille **n=10**.

L'élément **12** est un médian de **L**, car :

- **3** éléments de **L** sont supérieurs strictement à **12**, et **3 ≤ n/2** ;
- **5** éléments de **L** sont inférieurs strictement à **12**, et **5 ≤ n/2**.

1 pt **Q 4-** Écrire la fonction **`median(L)`** qui reçoit en paramètre une liste de nombres **L** non vide, et qui renvoie un élément médian de la liste **L**.

0.5 pt **Q 5-** Déterminer la complexité de la fonction **median(L)**, et justifier votre réponse.

Partie I :

Bases de données et langage SQL

Dans les réseaux sociaux, on peut créer des groupes, et d'inviter des personnes à ces groupes. Dans le but de faire des statistiques sur les groupes et leurs membres, nous avons créé une base de données relationnelle composée de trois tables : **Personnes**, **Membres** et **Groupes**.

A- Structure de la table 'Personnes' :

La table 'Personnes' contient les personnes qui ont créé des groupes, et les personnes qui ont accepté l'invitation pour se joindre aux groupes, et devenir membres de ces groupes.

- Le champ **code** de type entier, contient un entier positif unique pour chaque personne ;
- Le champ **nom** de type texte, contient le nom de chaque personne ;
- Le champ **prenom** de type texte, contient le prénom de chaque personne ;
- Le champ **dateNaiss** de type date, contient la date de naissance de chaque personne.

Exemples d'enregistrements dans la table **'Personnes'** :

code	nom	prenom	dateNaiss
1	Boudara	Mohamed	1996-10-15
2	Mernissi	Salima	2000-03-29
3	Sabri	Ahmed	2001-02-20
4	Oubaidi	Mouniya	1990-05-25
5	Jamali	Ahmed	1997-09-08
6	Bekkali	Hamza	1999-11-05
7	Chafi	Asmaa	2002-03-13
8	Chahbouni	Soukaina	2000-01-20
…	…	…	…

B- Structure de la table 'Groupes' :

La table 'Groupes' contient des informations relatives à chaque groupe.

- Le champ **id** de type entier, contient les identifiants des groupes. L'identifiant de chaque groupe est unique ;
- Le champ **nom** de type texte, contient les noms des groupes ;
- Le champ **description** de type texte, contient un texte qui précise les objectifs de chaque groupe ;
- Le champ **codeCr**, de type date, contient les codes des personnes ayant créé les groupes ;
- Le champ **dateCr**, de type date, contient la date de création de chaque groupe.

Exemples d'enregistrements dans la table **'Groupes'** :

id	nom	description	codeCr	dateCr
253	Amis	Les amis les plus proches	2	2017-07-09
471	Fans	Discussion à propos de divers sujets	6	2016-09-14
96	Famille	Les membres de la grande famille	25	2016-10-10
	…	…	…	…

C- Structure de la table 'Membres' :

La table 'Membres' est composée de trois champs :

- Le champ **code** de type entier, contient les codes des personnes qui ont accepté l'invitation pour devenir membres aux groupes ;
- Le champ **id** de type entier, contient les identifiants des groupes ;
- Le champ **dateM** de type date, contient les dates aux quelles chaque personne a rejoint un groupe.

Exemples d'enregistrements dans la table **'Membres'** :

code	id	dateM
1	253	2017-09-18
3	471	2018-01-30
4	253	2016-09-25
5	253	2018-01-22
7	253	2017-07-29
8	471	2016-11-10
...	...	...

I. 1- Déterminer les clés primaires et les clés étrangères dans les tables de cette base de données. Justifier votre réponse.

Écrire, en langage **SQL**, les requêtes qui donnent les résultats suivants :

I. 2- Les personnes qui sont nées le mois **6**, triées dans l'ordre croissant des âges (du moins âgée au plus âgée).

I. 3- Les identifiants des groupes, les dates de création des groupes et les noms et prénoms des personnes ayant crée ces groupes.

