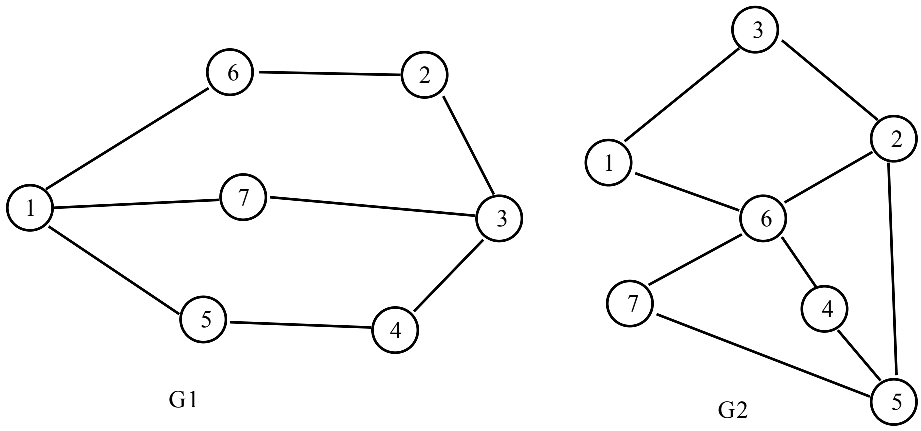
Problème Graphe Biparti et bicoloration:

On rappelle qu'un graphe G=(X,U) est biparti si toutes ses arêtes ont une extrémité dans un sous-ensemble X_1 et l'autre extrémité dans $X-X_1$. L'algorithme BIPAR(G) a pour objet de tester si un graphe G connexe est biparti.

Algorithme BIPAR(G) :

```
{-1- Initialisations}
Lire un graphe connexe G=(X,U); {initialement aucun sommet n'est coloré}
colorer le sommet 1 en rouge;
initialiser le booléen BIPARTI à VRAI.

{-2- Examen des sommets colorés}
Tant qu'il existe un sommet i coloré et non examiné faire
Début
    Si i est rouge alors
        pour chaque voisin j de i faire
            Début
                si j est rouge alors BIPARTI=FAUX
                si j est non coloré alors colorer j en vert
            Fin
    sinon { i est vert alors }
        pour chaque voisin j de i faire
            Début
                si j est  vert alors BIPARTI=FAUX
                si j est non coloré alors colorer j en rouge
            Fin
Fin
{-3- Sortie de l'algorithme}
Si BIPARTI = VRAI alors écrire "G est biparti", sinon écrire "G n'est pas biparti"..
```



Question 1: Faire tourner l'algorithme sur les graphes G_1 et G_2. On rapportera les couleurs pour chaque itération de la boucle.

Question 2: On code le graphe G=(X,U) par la matrice A associée, c'est-à-dire : si (i,j) est une arête du graphe alors $a_{ij} = a_{ji} = 1$, sinon $a_{ij} = 0$.

(a) Quels tableaux supplémentaires sont-ils nécessaires pour BIPAR(G) ?

(b) Quelle est la complexité résultante de l'algorithme ? Justifier.

Question 3: Proposer de façon explicite une meilleure structure de données. Que devient la complexité ? Justifier.

Question 4: (Cette question est indépendante de l'algorithme)

Montrer qu'un graphe est biparti si et seulement s'il est bicolorable.

Question 5: (Cette question est indépendante de l'algorithme)

Montrer qu'un graphe biparti ne contient pas de cycle de longueur impaire.

Question 6: Quelles sont les propriétés des sommets colorés en rouge et en vert au cours de l'algorithme? Justifier. Où intervient la connexité ?

Question 7: Montrer la validité de l'algorithme. En déduire la réciproque de la question 5 dans le cas connexe.

Question 8: Comment proposez-vous de modifier l'algorithme pour tester si un graphe quelconque, c'est-à-dire non nécessairement connexe est biparti?

Question 9: Le problème de coloration minimale d'un graphe quelconque est NP-difficile. Existe-il un algorithme de complexité polynomiale permettant de le résoudre ?

Exercice 1 :

Montrez qu'un graphe est un arbre si et seulement si il existe une chaîne élémentaire et une seule entre chaque paire de sommets.

Exercice 2 :

Soit G un graphe non orienté à n sommets et m arêtes. Les arêtes de G sont notées u_1, u_2 .. u_m. G_i= (V, E_i), est le graphe formé des arêtes u_1, u_2 .. u_i. On pose G_0= (V, Φ).

Question 1. Quel est le nombre de composantes connexes de G_0.

