Chaînes : (<=> chemin)

k : est la longueur de la chaine (<=> n-1 sommet)
↑ nbr de sommet.

Une chaine est élémentaire si les sommet sont 2 à 2 distinct

chaîne simple < chaîne élémentaire.
↑ sommet 2 à 2 disjoint.

Cycles :

k : le cycle est pair ou impair

Graph sans cycle : graphe acyclique (sans boucle)
L'arbre : graph acyclique et connexe. (sans racine != informatique)

Connexe : chaque pair de sommet est relier

connexe non connexe

Composant connexe <= sous graphe connexe
↑ def avec des relations d'équivalence

Déterminer si c'est un arbre (théorème : caractérisation des arbres

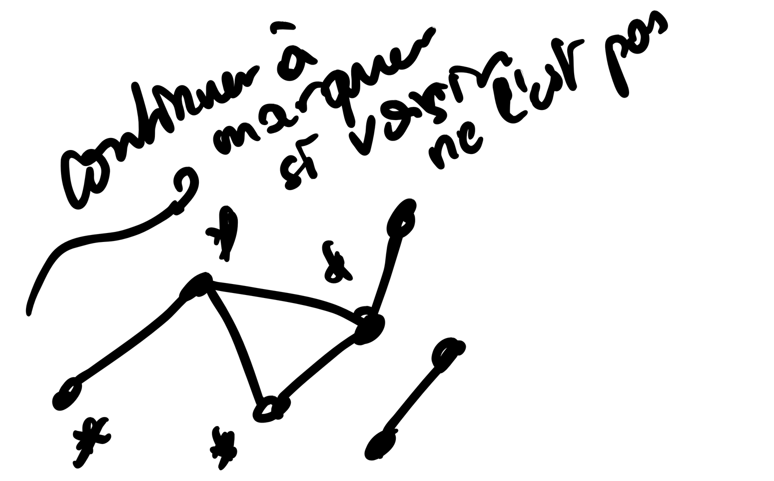
1. G est un arbre
2. pour ∀ paire uv ∃ une unique chaine entre u et v.
3. G est connexe et si on supp n'importe quelle chaine alors le graph est !connexe
4. Le graphe est acyclique : l'ajout d'une arête → graphe cyclique
5. G est connexe } nbr sommet = nbr arête +1 <=> $|V(G)| = |E(G)| + 1$

G un graphe, sous graph couvrant (contient H les sommets + connexe + acyclique)
par
arbre couvrant de G

Un graph connexe <=> Il contient un arbre couvrant

Pour savoir si un graph est connexe ~> Algorithme de marquage
//regarder au final s'il existe des arêtes non marquées
Entrées: graphe $G = (V, E)$ et sommet $s \in V$
début

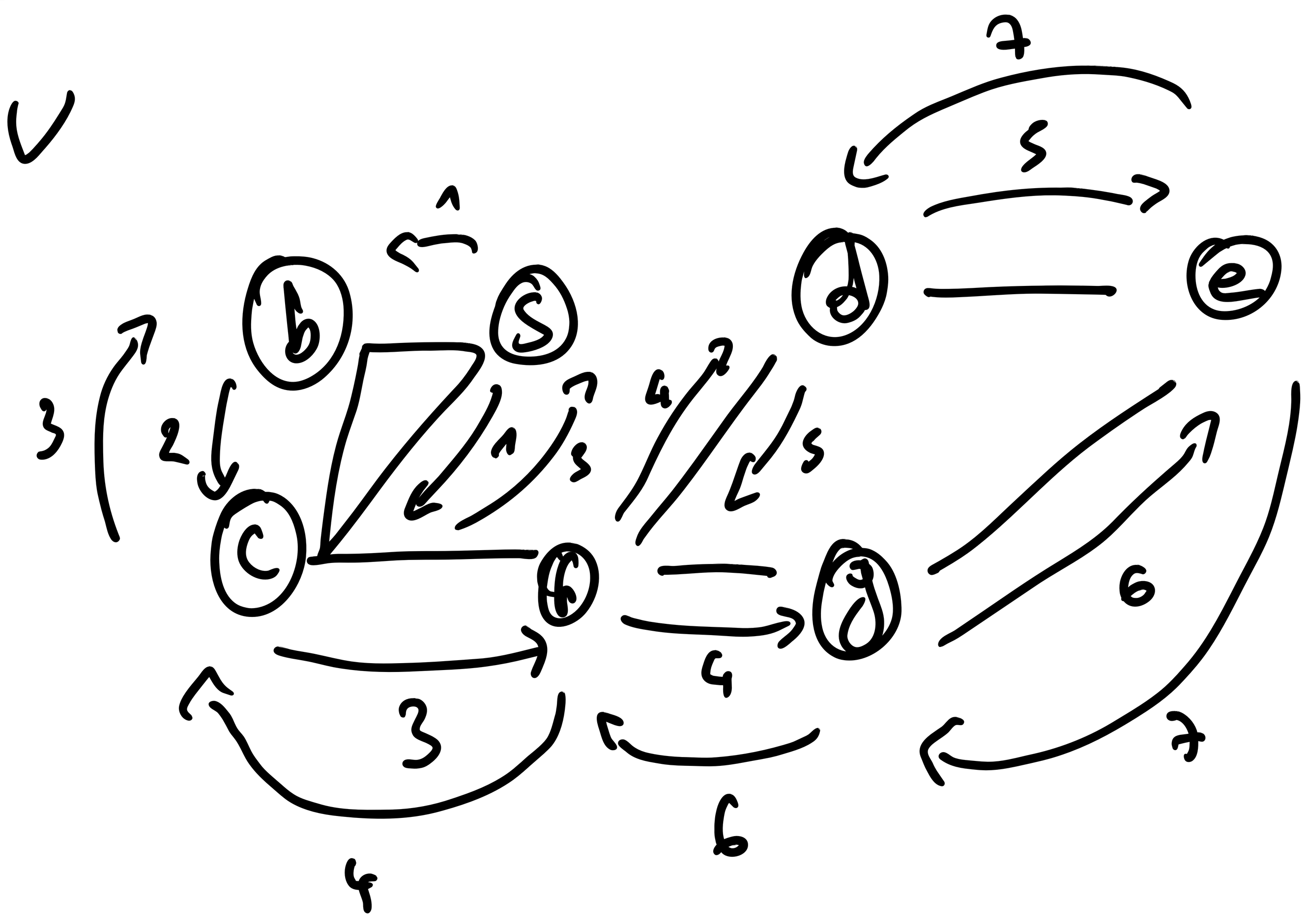
| marquer(s);
| tant que $\exists$ uv $\in$ E où u est marqué et v non marqué faire
| | marquer(v)