I. 4- Écrire la requête **I. 3** en algèbre relationnelle.

I. 5- Les noms et les descriptions des groupes qui contiennent au moins **1000** personnes, triés dans l'ordre décroissant des nombres de personnes.

I. 6- Les codes, les noms et les prénoms de toutes les personnes (créateur et membres) qui appartiennent au groupe ayant le nom : '**Sport**'.

I. 7- Supprimer les personnes qui n'appartiennent à aucun groupe.

I. 8- Modifier dans la table 'Membres', pour que tous les membres du groupe ayant le nom **'Sport'** deviennent membres du groupe ayant le nom **'Voyage'**.

Partie II :

Grille binaire

Une **grille binaire** de **n** lignes et **p** colonnes, est une grille dans laquelle chaque case est de **couleur blanche**, ou bien de **couleur noire**.

Exemple :

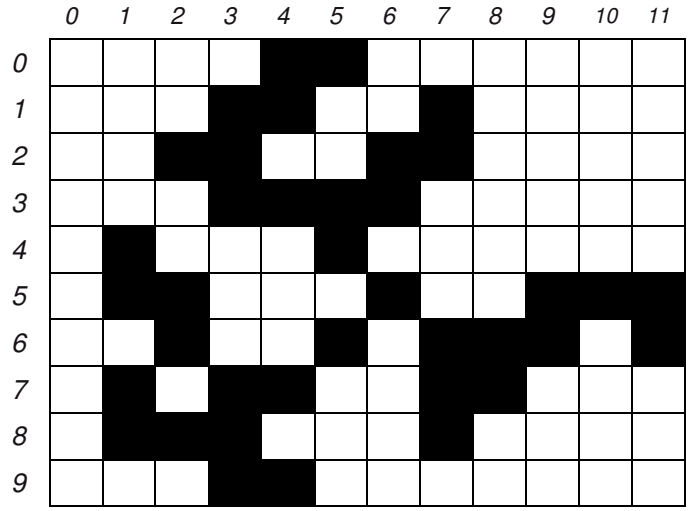


Figure 1 : Exemple de grille binaire de **10** lignes et **12** colonnes

Représentation de la grille binaire :

Pour représenter une grille binaire de **n** lignes et **p** colonnes, on utilise une matrice **G** de **n** lignes et **p** colonnes.

Chaque case de la grille **G** est représentée par un tuple **(i, j)** avec **0 ≤ i < n** et **0 ≤ j < p**, tels que :

- `G[i,j]=1` , si la couleur de la case **(i, j)** est **blanche** ;
- `G[i,j]=0` , si la couleur de la case **(i, j)** est **noire**.

Exemple :

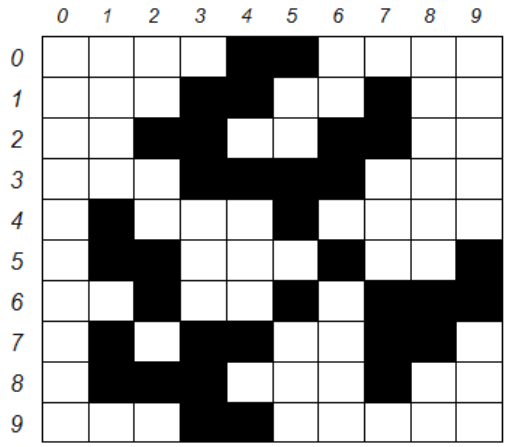
La grille binaire de la **figure 1** est représentée par la matrice **G** de **10** lignes et **12** colonnes, suivante :

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

II. 1- Calcul du déterminant d'une grille binaire carrée

Une grille binaire **carrée** est une grille binaire dans laquelle le nombre de lignes et le nombre de colonnes sont égaux.

Exemple : Grille binaire carrée d'ordre **10** (10 lignes et 10 colonnes).



