

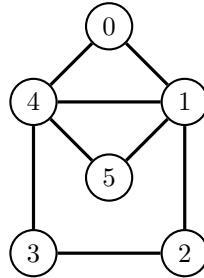
I Graphes eulériens

Dans cet exercice, on considère un graphe non orienté G non vide et dont on suppose que les sommets sont numérotés à partir de 0. Le graphe G contiendra donc au moins un sommet numéroté 0.

Le graphe est représenté par son dictionnaire d'adjacence : un dictionnaire dont les clefs sont tous les sommets du graphe et, pour chaque sommet s , la valeur associée est la liste des sommets reliés à s par une arête.

Un cycle eulérien est un cycle qui passe une seule fois par chacune des arêtes du graphe, ce qui correspond à un moyen de dessiner le graphe en une seule fois, sans lever le stylo et sans repasser deux fois sur la même arête. Un graphe est dit eulérien s'il possède un cycle eulérien.

Pour illustrer les différents points abordés, on utilisera le graphe suivant :



Le dictionnaire associé est $G = \{0:[1,4] \text{ , } 1:[0,2,4,5] \text{ , } 2:[1,3] \text{ , } 3:[2,4] \text{ , } 4:[0,1,3,5] \text{ , } 5:[1,4]\}$

La complexité, ou le temps d'exécution, d'une fonction P est le nombre d'opérations élémentaires (addition, soustraction, multiplication, division, affectation, etc...) nécessaires à l'exécution de P . Lorsque cette complexité dépend d'un paramètre n , on dira que P a une complexité en $O(f(n))$, s'il existe $K > 0$ tel que la complexité de P est au plus $K f(n)$, pour tout n .

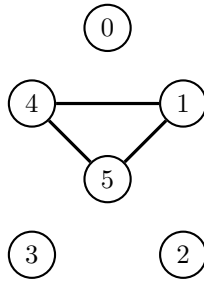
1. Le degré d'un sommet s de G est le nombre d'arêtes qui partent de ce sommet.
 - a) Écrire une fonction `degre(G:dict,s:int)->int` qui prend en argument le dictionnaire du graphe et le numéro d'un sommet et qui renvoie le degré du sommet s .
 - b) Quelle est la complexité de votre fonction ; expliquez très sommairement votre résultat.
2. Un graphe est dit « connexe » s'il est possible, depuis le sommet 0 de rejoindre tous les autres sommets du graphe. En utilisant un parcours en profondeur du graphe, écrire une fonction `connexe(G:dict)->bool` qui renvoie `True` si le graphe est connexe et `False` sinon.
3. On admet qu'un graphe est eulérien si et seulement s'il est connexe et tous ses sommets sont de degré pair. Écrire une fonction `eulerien(G:dict)->bool` qui renvoie `True` si G est eulérien et `False` sinon.
4. Quelques fonction annexes :
 - a) Un sommet est dit isolé s'il n'est relié à aucun autre sommet du graphe. Écrire une fonction `isole(G:dict,s:int)->bool` qui renvoie `True` si le sommet s de G est isolé et `False` sinon.
 - b) Écrire une fonction `aretes(G:dict)->int` qui renvoie le nombre d'arêtes du graphe G .
 - c) Écrire une fonction `supprimer(G:dict,s:int,t:int)->None` qui prend en argument un graphe G et deux sommets distincts s et t , supposés reliés par une arête, et qui supprime cette arête dans le graphe G (c'est le graphe G lui même qui doit être modifié).
Si $L = [1,2,3]$ est une liste, l'utilisation de `L.remove(2)` modifie la liste L en $L = [1,3]$. La complexité de cette méthode est linéaire par rapport à la longueur de la liste sur laquelle on l'applique.

5. Construction d'un cycle eulérien

Dans la suite, on supposera que le graphe est eulérien et on décrit un algorithme permettant d'en trouver un cycle eulérien :

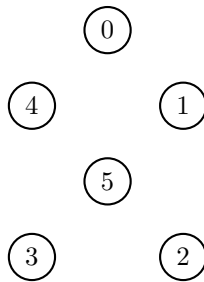
On appelle boucle depuis le sommet s , un cycle depuis s et qui revient à s sans passer deux fois par la même arête, mais qui n'emprunte pas forcément toutes les arêtes du graphe.

- On commence par déterminer une boucle depuis le sommet 0 et on supprime du graphe les arêtes empruntées. Une telle boucle sera représentée par la liste des sommets rencontrés, dans l'ordre dans lequel ils sont traversés. On choisit sur l'exemple traité, lorsque plusieurs arêtes partent d'un sommet, de commencer par celle qui joint le sommet dont le numéro est le plus petit.
Une boucle depuis 0 est alors $L = [0,1,2,3,4,0]$ et le graphe, après suppression des arêtes devient



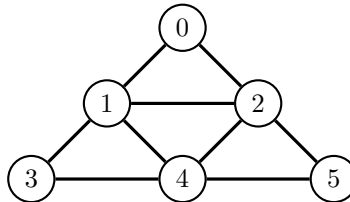
- Une fois cette première boucle déterminée, on parcourt les sommets s de L :
 - Si s est isolé, on ne fait rien et on passe au sommet suivant dans L .
 - Sinon, on détermine une boucle depuis s dans le graphe restant, toujours en supprimant les arêtes empruntées au fur et à mesure.
 - On insère alors la nouvelle boucle obtenue dans la liste L , à la place du sommet s
 - On réitère le processus sur cette nouvelle liste L jusqu'à avoir parcouru toutes les arêtes du graphe.

