Université Bordeaux 1 — Devoir surveillé du 16 janvier 2012
Département Licence, **J1MI1003** — 11h-12h30

Numéro d'anonymat : Groupe :

Le sujet comporte 7 exercices et 11 pages, dont une page d'annexe.
Aucun document n'est autorisé : toutes les fonctions de manipulation des graphes disponibles sont rappelées en annexe, page 11, que vous pouvez détacher pour plus de facilités.

Exercice 1 Soit la fonction `Python` suivante :

```
def mystere(x,y,z):
    p = (y-x)/z
    s = x
    r = [x]
    for i in range(z):
        s = s+p
        r = r+[s]
    return r
```

1. Donnez les valeurs successives prises par les variables `i`, `s` et `r` lors de l'appel de `mystere(0,20,5)` ainsi que la valeur retournée.

i	
s	
r	

2. Même question pour `mystere(6,2,4)`.

i	
s	
r	

3. Que calcule cette fonction de manière générale ?

4. Quel est l'ordre de grandeur (en fonction de `x`, `y` et `z`) du nombre d'opérations effectué par `mystere(x,y,z)` ?

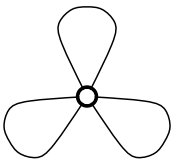
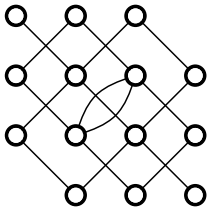
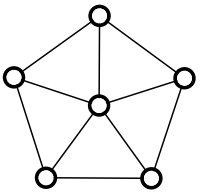
Exercice 2 (Sommets et degrés)

1. Écrire une fonction `nombreBoucles(s)` qui renvoie le nombre de boucles incidentes au sommet `s`.

2. Écrire une fonction `sommetDegreMax(G)` qui renvoie un sommet de degré maximum du graphe `G`.

Numéro d'anonymat : Groupe :

Exercice 3 On rappelle que K_5 désigne le graphe complet à 5 sommets. On appelle F, G et H les graphes suivants :

F =  G =  H = 

1. Remplir le tableau suivant (en répondant par OUI ou par NON lorsque cela est demandé). **Attention**, les réponses fausses enlèvent des points :

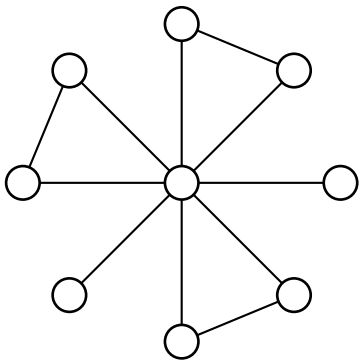
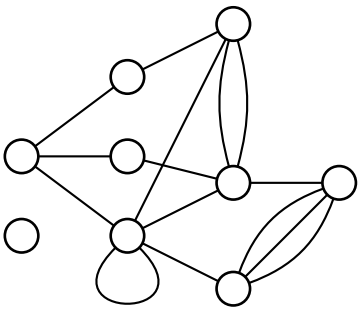
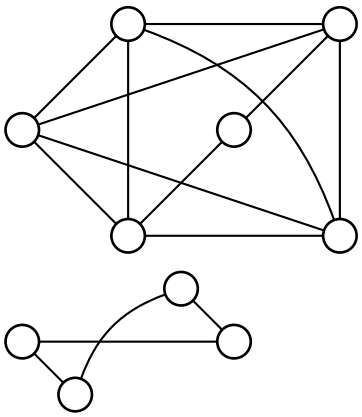
Graphe	F	G	H	K_5
Connexe ?				
Simple ?				
Eulérien ? (courte justification souhaitée)				
Coloriable ?				
Si oui, nombre chromatique ?				

2. Un arbre est-il toujours 2-coloriable ? Justifier.

Exercice 4 Une **clique** dans un graphe G est un sous-ensemble de sommets $C \subseteq S(G)$, tel que pour toute paire de sommets distincts de C, il existe au moins une arête dans G reliant les deux sommets.

Si l'ensemble C contient k sommets et forme une clique, on dira que C est une **k-clique** ou que C forme une **clique de taille k**.

1. Sur chacun des trois graphes ci-dessous, colorier un et un seul ensemble de sommets et d'arêtes formant une :

2−clique	3−clique	4−clique
		

2. Écrire une fonction `estUneClique(L)` qui teste si la liste de sommets `L` est une *clique*. Cette fonction doit donc renvoyer `True` si pour toute paire (x, y) de sommets de la liste `L` (avec $x \neq y$), x et y sont voisins, et `False` sinon.

3. Quel est le nombre minimum d'arêtes contenues dans une k-clique pour $k \geq 1$?

Numéro d'anonymat : Groupe :

4. Comment appelle-t-on un graphe simple ayant n sommets et qui possède une n-clique ?

Exercice 5 Un traitement médical lourd doit être prescrit par un médecin à son patient. Il doit être attentif à l'incompatibilité de certains médicaments entre eux. Le tableau ci-dessous indique quels sont les médicaments qui ne doivent pas être pris en même temps (par exemple, A est donc incompatible avec B, C et D) :

Médicament	A	B	C	D	E
Incompatibilité	B,C,D	A	A,D	A,C	

1. Modéliser les incompatibilités médicamenteuses entre ces cinq médicaments par un graphe.

2. Quel est le nombre chromatique de ce graphe ? (justifier)

3. Sachant que chaque médicament doit être pris durant toute une semaine, quel sera la durée minimale du traitement ?

Exercice 6 (Chaînes dans les graphes) En Python, on représente une chaîne dans un graphe simple par une liste de sommets.