Dans cette section (**II. 1**), on suppose que le module **numpy** est importé :

from numpy import *

On suppose que la matrice carrée **G**, qui représente la grille binaire carrée, est créée par la méthode **array( )** du module **numpy.**

Exemple :

La grille binaire carrée d'ordre **10** est représentée par la matrice carrée **G** suivante :

```
G = array ([[1,1,1,1,0,0,1,1,1,1] ,
            [1,1,1,0,0,1,1,0,1,1] ,
            [1,1,0,0,1,1,0,0,1,1] ,
            [1,1,1,0,0,0,0,1,1,1] ,
            [1,0,1,1,1,0,1,1,1,1] ,
            [1,0,0,1,1,1,0,1,1,0] ,
            [1,1,0,1,1,0,1,0,0,0] ,
            [1,0,1,0,0,1,1,0,0,1] ,
            [1,0,0,0,1,1,1,0,1,1] ,
            [1,1,1,0,0,1,1,1,1,1]  ] , float)
```

Dans le but de calculer le déterminant d'une matrice carrée **G** qui représente une grille binaire carrée, on propose d'utiliser la méthode du *'pivot de Gauss'*, dont le principe est le suivant :

1. Créer une matrice **C** copie de la matrice **G** ;
2. En utilisant la méthode du *pivot de Gauss*, transformer la matrice **C** en matrice triangulaire inférieure, ou bien en matrice triangulaire supérieure, en comptant le nombre d'échanges de lignes dans la matrice **C**. On pose **k** le nombre d'échanges de lignes dans la matrice **C** ;
3. Calculer le déterminant **D** de la matrice triangulaire **C** ;
4. Le déterminant de la matrice **M** est égal à : $\mathbf{D} * (-\mathbf{1})^{\mathbf{k}}$**.**

Q 1. a- Écrire la fonction `copie_matrice(G)`, qui reçoit en paramètre une matrice carrée **G** qui représente une grille binaire carrée, et qui renvoie une matrice **C** copie de la matrice **G**.

Q 1. b- Écrire la fonction `echange_lignes(C,i,j)`, qui reçoit en paramètres une matrice carrée **C** qui représente une grille binaire carrée. La fonction échange les lignes **i** et **j** dans la matrice **C**.

Q 1. c- Écrire la fonction `triangulaire(C)`, qui reçoit en paramètre une matrice carrée **C** qui représente une grille binaire carrée. En utilisant la méthode du Pivot de Gauss, la fonction transforme la matrice **C** en matrice triangulaire inférieure ou bien triangulaire supérieure, tout en comptant le nombre de fois qu'il y a eu échange de lignes dans la matrice **C**. La fonction doit retourner le nombre d'échanges de lignes.

Exemple :

Après l'appel de la fonction **triangulaire(C)**, on obtient la matrice triangulaire supérieure suivante :

$$\begin{bmatrix} 1. & 1. & 1. & 1. & 0. & 0. & 1. & 1. & 1. & 1. \\ 0. & -1. & 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & -1. & -1. & 1. & 1. & -1. & -1. & 0. & 0. \\ 0. & 0. & 0. & -1. & 0. & 0. & -1. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & -1. & 0. & -1. & 1. & 0. & -1. \\ 0. & 0. & 0. & 0. & 0. & 1. & 1. & -1. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 2. & -1. & 0. & 1. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -1. & -1.5 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 2. \end{bmatrix}$$

La fonction **triangulaire(C)** renvoie le nombre d'échanges de colonnes dans **C** : **4**

Q 1. d- Écrire la fonction `deteminant(G)`, qui reçoit en paramètre une matrice **G** qui représente une grille binaire carrée. En utilisant la méthode du *pivot de Gauss*, la fonction renvoie la valeur du déterminant de **G**,

Exemple :

La fonction **determinant (G)** renvoie : **−4**

II. 2- Représentation de la grille binaire par une liste de listes

Dans la suite de cette partie, pour représenter une grille binaire de **n** lignes et **p** colonnes, on utilise une liste **G** contenant **n** listes toutes de même longueur **p**.