Sur notre exemple, le premier élément de L est 0, qui est isolé. Le deuxième est 1 qui n'est pas isolé ; une boucle depuis 1 dans le nouveau graphe est $M = [1, 4, 5, 1]$ et le graphe devient



On insère alors la liste M dans L à l'emplacement initial du sommet 1 : on obtient $L = [0, 1, 4, 5, 1, 2, 3, 4, 0]$. On a déjà parcouru toutes les arêtes du graphe donc inutile de poursuivre. Un cycle eulérien est donc représenté par la liste $[0, 1, 4, 5, 1, 2, 3, 4, 0]$.

- a) Un exemple : on considère le graphe suivant



Déterminer un cycle eulérien de ce graphe en appliquant l'algorithme décrit précédemment : pour chaque boucle, comme dans l'exemple expliqué, lorsque plusieurs sommets sont possibles, vous choisirez systématiquement celui qui porte le plus petit numéro ; vous représenterez l'état du graphe après chaque itération et la valeur de L correspondante.

- b) Une boucle depuis s peut-elle passer plusieurs fois par s ?
 Pourquoi existe-t-il, lorsque s n'est pas isolé, une boucle depuis s pour laquelle s sera isolé après suppression des arêtes empruntées au cours de cette boucle ?
 Justifier vos affirmations.
- c) Écrire une fonction `boucle(G:dict,s:int)->list` qui renvoie la liste associée à une boucle depuis s dans G . Cette fonction doit supprimer les arêtes empruntées dans le graphe G . Après utilisation de cette fonction depuis un sommet s , le sommet s doit être isolé dans le graphe restant.
- d) En déduire une fonction `cycle(G:dict)->list` qui renvoie la liste représentant un cycle eulérien du graphe G .
- e) Quelle est la complexité de `cycle` en fonction du nombre n de sommets et du nombre p d'arêtes du graphe G ; expliquez votre raisonnement.

————— Fin de l'exercice I —————

II Algorithme de Kaprekar

1. On considère la fonction $f(n,p)$ suivante; n et p sont deux entiers naturels tels que $n < 10^p$.

```
def f(n,p) :  
    L = []  
    for k in range(p) :  
        n,r = n//10,n%10  
        L.append(r)  
    return L
```

- a) Exécuter $f(123,3)$ et $f(22,3)$; remplissez un tableau faisant apparaître les valeurs de k,n,r,L au cours de l'exécution et la valeur renvoyée.
- b) Quel est le rôle de cette fonction?
2. Écrire une fonction $occ(L:list) \rightarrow list$ qui prend en argument une liste L d'entiers de $\llbracket 0,9 \rrbracket$ (donc des chiffres), supposée non vide, et qui renvoie une liste F , de longueur 10, telle que $F[k]$ est le nombre de fois où le chiffre k apparaît dans la liste L .
Par exemple $occ([0,1,1,2,5,1,7,3,0,4,5,0])$ devra renvoyer $[3,3,1,1,1,2,0,1,0,0]$.
3. On se donne deux entiers n et p tels que $n < 10^p$. On suppose posséder une fonction $chiffres(n,p)$ qui renvoie la liste des p chiffres de l'entier n (en rajoutant des zéros si besoin); $chiffres(121,4)$ renvoie $[1,2,1,0]$ (l'ordre des chiffres n'a pas d'importance). À partir des chiffres de l'entier n , on écrit les deux nombres M et m tels que M (*resp.* m) soit le plus grand (*resp.* le plus petit) des nombres que l'on peut écrire avec les p chiffres de n .
Par exemple si $n=6106$ et $p=5$, la liste des p chiffres de n est $[6,1,0,6,0]$; on a alors $M=66100$ et $m=166$ (ou $m=00166$). On peut remarquer que pour écrire les nombres M et m , il suffit de ranger les p chiffres de n dans l'ordre croissant ou décroissant.
- a) Si on suppose que $occ(chiffres(n,p))$ renvoie la liste $[2,0,1,0,2,1,0,3,1,0]$, quelles sont les valeurs des entiers M et m associés à n ?
- b) Écrire une fonction $K(n:int,p:int) \rightarrow tuple$ qui renvoie le couple (M,m) associé aux p chiffres de n .
4. L'algorithme de Kaprekar consiste, à partir d'un entier n , écrit sur p chiffres, à calculer les entiers M et m précédents, puis à recommencer à partir de l'entier $M-m$ jusqu'à retrouver une valeur déjà rencontrée (une fois cette répétition atteinte, les valeurs suivantes décrivent un cycle). À partir de $n=42$, avec $p=2$, les valeurs obtenues sont les suivantes

n	M	m	$M-m$
42	42	24	18
18	81	18	63
63	63	36	27
27	72	27	45
45	54	45	9
9	90	9	81
81	81	18	63

Écrire une fonction $Kaprekar(n:int,p:int) \rightarrow list$ qui prend en argument deux entiers n et p (tels que $n < 10^p$) et qui renvoie la liste des différentes valeurs de $M-m$ définies précédemment, jusqu'à l'apparition d'une répétition. On rappelle que *elt in L* renvoie *True* ou *False* selon que *elt* appartient ou non à la liste L .

$Kaprekar(42,2)$ devra renvoyer $[42,18,63,27,45,9,81,63]$

_____ Fin de l'exercice II _____