S'arrêter puis comparer si Il sommet est marqué
↳ si non connexe alors graph pas entièrement marqué.

Parcours en largeur (BFS) Breadth-first search
Entrées : graphe $G = (V,E)$ et sommet $s \in V$
début

créer file(Q)
marquer(s);
enfiler(Q, s)
tant que $Q \neq \emptyset$ faire
| u ← défiler(Q)
| pour tous les uv $\in$ E faire
| si v non marqué alors
| | marquer(v)
| | enfiler(Q, v)



BFS (version avec distance)

Entrées : graphe $G=(V,E)$ et sommet $s \in V$

début

pour tous les $u \in V \setminus \{s\}$ faire

$\quad d(u) \leftarrow \infty$

$d(s) \leftarrow 0$

$Q \leftarrow [s]$

tant que $Q \neq \emptyset$ faire

$\quad u \leftarrow$ défiler(Q)

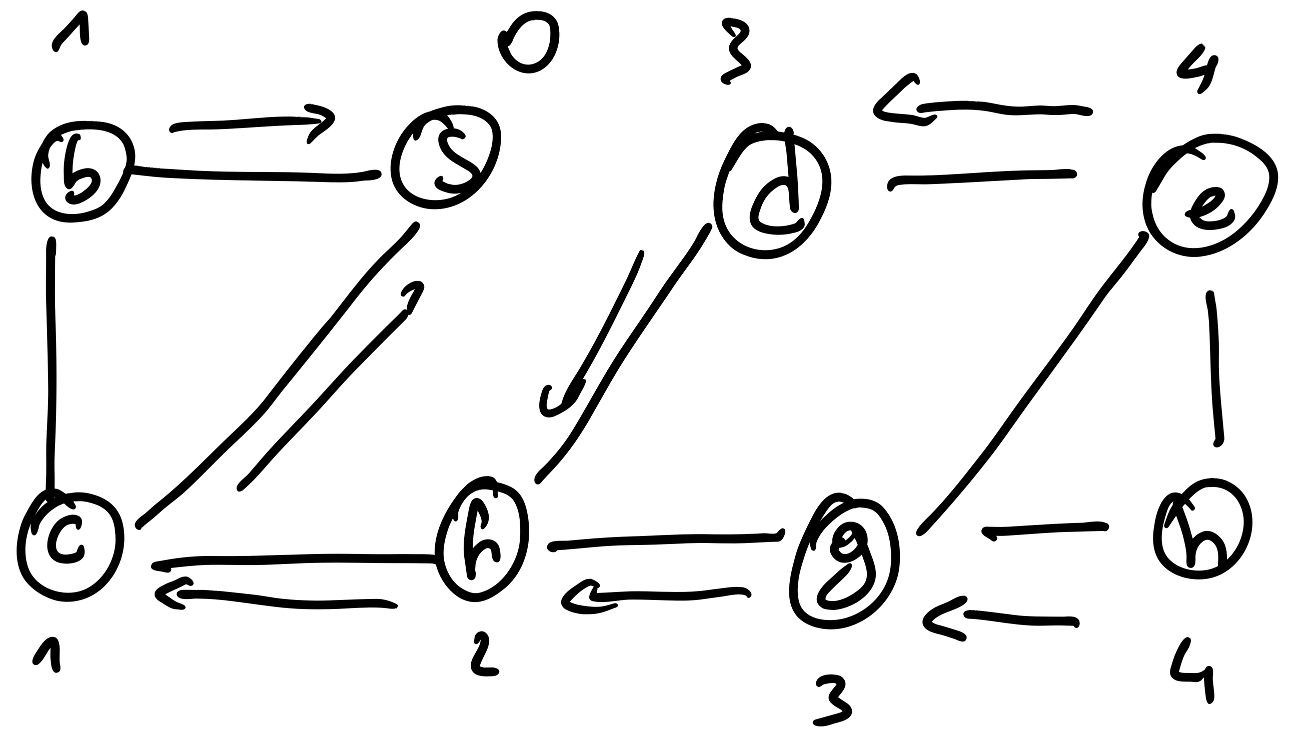
$\quad$ pour tous les $uv \in E$ faire