Chaque case de la grille **G** est représentée par un tuple **(i, j)** avec **0 ≤ i < n** et **0 ≤ j < p**, tels que :

- `G[i][j]=1,` si la couleur de la case **(i , j)** est **blanche** ;
- `G[i][j]=0,` si la couleur de la case **(i , j)** est **noire**.

Pour plus de clarté, tous les exemples qui suivront, seront appliqués sur la grille binaire de la **figure 1**.

Exemple :

La grille binaire de la **figure 1** est représentée par la liste **G** suivante, composée de **10** listes, de taille **12** chacune :

```
G =[ [1,1,1,1,0,0,1,1,1,1,1,1] ,[1,1,1,0,0,1,1,0,1,1,1,1] ,
     [1,1,0,0,1,1,0,0,1,1,1,1] ,[1,1,1,0,0,0,0,1,1,1,1,1] ,
     [1,0,1,1,1,0,1,1,1,1,1,1] ,[1,0,0,1,1,1,0,1,1,0,0,0] ,
     [1,1,0,1,1,1,0,0,0,0,1,0] ,[1,0,1,0,0,1,1,0,0,1,1,1] ,
     [1,0,0,0,1,1,1,0,1,1,1,1] ,[1,1,1,0,0,1,1,1,1,1,1,1]
   ]
```

i/j	0	1	2	3	4	5	6	7	8	9	10	11
0	1	1	1	1	0	0	1	1	1	1	1	1
1	1	1	1	0	0	1	1	0	1	1	1	1
2	1	1	0	0	1	1	0	0	1	1	1	1
3	1	1	1	0	0	0	0	1	1	1	1	1
4	1	0	1	1	1	0	1	1	1	1	1	1
5	1	0	0	1	1	1	0	1	1	0	0	0
6	1	1	0	1	1	1	0	0	0	0	1	0
7	1	0	1	0	0	1	1	0	0	1	1	1
8	1	0	0	0	1	1	1	0	1	1	1	1
9	1	1	1	0	0	1	1	1	1	1	1	1

`G[i][j]` est l'élément qui se trouve à la **i**^{ème} ligne et la **j**^{ème} colonne dans **G**.

Exemples :

- **G[0][3]** est l'élément **1**
- **G[1][4]** est l'élément **0**.

`G[i]` est la ligne d'indice **i** dans **G**.

Exemples :

- **G[2]** est la liste **[1 , 1 , 0 , 0 , 1 , 1 , 0 , 0 , 1, 1, 1, 1]**, qui représente la ligne d'indice **2** dans **G**.
- **G[5]** est la liste **[1 , 0 , 0 , 1 , 1 , 1 , 0 , 1 , 1, 0, 0, 0]**, qui représente la ligne d'indice **5** dans **G**.

Q 2- À partir de la liste **G**, de l'exemple ci-dessus, donner le résultat de chacune des expressions suivantes :

G[4][2] , len(G[3]) , G[5] , len(G)

II. 3- Position valide d'une case

Q 3- Écrire la fonction `valide(i,j,G)`, qui reçoit en paramètres deux entiers **i** et **j**. La fonction renvoie **True**, si **i** et **j** sont positifs, et s'ils représentent, respectivement la ligne et la colonne d'une case qui existe dans la grille binaire **G**, sinon**,** la fonction renvoie **False**.

Exemples :

- La fonction **valide (0, 3, G)** renvoie : **True**
- La fonction **valide (7, 15, G)** renvoie : **False**
- La fonction **valide (-2, -8, G)** renvoie : **False**

II. 4- Couleur d'une case

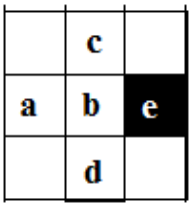
Q 4- Écrire la fonction `couleur(t,G)`, qui reçoit en paramètres un tuple **t** qui représente une case dans la grille binaire **G**. La fonction renvoie **1** si la couleur de la case **t** est blanche, sinon, elle renvoie **0**.

