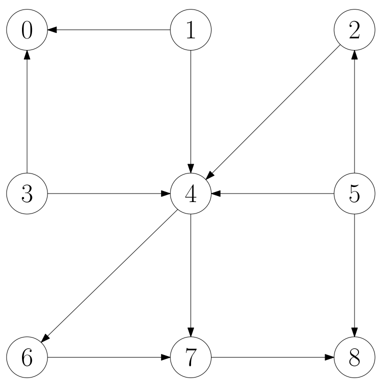
I Sujet d'oral CCINP

Dans cet exercice, tous les graphes sont orientés. On représente un graphe orienté $G = (S, A)$, avec $S = \{0, \ldots, n-1\}$, en C par la structure suivante :

```c
typedef struct {
    int n; // nombre de sommets
    int degre[100]; // degre[s] est le degré sortant de s
    int voisins[100][10]; // voisins[s] contient les successeurs de s
} graphe;
```




```c
graphe g_exemple = {
      .n = 9,
      .degre = {0,2,1,2,2,3,1,1,0},
      .voisins = {
      /* 0 */ {-1}, // Degré 0 : ignoré
      /* 1 */ {0, 4},
      /* 2 */ {4},
      /* 3 */ {0, 4},
      /* 4 */ {6, 7},
      /* 5 */ {2, 4, 8},
      /* 6 */ {7},
      /* 7 */ {8},
      /* 8 */ {-1} // Degré 0 : ignoré
      }
};
```

Pour $s \in S$ on note $A(s)$ l'ensemble des sommets accessibles à partir de s. Pour $s \in S$, le maximum des degrés des sommets accessibles depuis s est noté $d^*(s)$. Par exemple, pour le graphe ci-dessus, $A(2) = \{2, 4, 6, 7, 8\}$ et $d^*(2) = 2$ car le degré sortant de 4 est $d^+(4) = 2$. Dans cet exercice, on cherche à calculer $d^*(s)$ pour chaque sommet $s \in S$.

On représente un sous-ensemble de sommets $S' \subseteq S$ par un tableau de booléens de taille n, contenant `true` à la case d'indice s' si $s' \in S'$ et `false` sinon.

1. Écrire une fonction `int degre_max(graphe* g, bool* partie)` qui calcule le degré maximum d'un sommet $s' \in S'$ dans un graphe $G = (S, A)$ pour une partie $S' \subseteq S$ représentée par `partie`, c'est-à-dire qui calcule $\max\{d^+(s') \mid s' \in S'\}$.

2. Écrire une fonction `bool* accessibles(graphe* g, int s)` qui prend en paramètre un graphe et un sommet s et qui renvoie un tableau de booléens de taille n représentant $A(s)$.

3. Écrire une fonction `int degre_etoile(graphe* g, int s)` qui calcule $d^*(s)$ pour un graphe et un sommet passés en paramètre. Quelle est la complexité de cette fonction ?

4. Donner un tri topologique du graphe ci-dessus.

5. Dans cette question, on suppose que le graphe $G = (S, A)$ est acyclique. Décrire un algorithme permettant de calculer tous les $d^*(s)$ pour $s \in S$ en $\mathrm{O}(|S| + |A|)$.

6. On ne suppose plus le graphe acyclique. Décrire un algorithme permettant de calculer tous les $d^*(s)$ pour $s \in S$ en temps $\mathrm{O}(|S| + |A|)$.

II Graphes semi-connexes

Soit $G = (S, A)$ un graphe orienté. On dit que G est semi-connexe si pour tous $(s, t) \in S^2$, il existe un chemin de s à t ou un chemin de t à s.

1. Donner un algorithme très simple de complexité $\mathrm{O}(n(n+p))$ (avec $n = |S|$ et $p = |A|$) permettant de déterminer si un graphe orienté acyclique est semi-connexe.

2. Soit $G = (S, A)$ un graphe acyclique et $(s_0, \ldots, s_{n-1})$ un ordre topologique de G. Montrer que G est semi-connexe si et seulement si pour tout $i \in [\![0, n-2]\!]$, $(s_i, s_{i+1}) \in A$.

3. En déduire un algorithme plus efficace qui décide si un graphe orienté quelconque est semi-connexe et préciser sa complexité.

III Hypercube

Un hypercube Q_n a pour sommets les mots binaires de taille n, 2 sommets étant reliés s'ils différent d'un bit.

1. Dessiner Q_3.
2. Quel est le nombre de sommets et d'arêtes de Q_n?
3. Montrer que Q_n est biparti.
4. Montrer que Q_n possède un couplage parfait.
5. Soit $n \geq 2$. Montrer que Q_n est hamiltonien : il existe un cycle qui visite tous les sommets exactement une fois. Dessiner un tel cycle de Q_3.

IV Questions sur les couplages

1. Soit G un graphe. Montrer que si G a un couplage parfait alors G possède un nombre pair de sommets. La réciproque est-elle vraie ?
2. Soit M_1 et M_2 deux couplages d'un graphe G, avec M_2 maximal (c'est-à-dire qu'on ne peut pas ajouter d'arête à M_2 en conservant un couplage). Montrer que $|M_1| \leq 2|M_2|$, puis donner un cas d'égalité.
3. Soit $G = (V, E)$ un graphe. Une couverture par sommets (*vertex cover*) de G est un ensemble C de sommets tels que chaque arête de G est adjacente à au moins un sommet de C. L'objectif est de trouver une couverture par sommets C^* de cardinal minimum.
   On propose l'algorithme suivant :

   2-approximation de vertex cover

   $M \longleftarrow$ couplage maximal de G
   $C \longleftarrow$ ensemble des sommets couverts par M

   Montrer que C est bien une couverture par sommet et que $|C| \leq 2|C^*|$.
4. Soit M un couplage d'un graphe G. Rappeler le fonctionnement de l'algorithme de recherche de chemin M-augmentant dans G. Quelle condition G doit-il vérifier ? Donner un contre-exemple si ce n'est pas le cas.
5. Soit $M = (m_{i,j})$ une matrice de taille $n \times n$. On définit le permanent de M :

   $$\text{per}(M) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} m_{i,\sigma(i)}$$

   où S_n est l'ensemble des permutations de $\{1, \ldots, n\}$.
   Définir un graphe G dont le nombre de couplages parfaits est égal à $\text{per}(M)$.
   Remarque : il n'existe pas d'algorithme efficace pour calculer le permanent d'une matrice, contrairement au déterminant qui peut être calculé en $O(n^3)$ avec l'algorithme du pivot de Gauss.