1. Écrire une fonction `passeParTousLesSommets(L, G)` qui prend en argument une chaîne représentée par une liste de sommets `L` et qui vérifie si cette chaîne passe par tous les sommets de `G`.

2. Soit G un graphe simple possédant une chaîne qui passe par tous les sommets. Le graphe est-il forcément eulérien (justifier ou donner un contre-exemple) ?

Numéro d'anonymat : Groupe :

Exercice 7 (Coloration à distance 2) Dans cet exercice, on considère que vous disposez de la fonction `sontVoisins(s1,s2)` qui renvoie `True` si deux sommets `s1` et `s2` sont voisins dans un graphe et `False` sinon.

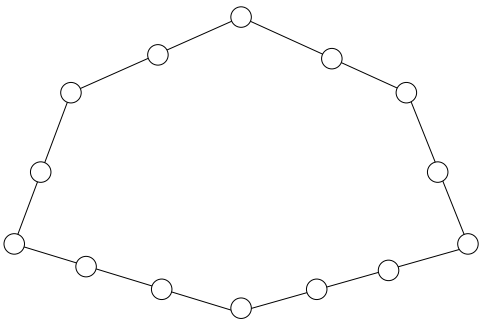
On dit que deux sommets `s1` et `s2` d'un graphe sont à *distance* 2 s'ils ont un voisin en commun, c'est-à-dire s'il existe une chaîne de longueur 2 entre `s1` et `s2` et que `s1` et `s2` ne sont pas voisins entre eux.

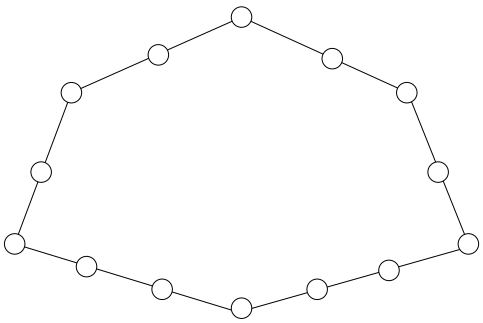
1. Écrire une fonction `sontADistance2(s1,s2)` qui renvoie `True` si les sommets `s1` et `s2` sont à distance 2 dans un graphe et `False` sinon.

On a vu en cours qu'un graphe est dit « *bien colorié* » (on dit aussi qu'un graphe a une « *coloration propre* ») si deux sommets voisins ont toujours des couleurs différentes.

Un graphe est dit « *bien colorié à distance* 2 » (on dit aussi qu'un graphe a une « *coloration propre à distance 2* ») si le graphe est « *bien colorié* » et si deux sommets à distance au plus 2 ont toujours des couleurs différentes.

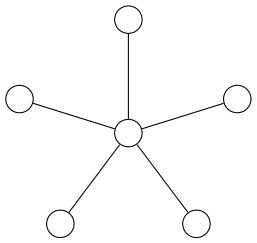
2. Ci-dessous sont représentés deux cycles de longueur 14. En utilisant un minimum de couleur, donnez une « *coloration propre* » du premier (figure de gauche) et une « *coloration propre à distance* 2 » du second (figure de droite).





3. Écrivez une fonction `estBienColorieADistance2(G)` qui renvoie `True` si un graphe `G` est « *bien colorié à distance* 2 » et `False` sinon.

4. Quel est le nombre de couleurs nécessaires pour bien colorier à distance 2 le graphe en étoile ci-dessous ?



Soit un graphe G avec n sommets et de degré maximum Δ (c'est-à-dire qu'il y a au moins un sommet de degré $d = \Delta$, mais aucun de degré $d > \Delta$).

5. Montrer que pour « *bien colorier un graphe à distance 2* » il est nécessaire d'utiliser au moins Δ couleurs.

Numéro d'anonymat : Groupe :

6. **Bien plus difficile :** Contruire un graphe qui ne peut pas être « *bien colorié à distance 2* » avec $\Delta + 5$ couleurs.

FIN.

Annexe : voici un rappel des principales fonctions disponibles pour manipuler les graphes (à vous de voir celles qui sont utiles pour ce devoir).

L'argument G est un graphe	
`listeSommets(G)`	retourne la *liste* des *sommets* de G.
`nbSommets(G)`	retourne le *nombre* de *sommets* de G.
`sommetNom(G,etiquette)`	retourne le *sommet* de G désigné par son *nom* (étiquette). Exemple : `sommetNom (Europe, 'Italie').`
`sommetNumero(G,i)`	retourne le *sommet* numéro i dans G ; la numérotation commence à 0.

L'argument s est un sommet	
`listeVoisins(s)`	retourne la *liste* des *voisins* de s.
`degre(s)`	retourne le *degré* de s.
`nomSommet(s)`	retourne le *nom* (étiquette) de s.
`colorierSommet(s,c)`	colorie s avec la couleur c. Exemples de couleurs : `'red'`, `'green'`, `'blue'`, `None`.
`couleurSommet(s)`	retourne la *couleur* de s.
`marquerSommet(s)` `demarquerSommet(s)`	marque ou démarque s.
`estMarqueSommet(s)`	retourne `True` si s est marqué, `False` sinon.
`listeAretesIncidentes(s)`	retourne la *liste* des arêtes *incidentes* à s.

L'argument a est une arête	
`nomArete(a)`	retourne le *nom* (étiquette) de a.
`marquerArete(a)` `demarquerArete(a)`	marque ou démarque a.
`estMarqueeArete(a)`	retourne `True` si a est marquée, `False` sinon.

Arguments : un sommet s et une arête a	
`sommetVoisin(s,a)`	retourne le voisin de s en suivant l'arête a.