Exemples :

- La fonction **couleur ((0, 3), G)** renvoie le nombre : **1**
- La fonction **couleur ((0, 4), G)** renvoie le nombre : **0**

II. 5- Cases voisines

Dans une grille binaire, deux cases sont voisines, si elles ont la même couleur et si elles ont un seul côté en commun.



Exemples :

- Les cases **a** et **b** sont voisines ;
- Les cases **b** et **c** sont voisines ;
- Les cases **b** et **d** sont voisines ;
- Les cases **b** et **e** ne sont pas voisines, (**b** et **e** n'ont pas la même couleur) ;
- Les cases **a** et **c** ne sont pas voisines, (**a** et **c** n'ont aucun côté en commun) ;
- Les cases **a** et **c** ne sont pas voisines (**a** et **a** n'ont pas un seul côté en commun).

Q 5- Écrire la fonction `list_voisines(t,G)`, qui reçoit en paramètres un tuple **t** représentant une case dans la grille binaire **G**. La fonction renvoie la liste des cases voisines à la case **t**, dans la grille **G**.

Exemples :

- La fonction **list_voisines ((5, 4), G)** renvoie la liste : **[(4, 4), (6, 4), (5, 3), (5, 5)]**
- La fonction **list_voisines ((2, 4), G)** renvoie la liste : **[(2, 5)]**
- La fonction **list_voisines ((5, 1), G)** renvoie la liste : **[(4, 1), (5, 2)]**
- La fonction **list_voisines ((7, 2), G)** renvoie la liste : **[ ]**

II. 6- Chemin dans une grille

*On considère une liste **L** de tuples qui représentent des cases dans une grille binaire **G.***

*La liste **L** est un **chemin** dans la grille **G**, si la liste **L** satisfait les **trois** conditions suivantes :*

c1 - La liste **L** contient au moins deux cases de la grille **G** ;
c2 - Toutes les cases de **L** sont différentes deux à deux, (pas de doublon dans **L**) ;
c3 - Deux cases consécutives dans **L** sont voisines.

Exemples :

- **L = [(0, 3), (0, 2), (0, 1), (0, 0), (1, 0), (2, 0), (3, 0), (3, 1), (3, 2), (4, 2), (4, 3), (4, 4)]**
  La liste **L** est un chemin dans la grille **G**.

- **T = [(0, 0), (1, 1), (2, 1), (2, 2), (1, 2), (1, 1)].**
  La liste **T** n'est pas un chemin dans la grille **G**.
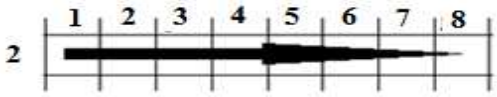
Q 6- Écrire la fonction **`chemin(L,G)`**, qui reçoit en paramètres une liste **L** de tuples représentant des cases dans la grille binaire **G**. La fonction renvoie **True** si la liste **L** est un chemin dans **G**, sinon, la fonction renvoie **False**.

II. 7- Compression d'un chemin dans une grille binaire

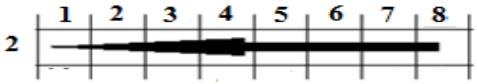
La compression d'un chemin dans une grille binaire, consiste à remplacer, dans ce chemin, chaque suite d'au moins **trois** cases voisines qui se trouvent sur la même ligne ou bien sur la même colonne, par deux cases seulement : la première case et la dernière case.

Exemples :

- Cas d'une suite d'au moins **3** cases voisines, qui se trouvent sur la même ligne :

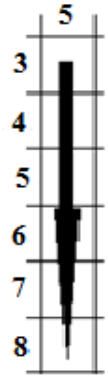


  - La suite des cases voisines : **(2, 1)**, (2, 2), (2, 3), …, **(2, 8),** sera remplacée par : **(2, 1)**, **(2, 8)**

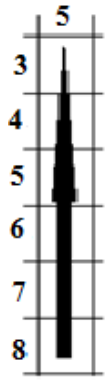


  - La suite des cases voisines : **(2, 8)**, (2, 7), (2, 6), …, **(2, 1),** sera remplacée par : **(2, 8), (2, 1)**.

