

# Interrogation Écrite de Fin de Semestre ISN3

## 2<sup>ème</sup> année - Février 2025



**Durée totale :** 1h30

**Documents autorisés :** Synthèse personnelle de 2 pages format A4 (1 feuille).

- Le barème est sur 19 points, auxquels s'ajoutent :
  - 1 point issu de l'évaluation par les pairs de l'APP
  - un modificateur lié aux parcours Pix : de -0,5 à +0,5 points
  - un modificateur lié à l'évaluation par les pairs : de -0,5 à +0,5 points
- Le sujet est sur 11 pages - il y a 3 exercices.
- **Rappel :** Toute fonction demandée dans une question à le droit d'être utilisée dans une autre question, même si elle n'a pas été écrite.

Nom :

Prénom :

Groupe :

### Exercice 1 : Questions de cours (3 pts)

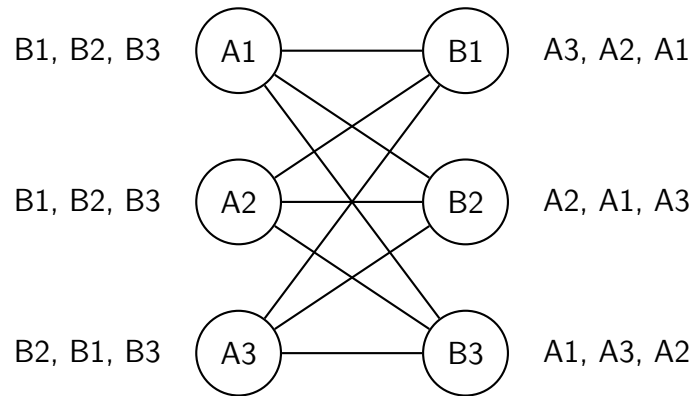
(1.1) *Donnez la définition d'un arbre (0.25pt)*

(1.2) *Donnez la définition d'un arbre couvrant d'un graphe (0.25pt)*

(1.3) *Donnez le nombre d'arêtes d'un arbre à  $n$  sommets (0.25pt)*

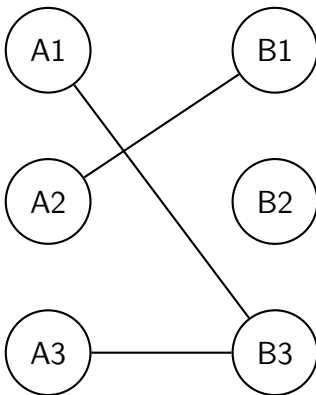
(1.4) *Donnez le nombre d'arêtes d'un graphe non orienté complet à  $n$  sommets. Justifiez (0.5pt)*

Voici un graphe biparti sur lequel on veut calculer un appariement avec l'algorithme de Gale-Shapley. Les préférences se lisent ainsi :  $A1$  préfère être affecté à  $B1$ , puis à  $B2$ , puis à  $B3$ , tandis que  $B2$  préfère être associé à  $A2$ , puis à  $A1$ , et sinon à  $A3$ .

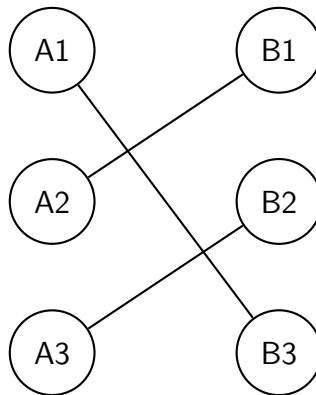


(1.5) Pour chacun des 3 cas suivants **nommez la propriété** de l'algorithme de Gale-Shapley qui n'est pas respectée et expliquez brièvement pourquoi. (0.25 par cas)