Question 2. On note $C(G_i)$ le nombre de composantes connexes de G_i. Montrer que

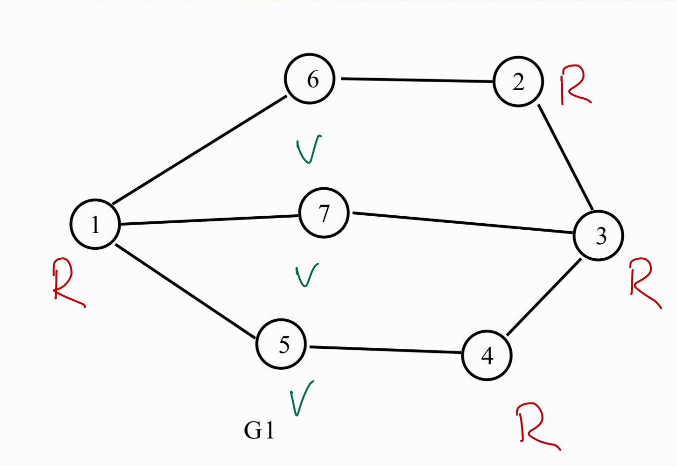
$C(G_i) = C(G_{i-1})$ ou $C(G_i) = C(G_{i-1}) - 1$.

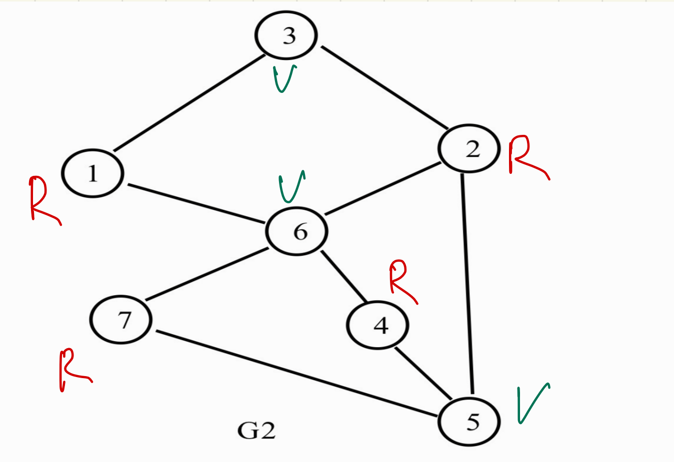
Question 3. Interpréter le cas $C(G_i) = C(G_{i-1})$, puis le cas $C(G_i) = C(G_{i-1}) - 1$.

Question 4. Montrer qu'un graphe connexe a au moins n-1 arêtes.

Question 5. Montrer qu'un graphe sans cycle a au plus n-1 arêtes.

Question 1: Faire tourner l'algorithme sur les graphes G_1 et G_2. On rapportera les couleurs pour chaque itération de la boucle.





BIPARTI = VRAI
1) Sommet 1 Rouge
2) examen du sommet 1
6, 7 et 5 = Vert
3) examen du sommet 6
2 devient Rouge
4) examen du sommet 7
3 devient rouge
5) examen du sommet 5
4 devient Rouge
6) examen du sommet 2
BIPARTI = FAUX
7) examen du sommet 3
BIPARTI = FAUX
8) examen du sommet 4
BIPARTI = FAUX
Conclusion G_1 n'est pas biparti

En appliquant l'algorithme sur G2 on obtiendra la coloration donnée ci-dessus. Conclusion G_2 est biparti

Question 2: On code le graphe G=(X,U) par la matrice A associée, c'est-à-dire : si (i,j) est une arête du graphe alors $a_{ij} = a_{ji} = 1$, sinon $a_{ij} = 0$.

(a) Quels tableaux supplémentaires sont-ils nécessaires pour BIPAR(G) ?

(b) Quelle est la complexité résultante de l'algorithme ? Justifier.

a) On aura besoin d'un tableau "COULEUR" pour stocker les couleurs données aux sommets et une pile ou file pour stocker les sommets colorés mais non examinés.

b) Complexité : 1) Initialisation → lecture de A = $O(n^2)$
2) Examen des sommets : Examen d'un sommet correspond au parcour de la ligne ⇒ $O(n)$.
Puisqu'il y a n sommets ⇒ $O(n^2)$
3) Sortie de l'algorithme $O(1)$

Complexité globale $O(n^2)$

Question 3: Proposer de façon explicite une meilleure structure de données. Que devient la complexité ? Justifier.

Le graphe initial est un graphe non orienté $G=(X,U)$ on considère le graphe orienté $H=(X, U+\bar{U})$ où pour toute arête (i,j) on associe l'arc (i,j) et (j,i). Ensuite on code ce nouveau graphe avec les files α, β, et on obtient une complexité globale de $O(m)$.