- Cas d'une suite d'au moins **3** cases voisines, qui se trouvent sur la même colonne :



  - La suite des cases : **(3, 5)**, **(**4, 5), (5, 5), …, **(8, 5),** sera remplacée par : **(3, 5), (8, 5)** ;



  - La suite des cases : **(8, 5)**, (7, 5), (6, 5), …, **(3, 5),** sera remplacée par : **(8, 5), (3, 5)**.

Q 7- Écrire la fonction **`compresse_chemin(L)`**, qui reçoit en paramètre une liste **L** qui contient un chemin dans une grille binaire. La fonction renvoie la liste qui contient le chemin compressé de **L**.

Exemples :

- **L = [(0, 0),** (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0), (7, 0), (8, 0), **(9, 0),** (9, 1), **(9, 2)]**
  La fonction **compresse_chemin(L)** renvoie la liste : **[(0, 0), (9, 0), (9, 2)]**

- **L = [(0, 0), (1, 0), (1, 1), (2, 1), (2, 0), (3, 0)]**
  La fonction **compresse_chemin(L)** renvoie la liste : **[(0, 0), (1, 0), (1, 1), (2, 1), (2, 0), (3, 0)]**

II. 8- Appartenance d'une case à un chemin compressé

Q 8- Écrire la fonction **`appartient(t,L)`**, qui reçoit en paramètres une liste **L** contenant un chemin compressé dans une grille binaire, et un tuple **t** représentant une case. La fonction renvoie **True** si le chemin compressé **L** passe par la case **t**, sinon, la fonction renvoie **False**.

Exemples :

On considère le chemin compressé, de l'exemple précédent : **L = [(0, 0) , (9, 0) , (9, 2)]**

- Les cases **(0, 0)**, **(1, 0)**, **(2, 0), (8, 0), (9, 1)** appartiennent au chemin **L** ;
- La case **(0, 2)** n'appartient pas au chemin **L**.

II. 9- Longueur d'un chemin compressé

Dans une grille binaire, la longueur d'un chemin compressé, est égale au nombre total des cases par les quelles passe ce chemin.

Exemples :

- La longueur du chemin compressé **[(0, 0), (9, 0), (9, 2)]** est : **12** ;
- La longueur du chemin compressé **[(0, 0), (1, 0), (1, 1), (2, 1), (2, 0), (3, 0), (3, 1)]** est : **7**.

Q 9- Écrire la fonction, de complexité linéaire, **`longueur_chemin(L)`**, qui reçoit en paramètre une liste **L** qui contient un chemin compressé. La fonction renvoie la longueur du chemin **L**.

II. 10- Chemin entre deux cases

*On suppose que **a** et **b** sont deux tuples qui représentent deux cases distinctes, de même couleur, dans une grille binaire **G**.*

*Un chemin entre les cases **a** et **b**, s'il existe, est un chemin compressé, tel que le tuple **a** est son premier élément, et le tuple **b** est son dernier élément.*

***NB** : On peut trouver plusieurs chemins entre les cases **a** et **b**.*

Exemple :

Les chemins suivants sont des chemins compressés entre les cases **a = (0, 3)** et **b = (4, 4)** :

- [**(0, 3)**, (0, 2), (1, 2), (1, 1), (0, 1), (0, 0), (3, 0), (3, 2), (4, 2), (4, 3), (6, 3), (6, 4), **(4, 4)**]
- [**(0, 3)**, (0, 2), (1, 2), (1, 1), (3, 1), (3, 2), (4, 2), **(4, 4)**]
- [**(0, 3)**, (0, 2), (1, 2), (1, 1), (2, 1), (2, 0), (3, 0), (3, 2), (4, 2), **(4, 4)**]
- [**(0, 3)**, (0, 0), (1, 0), (1, 1), (3, 1), (3, 2), (4, 2), (4, 3), (6, 3), (6, 4), **(4, 4)**]
- **…**