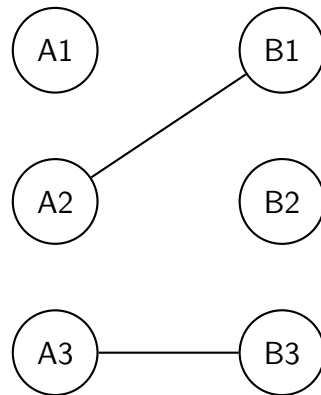
Cas 1



Cas 2

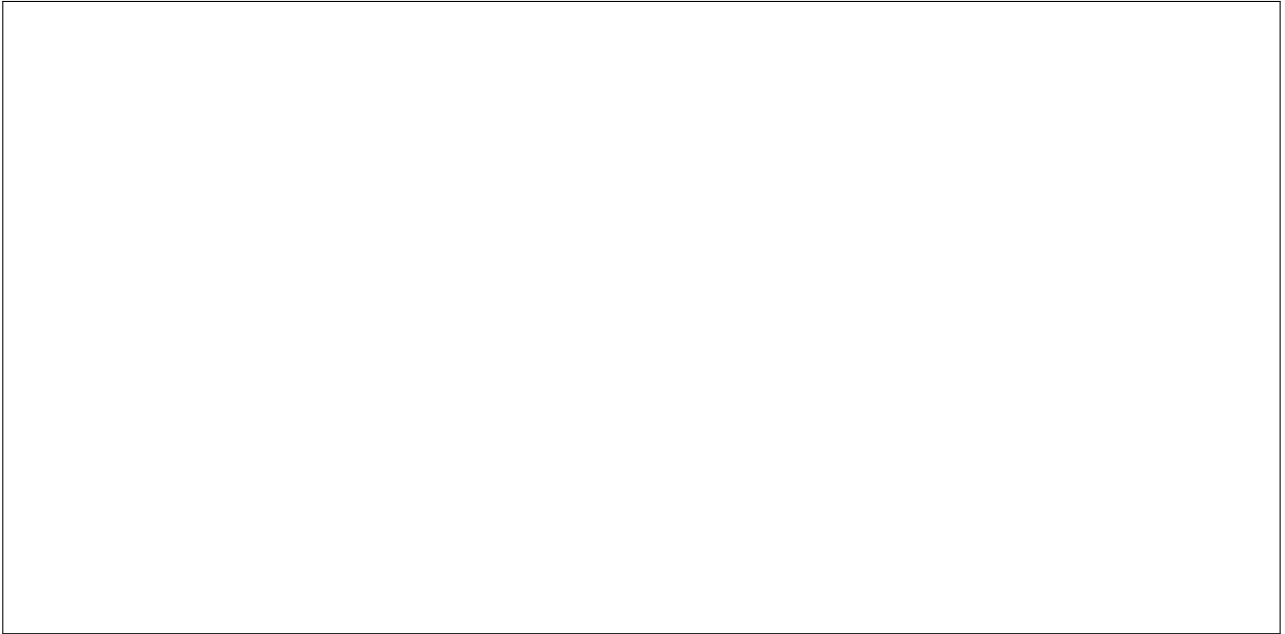


Cas 3



---

*(1.6) Dessinez l'affectation que calcule l'algorithme de Gale-Shapley (1pt)*



---

## Exercice 2 : Épidémiologie (9 pts)

On cherche à étudier la propagation d'une maladie au sein d'un groupe de personnes ayant des interactions plus ou moins nombreuses entre elles. Une personne malade va contaminer les personnes saines avec qui elle a des interactions. Plus il y a d'interactions, plus la contamination est rapide.

Un graphe modélise cette notion d'interactions : chaque sommet représente une personne, chaque arête représente l'existence d'interactions entre 2 personnes et la pondération représente la quantité des interactions entre ces 2 personnes. La figure 1 est un exemple d'interactions entre 6 personnes (A à F). Dans cet exemple, on pourra lire, par exemple, que la personne D a plus d'interactions avec C (14) qu'avec A (8), E (8) ou F (7), et n'en a pas avec B (pas d'arête).

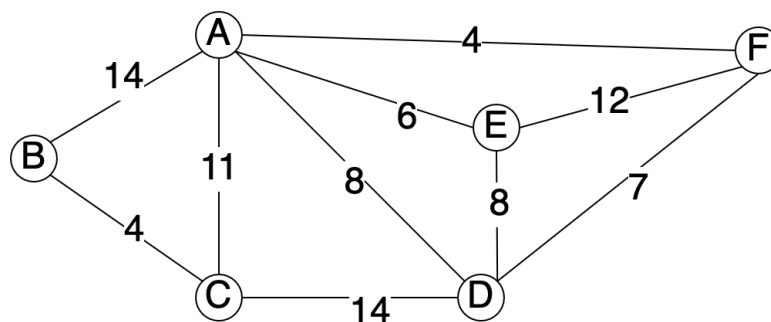


FIGURE 1 – Exemple de graphe d'interactions entre 6 personnes

On considère qu'un graphe pondéré non orienté connexe décrivant un tel système est stocké sous forme d'un dictionnaire d'adjacence dans la variable `graphe`.