La phase d'initialisation coûterait $O(n+m) \sim O(m)$

L'examen d'un sommet i coûterait $O(d^+(i))$, donc la complexité de la phase 2 coûterait

$$O\left(\sum_{i=1}^{n} d^+(i)\right) = O(m) \Rightarrow \text{Complexité globale } O(m).$$

Question 4: (Cette question est indépendante de l'algorithme)

Montrer qu'un graphe est biparti si et seulement s'il est bicolorable.

On démontrera que :

G est biparti $\Leftrightarrow$ G est bicolorable

G est biparti $\Rightarrow$ $\exists$ une partition des sommets en V_1 et V_2 tel que toute arête a un sommet dans V_1 et l'autre dans V_2 $\Rightarrow$ cette propriété implique qu'on peut colorer les sommets dans V_1 avec Rouge et ceux dans V_2 en vert et cette coloration est valide.

Vice-versa, si le graphe est bicolorable, supposons en rouge et en vert, alors notons les sommets rouge V_1 et les verts V_2, ce qui donne une partition valide puisque aucune arête a les deux extrémités rouge (dans V_1) ou vert (dans V_2).

Question 5: (Cette question est indépendante de l'algorithme)

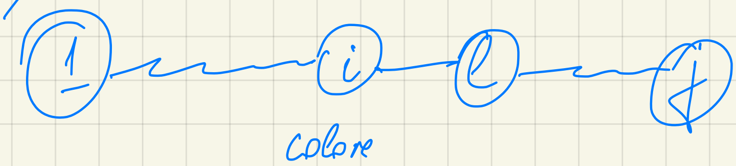
Montrer qu'un graphe biparti ne contient pas de cycle de longueur impaire.

Dans un graphe biparti toute arête a une extrémité dans une partie et l'autre dans la 2ème partie, donc toute chaîne a ses sommets successives alternativement dans une et l'autre partie. Par conséquent toute chaîne ayant son origine et sa destination dans la même partie est forcement de longueur paire. Or, tout cycle a forcement son origine = sa destination, donc dans la même partie => tout cycle est donc de longueur paire => pas de cycle de longueur impaire.

Question 6: Quelles sont les propriétés des sommets colorés en rouge et en vert au cours de l'algorithme? Justifier. Où intervient la connexité ?

La propriété est la suivante : Pour tout sommet rouge (respectivement vert) il existe une chaîne de longueur paire (respectivement impaire) depuis le sommet 1.
Cette propriété peut être démontrée par récurrence sur le nombre k d'itérations de la boucle principale de l'algorithme.
Soit k=1. => seulement le sommet 1 est coloré en rouge, donc la propriété est vraie.
Supposons que la propriété est vraie après les k premières itérations. Supposons qu'lors de la k+1-ème itération on examine le sommet i coloré en rouge. Tous les nouveaux sommets colorés seraient verts. Pour ces nouveaux sommets colorés, notons un tel sommet j, il existe une chaîne de longueur impaire composé d'une chaîne de 1 à i de longueur paire (hypothèse de récurrence) + arête (i,j).
Le même raisonnement peut être fait si i est un sommet vert.

La connexité garantie qu'à la fin de l'algorithme tous les sommets sont colorés. En effet, raisonnons par absurde et supposons qu'il existe un sommet j non coloré à la fin de l'algorithme. Puisque le graphe est connexe il existe alors une chaine de 1 vers j. Parcourons cette chaine en commençant par le sommet 1 et notons par i le dernier sommet coloré et l (le sommet suivant dans la chaine) devait le premier sommet non coloré de la chaine. Or, le sommet l aurait du être coloré lors de l'examen du sommet i (On rappele que tout sommet coloré est examiné à la fin de l'algo).



Le fait que le sommet l devrait être coloré à la fin de l'algorithme conduit à une contradiction puisque l est un sommet non coloré. Fin de la preuve.

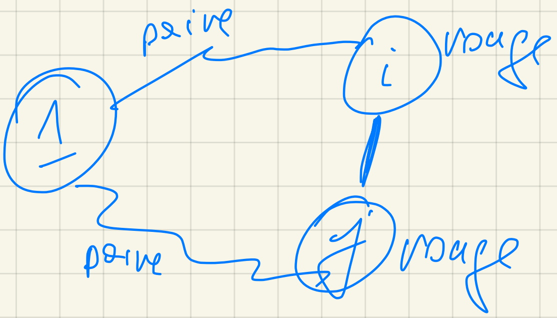
Question 7: Montrer la validité de l'algorithme. En déduire la réciproque de la question 5 dans le cas connexe.

Pour montrer la validité de l'algorithme il faudra qu'on montre que si la sortie de l'algorithme est : "G est biparti" alors le graphe G est vraiment biparti et quand la sortie est "G n'est pas biparti" alors le graphe G ne peut pas être biparti.