Q 10. a- Écrire la fonction `chemins(G,a,b,chemin)`, reçoit en paramètres deux tuples **a** et **b** distincts qui représentent deux cases de même couleur, dans la grille binaire **G**. Le paramètre **chemin** est une liste initialisée par la liste vide. Cette fonction renvoie une nouvelle liste contenant tous les chemins entre la case **a** et la case **b** dans la grille **G**.

Exemples :

- La fonction **chemins (G, (0, 3), (4, 4), [ ])** renvoie la liste des chemins suivants :
  [[(0, 3), (0, 2), (1, 2), (1, 1), (2, 1), (3, 1), (3, 2), (4, 2), (4, 3), **(4, 4)] , [(0, 3)**, (0, 2), (1, 2), (1, 1), (0, 1), (0, 0), (1, 0), (2, 0), (2, 1), (3, 1), (3, 2), (4, 2), (4, 3), (5, 3), (5, 4), **(4, 4)] , [** (**0, 3)**, (0, 2), (1, 2), (1, 1), (2, 1), (3, 1), (3, 2), (4, 2), (4, 3), (5, 3), (5, 4), **(4, 4)] , [(0, 3)**, (0, 2), (1, 2), (1, 1), (0, 1), (0, 0), (1, 0), (2, 0), (3, 0), (3, 1), (3, 2), (4, 2), (4, 3), **(4, 4)] , …]**

- La fonction **chemins (G, (6, 0), (7, 2), [ ])** renvoie la liste : **[ ]**

Q 10. b- Écrire la fonction `list_chemins(G,a,b)`, reçoit en paramètres deux tuples **a** et **b** qui représentent deux cases quelconques dans la grille binaire **G**. Cette fonction renvoie la liste de tous les chemins compressés entre la case **a** et la case **b**, dans la grille **G**.

Exemples :

- La fonction **list_chemins (G, (0, 3), (4, 4), [ ])** renvoie la liste des chemins compressés suivants :
  [[(0, 3), (0, 2), (1, 2), (1, 1), (3, 1), (3, 2), (4, 2), **(4, 4)] , [(0, 3)**, (0, 2), (1, 2), (1, 1), (0, 1), (0, 0), (2, 0), (2, 1), (3, 1), (3, 2), (4, 2), (4, 3), (5, 3), (5, 4), **(4, 4)] , [** (**0, 3)**, (0, 2), (1, 2), (1, 1), (2, 1), (3, 1), (3, 2), (4, 2), (4, 3), (5, 3), (5, 4), **(4, 4)] , [(0, 3)**, (0, 2), (1, 2), (1, 1), (0, 1), (0, 0), (3, 0), (3, 2), (4, 2), **(4, 4)] , …]**

- La fonction **list_chemins (G, (6, 0), (8, 1), [ ])** renvoie la liste : **[ ]**

II. 11- Tri d'une liste de chemins

Q 11- Écrire la fonction `tri_chemins(R)`, qui reçoit en paramètre une liste **R** contenant tous les chemins compressés entre deux cases dans une grille binaire. La fonction trie les chemins compressés de **R** dans l'ordre croissant des longueurs des chemins. La longueur d'un chemin telle qu'elle est définie dans la question **II. 9**

***NB** : Ne pas utiliser la méthode sort( ), ni la fonction sorted( ).*

Exemple :

On considère la liste **R** suivante, qui contient **3** chemins compressés dans la grille binaire **G**.

