

Question 1

(3+4 points)

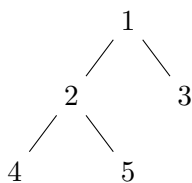
La recherche en profondeur (DFS) partage les arcs d'un graphe en 4 catégories : arc-arbre (arc dans l'arbre DFS), arc-avant (arc vers un descendant dans l'arbre DFS), arc-retour (arc vers un ancêtre dans l'arbre DFS) et arc-transverse (sinon).

On utilise la même classification pour la recherche en largeur (BFS) :

```
BFS (G = (V,E), s in V) // G graphe orienté
// F file de sommets de G
// marked(v) = 1 si v déjà rencontré, et 0 sinon
// D(v) calcule la distance d(s,v)
// Pred(v) calcule le predecesseur de v dans l'arbre BFS

F := (s)
marked(s) := 1
D(s) := 0
Tant que (F non-vide) faire
  v = defiler(F)
  Pour tout w tel que (v,w) in E faire
    Si marked(w) = 0 alors
      marked(w) := 1
      D(w) := D(v) + 1
      Pred(w) := v
      enfiler(F,w)
  FinSi
FinPour
FinTantque
```

On suppose que G est un graphe orienté avec ensemble de sommets $V = \{1, 2, 3, 4, 5\}$ et que l'arbre BFS obtenu à partir du sommet $s = 1$ est le suivant :



Autrement dit, on suppose que $\text{Pred}(2) = \text{Pred}(3) = 1$, et $\text{Pred}(4) = \text{Pred}(5) = 2$. On suppose également que les listes d'adjacence de G sont triées en ordre croissant.

1. Expliquez pourquoi E ne peut pas contenir ni l'arc $(1, 4)$, ni $(1, 5)$.
2. Soit $E = \{(i, j) \mid 1 \leq i, j \leq 5, i \neq j, (i, j) \neq (1, 4), \text{ et } (i, j) \neq (1, 5)\}$.
Montrez l'exécution de l'algorithme BFS avec $G = (V, E)$ et $s = 1$, en calculant le contenu de F , le sommet actuel v , les fonctions D et Pred , à chaque itération de la boucle "Tantque". Précisez pour chaque arc son type.

Question 2

(3+3 points)

L'entrée pour ce problème sont 1) la matrice d'adjacence $A \in \{\text{true}, \text{false}\}^{n \times n}$ d'un graphe orienté G , dont l'ensemble de sommets est $\{1, \dots, n\}$, et 2) un sous-ensemble de sommets $U \subseteq \{1, 2, \dots, n\}$. L'algorithme Floyd-Warshall ci-dessous devra être complété afin de calculer pour toute paire de sommets $(i, j) \in V \times V$ s'il existe un chemin de i à j qui **ne passe pas par l'ensemble** U (sauf éventuellement pour i ou j). Autrement dit, on demande qu'à la fin de Floyd-Warshall on ait $A[i, j] = \text{true}$ ssi i et j sont connectés par un chemin $i = u_0, u_1, \dots, u_{k-1}, u_k = j$ tel que $\{u_1, \dots, u_{k-1}\} \cap U = \emptyset$.

Floyd-Warshall ($A[1..n, 1..n]$ matrice d'adjacence,

U sous-ensemble de $\{1, 2, \dots, n\}$)

Initialisation : ...

Pour $k = 1$ à n faire

 Pour $i = 1$ à n faire

 Pour $j = 1$ à n faire

 Si (...) alors

$A[i, j] = \dots$ ou (... et ...)

 FinSi

 FinPour

 FinPour

FinPour

1. Complétez la fonction ci-dessus (Initialisation, instruction “Si alors”, affectation $A[i, j] := \dots$) de telle façon que l’invariant suivant est respecté :

Après la ℓ -ième exécution de la première boucle “Pour” on a $A[i, j] = \text{vrai}$ ssi il existe un chemin de i à j qui passe seulement par des sommets intermédiaires de l’ensemble $\{1, \dots, \ell\} \setminus U$.

2. Exécutez votre algorithme sur l’entrée suivante : $V = \{1, 2, 3, 4, 5\}$, $E = \{(1, 2), (1, 3), (2, 4), (2, 5), (3, 1), (3, 4), (3, 5), (4, 1), (5, 1), (5, 3)\}$, et $U = \{2, 4\}$. Indiquez pour chaque itération de la première boucle “Pour” le contenu de la matrice booléenne A .

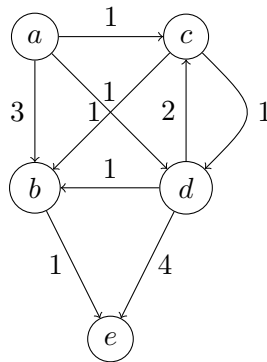
Question 3

(5+2 points)

Dijkstra ($G = (V, E)$ graphe orienté, $P : E \rightarrow \mathbb{R}$ fonction de poids, s in V sommet source)

```
// U, R sous-ensembles de V (U : terminé, R : candidats)
// D(v) poids (minimal) des chemins de s à v
// Pred(v) prédecesseur de v sur un chemin de poids D(v)
U := ∅
R := {s}
Pour tout v in V faire
    D(v) = infinity
FinPour
D(s) := 0
Tant que (R non-vide) faire
    v := delete_min(R)
    U := U union {v}
    Pour tout w tel que (v,w) in E faire
        Si (D(w) > D(v) + P(v,w)) alors
            Si (w in U) alors retourner "erreur"
            Sinon // meilleur chemin de s à w via v
                Pred(w) := v
                D(w) := D(v) + P(v,w)
            Si (w notin R) alors R := R union {w} FinSi
    FinSi
FinSi
FinPour
FinTantque
```

- Montrez l'exécution de l'algorithme de Dijkstra sur le graphe pondéré suivant, en supposant que les listes d'adjacence de chaque sommet sont en ordre croissant (exemple : $\text{Adj}(a) = (b, c, d)$). Indiquez pour chaque itération de l'algorithme : 1) les ensembles de nœuds $R, U \subseteq V$, 2) le prédécesseur $\text{Pred}(x)$ et la distance $D(x)$ pour chaque nœud $x \in U \cup R$, 3) le nœud v de la boucle.



- Donnez un exemple de graphe pondéré où l'algorithme retourne le message d'erreur.