Soit la fonction `nb_tours_epidemie` suivante :

---

```
1 def nb_tours_epidemie(graphe, depart):
2     malades = [depart]
3     nb_tours = 0
4     while len(malades) < len(graphe):
5         nb_tours += 1
6         contamines = []
7         for sommet in malades:
8             voisins = graphe.get(sommet, {})
9             v_max = None
10            poids_max = -math.inf
11            for v, p in voisins.items():
12                if v not in malades and v not in contamines and p > poids_max:
13                    v_max = v
14                    poids_max = p
15            if v_max is not None:
16                contamines.append(v_max)
17        malades = malades + contamines
18    return nb_tours
```

---

- 
- (2.1) Déroulez ce programme sur le graphe de la figure 1, avec *depart* = A. Écrivez le contenu de la variable *malades* à chaque fin de tour de la boucle *while*. Quelle est la valeur *nb\_tours* renvoyée à la fin du déroulé sur cet exemple ? (1 pt)

- (2.2) Expliquez ce que calcule cette fonction et ce que font ses différentes parties (ses différentes boucles) **en utilisant le vocabulaire du problème (personnes, interaction, malade, sain, contamination, etc.)** (1 pt)

- (2.3) Combien de fois la boucle *while* est exécutée dans le pire des cas (dans un graphe pondéré connexe non orienté à  $n$  sommets) ? Justifiez. (0,5pt)

- 
- (2.4) Écrivez la fonction `centre_epidemie` prenant un graphe d'interactions en paramètre, et qui cherche (et renvoie) le sommet de départ minimisant le nombre d'étapes nécessaires pour contaminer tout le graphe. (1,5pt)

**Vous n'avez pas besoin des réponses aux questions précédentes pour faire la suite.**

Au sein de la population, on connaît une liste des personnes qui sont vaccinées et une liste de personnes qui sont fragiles. Nous faisons l'hypothèse que le vaccin est fiable à 100% (une personne vaccinée ne peut ni être malade, ni contaminer quelqu'un). Chaque fois que le centre de contrôle des épidémies est notifié de l'apparition d'une personne malade, il souhaite vérifier s'il y a un risque qu'une personne fragile soit contaminée.

**Zone de réponse pour les deux prochaines questions sur la page suivante..**

- (2.5) Écrivez une fonction `danger_contamination` qui prend en paramètre un graphe des interactions, un sommet malade, une liste de sommets vaccinés et une liste de sommets fragiles, et qui renvoie un booléen valant `True` si au moins une personne fragile peut-être contaminée par propagation de cette maladie à partir de la personne malade (et renvoyant `False` sinon). Cette fonction pouvant être exécutée sur de très grands graphes, vous ferez attention à l'optimisation en temps d'exécution de votre solution. (2pts)
- (2.6) Listez les optimisations que vous avez faites dans votre code et expliquez pourquoi cela réduit le temps d'exécution sur un grand graphe. (3pts)



---

### **Exercice 3 : État civil (7 pts)**

Dans cet exercice, on appellera *État civil* d'une ville la mémoire de l'ensemble des personnes qui y sont nées. Ces informations sont stockées dans une structure de dictionnaires.

Chaque personne a un identifiant qui est la clé du dictionnaire et est décrite par les champs suivants : *nom*, *prenom*, *date\_naissance*, *id\_parent\_1*, *id\_parent\_2*. Un *id\_parent* qui vaut *None* signifie que le parent correspondant n'est pas connu.

Exemple de dictionnaire d'état civil :

---

```
1 {'1': {'nom': 'Curie', 'prenom': 'Marie', 'date_naissance': '07/11/1867',  
2       'id_parent_1': None, 'id_parent_2': None},  
3  
4   '2': {'nom': 'Curie', 'prenom': 'Pierre', 'date_naissance': '15/05/1859',  
5       'id_parent_1': None, 'id_parent_2': None},  
6  
7   '3': {'nom': 'Joliot-Curie', 'prenom': 'Irène', 'date_naissance': '12/09/1897',  
8       'id_parent_1': '1', 'id_parent_2': '2'},  
9  
10  '4': {'nom': 'Joliot', 'prenom': 'Frédéric', 'date_naissance': '19/03/1900',  
11      'id_parent_1': '2', 'id_parent_2': None},  
12  
13  '5': {'nom': 'Joliot', 'prenom': 'Pierre', 'date_naissance': '12/03/1932',  
14      'id_parent_1': '3', 'id_parent_2': '4'}  
15 }
```

---



---

Dans ce format, l'état civil ne contient que les id des parents. On souhaite pouvoir compléter cet état civil avec la liste des enfants connus.

- (3.1) *Écrivez une fonction `maj_enfants` qui prend en paramètre un état civil et qui met à jour la liste des enfants. Cette fonction ajoute **ou complète** la liste des id des enfants de chaque personne dans l'état civil. Si besoin, cette fonction ajoute une nouvelle clef dans le dictionnaire de chaque personne ayant des enfants (2 pts)*

---

(3.2) Écrivez une fonction *coparents* qui prend en paramètre un état civil et l'identifiant d'une personne, et qui renvoie la liste (sans doublon) des identifiants des personnes avec qui elle a eu des enfants. (2 pt)

---

(3.3) Écrivez une fonction *ancetres* qui prend en paramètre un état civil et un id, et renvoie la liste sans doublon des ids de tous ses ancêtres (parents, grands parents, grand-grands parents, ...). (3pts)