R = [[(0, 3), (0, 0), (3, 0), (3, 2), (4, 2), (4, 4) **] , [** (0, 3), (0, 2), (1, 2), (1, 1), (3, 1), (3, 2), (4, 2), (4, 4) **] , [** (0, 3), (0, 1), (3, 1), (3, 2), (4, 2), (4, 4) **]]**

Après l'appel de la fonction **tri_chemins (R)**, on obtient la liste triée suivante :

[[(0, 3), (0, 2), (1, 2), (1, 1), (3, 1), (3, 2), (4, 2), (4, 4) **] , [** (0, 3), (0, 1), (3, 1), (3, 2), (4, 2), (4, 4) **] , [** (0, 3), (0, 0), (3, 0), (3, 2), (4, 2), (4, 4) **]]**

II. 12- Plus courts chemins entre deux cases

a *et* ***b*** *sont deux tuples qui représentent deux cases dans une grille binaire* ***G****. Le* ***plus court chemin*** *entre* ***a*** *et* ***b*** *est le chemin compressé entre* ***a*** *et* ***b*** *ayant la plus petite longueur.*

NB *: On peut trouver plusieurs plus courts chemins entre les cases* ***a*** *et* ***b****.*

Q 12- Écrire la fonction **`plus_court_chemins(G,a,b)`** , qui renvoie la liste de tous les plus courts chemins compressés entre deux cases **a** et **b**.

Exemples :

- La fonction **plus_court_chemins ((0, 3), (4, 4), G)** renvoie la liste des deux chemins suivants :
  [[(0, 3), (0, 2), (1, 2), (1, 1), (3, 1), (3, 2), (4, 2), **(4, 4)]** , **[(0, 3)**, (0, 1), (3, 1), (3, 2), (4, 2), **(4, 4)]]**

- La fonction **plus_court _chemins ((6, 0), (7, 2), G)** renvoie la liste : **[ ]**

II. 13- Zone d'une case dans une grille binaire

On considère un tuple ***t*** *qui représente une case dans une grille binaire* ***G****.*

La zone de la case ***t*** *est l'ensemble qui contient la case* ***t****, et toutes les cases de la grille binaire* ***G****, qu'on peut joindre par un chemin à partir de la case* ***t****.*

Exemples :

- La zone de la case **(8, 2)** est l'ensemble composé des cases suivantes :
  (8, 2) , (7, 3) , (8, 3) , (7, 1) , (8, 1) , (9, 3) , (7, 4) , (9, 4)

- La zone de la case **(9, 6)** est l'ensemble composé des cases suivantes :
  (9, 6) , (7, 9) , (8, 4) , (6, 6) , (7, 6) , (9, 8) , (9, 9) , (8, 9) , (7, 5) , (8, 8) , (8, 6) , (9, 5), (8, 5), (9, 7)

- La zone de la case **(7, 2)** est l'ensemble composé d'une seule cases : **(7, 2)**

Q 13-Écrire la fonction **`compte_zones(G)`** , qui reçoit en paramètre une grille binaire **G**, et qui renvoie le nombre de zones dans la grille binaire **G**.

Exemple :

La fonction **compte_zone (G)** renvoie le nombre : **10**

≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈ FIN DE l'ÉPREUVE ≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈

Travailler les chapitres de ce sujet

Prepa Help suit le programme de MP chapitre par chapitre : cours détaillé, méthodes, pièges, exercices et problèmes corrigés pas à pas, tirés des concours. L'essai gratuit donne accès aux chapitres offerts de ta filière, sans carte bancaire.

Chapitres liés à ce sujet

- Bases de Python : expressions, types, instructions, fonctions
- Complexité des algorithmes itératifs et récursifs
- Calcul algébrique : sommes, produits, binôme et systèmes linéaires
- Récursivité
- Tri par fusion et tri rapide

Ouvrir l'essai gratuit

Scanne le code ou clique sur le bouton.

prepahelp.com/#essai

Une question sur le concours ou sur ta filière : **écrire sur WhatsApp**

Les autres années et épreuves : prepahelp.com/annales/cnc/mp