$\quad\quad$ si $d(v)=\infty$ alors

$\quad\quad\quad d(v) = d(u)+1$

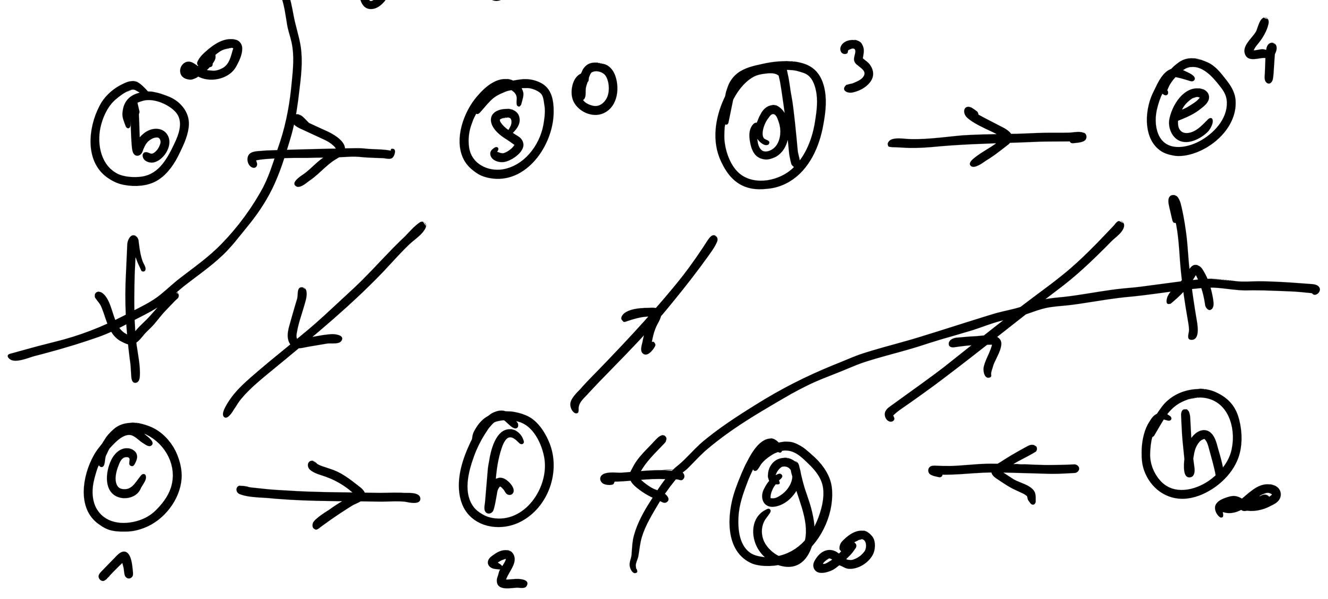
$\quad\quad\quad$ enfiler (Q, v);

$\quad\quad\quad$ parent$(v) \leftarrow u$



non orienté

orienté



// si on lève orientation graph connexe (

Complexité $O(n+m)$ où $n=|V|$ et $m=|E|$ // liste d'adjacence

\\ car // arrêtes est considérés 2 fois

) chaque sommet est enfilé 1 fois

Correction BFS :

Lemme 1 : Après terminaison du BFS, $d(v) \geq dist(s,v)$ pour tout sommet $v \in V$

Lemme 2 : si au cours BFS $Q=[v_1, v_2, \dots, v_r]$ alors $d(v_r) \leq d(v_1)+1$ et $d(v_i)$ pour $i=1,2,\dots,r-1$

Corollaire : Si le sommet v_i est enfilé dans Q avant v_j, alors $d(v_i) \leq d(v_j)$

// code Python + application ex.