Clairement, On affiche "G est biparti" si à la fin de l'algorithme la variable "BIPARTI" n'est jamais mise à FAUX, donc quand on arrive à colorer le graphe avec 2 couleurs. Selon la Q4 ce graphe est alors biparti.

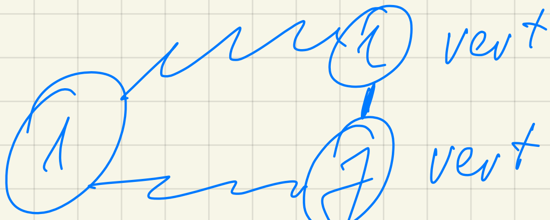
Ensuite, si à la fin de l'algorithme "BIPARTI" = FAUX alors cela est arrivé seulement dans deux cas:

Cas 1). Quand on examine un sommet i rouge et on découvre qu'il a un voisin j qui est déjà coloré en rouge. On en déduit qu'il existe un cycle de longueur impaire. (voir la figure).



Selon la Q5 on déduit que G n'est pas biparti.

Le même raisonnement peut être suivi pour le cas 2).



On en déduit que l'algorithme est valide.

Pour la réciproque de la Q5, on raisonne comme suit : Soit G un graphe sans cycles de longueur impaire. Alors si on applique l'algorithme BIPAR(G) alors les cas 1 et 2 ci-dessus ne peuvent jamais se produire, donc BIPARTI = VRAI $\Rightarrow$ G est biparti.

Question 8: Comment proposez-vous de modifier l'algorithme pour tester si un graphe quelconque, c'est-à-dire non nécessairement connexe est biparti?

Il suffit de tester l'algorithme BIPAR sur chaque composante connexe du graphe

Question 9: Le problème de coloration minimale d'un graphe quelconque est NP-difficile. Existe-il un algorithme de complexité polynomiale permettant de le résoudre ?

Selon la conjecture de la théorie de la Complexité $P \neq NP$ on peut alors déduire que le problème de coloration minimale d'un graphe quelconque n'admette pas un algorithme de complexité polynomiale.

Exercice 1 :

Montrez qu'un graphe est un arbre si et seulement si il existe une chaîne élémentaire et une seule entre chaque paire de sommets.

La définition donnée dans le pdf copié : un arbre est un graphe Connexe et Sans Cycles

le fait que le graphe est connexe implique qu'il existe une chaîne entre chaque paire de sommets

Le fait que le graphe est sans cycle implique qu'il ne peut pas y avoir deux chaînes distinctes reliant une paire de sommets car cela donnerait un cycle

Il existe une et une seule chaîne élémentaire entre chaque paire de sommets.

Exercice 2 :

Soit G un graphe non orienté à n sommets et m arêtes. Les arêtes de G sont notées u_1, u_2 .. u_m. G_i= (V, E_i), est le graphe formé des arêtes u_1, u_2 .. u_i. On pose G_0= (V, Φ).

Question 1. Quel est le nombre de composantes connexes de G_0.

Question 2. On note $C(G_i)$ le nombre de composantes connexes de G_i. Montrer que

$C(G_i) = C(G_{i-1})$ ou $C(G_i) = C(G_{i-1}) - 1$.

Question 3. Interpréter le cas $C(G_i) = C(G_{i-1})$, puis le cas $C(G_i) = C(G_{i-1}) - 1$.

Question 4. Montrer qu'un graphe connexe a au moins n-1 arêtes.

Question 5. Montrer qu'un graphe sans cycle a au plus n-1 arêtes.

Q1) il y a n composantes connexes dans G_0
chaque sommet donne une composante connexe.

Q2) Quand on rajoute l'arc u_i dans G_{i-1} il y a deux cas de figure : 1) les extrémités de u_i se trouvent dans la même composante connexe dans G_{i-1}, ce qui donne $C(G_i) = C(G_{i-1})$;
2) les extrémités de u_i se trouvent dans des composantes connexes différentes de G_{i-1} $\Rightarrow C(G_i) = C(G_{i-1}) - 1$.

Q3) $C(G_i) = C(G_{i-1})$ correspond au cas où rajout de u_i creera un cycle dans G_{i-1}, tandis que dans le cas $C(G_i) = C(G_{i-1}) - 1$ correspond au fait que l'ajout de u_i dans G_{i-1} ne crée pas de cycle.

Q4) Pour obtenir un graph connexe à partir de G_0 il faudra au moins n-1 ajouts d'arcs qui ne créent pas de cycle car à chaque ajout on diminue par 1 donc on passera de $C(G_0) = n$ à 1 composante $\Rightarrow$ au moins n-1 arcs.

Q5) Pour ne pas créer de cycle il faudra qu'on diminue le nombre de composantes connexes et puisqu'on ne peut pas descendre au dessous de 1, il faudra au plus n-1 arcs.