**Rattrapage : Algorithmique avancée et Complexité**

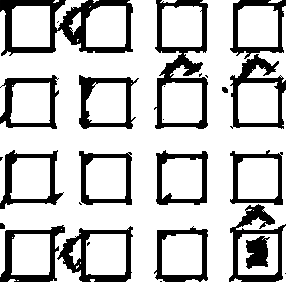
**Exercice 1 : (08 pts)**

On s’intéresse à l’implémentation d’un processus de tri sur la structure d’une liste chainée. Le processus de tri devra mettre à jour les chainages des éléments de la liste lorsque l’ordre n’est pas respecté.

* Décrire clairement la méthode de tri à adopter et l’ordre de complexité associé **(2 pts)**
* Ecrire un algorithme qui implémente cette méthode de tri sur la structure d’une liste chainée **(4 pts)**
* Selon vous, est-ce que l’ordre de complexité du tri dépend de la nature de la structure sur laquelle il est implémenté (contigüe ou chainée) ? Justifier votre réponse. **(2 pts)**

**Exercice 2 : (12 pts)**

Considérer le jeu du Futoshiki, qui signifie non égal en japonais. Il est basé sur une grille carrée dans laquelle doivent être inscrits des nombres suivant quelques règles très simples.

* Sur une grille nxn, les nombres de 1 à n doivent être placés dans chaque ligne et chaque colonne, sans aucune répétition (Voir l’exemple pour une grille 4x4).
* Les signes "plus grand que" ou "plus petit que" entre les cases sont des indices qui doivent obligatoirement être respectés. Ce qui signifie que les lignes ou les colonnes comportant ces signes doivent respectées les inégalités.
* Au début du jeu, quelques chiffres sont révélés - à vous de découvrir le reste. Vous devez respecter les inégalités entre les carrés et chaque grille possède une solution unique.

Formellement, le problème peut être défini comme suit :

* Une grille carrée de dimension nxn. Certains chiffres à certaines positions sont initialement révélés et des contraintes ‘ <’, ‘ >’ sont placées entre les cases.
* Existe-il un remplissage de la grille par des valeurs de 1 à n sans répétition et en respectant les valeurs initialement révélés et les inégalités ?

**Questions :**

1. Décrire les étapes de construction d’une solution en spécifiant la modélisation la plus adéquate **(5 pts)**
2. Estimer approximativement la taille de l’arbre de résolution et en déduire l’ordre de complexité de l’algorithme dans le pire cas **(2 pts)**
3. Quels sont les critères que doit satisfaire une solution donnée S’ pour être valide **(1 pt)**
4. Proposer un algorithme de validation d’une solution S et calculer sa complexité. **(4 pts)**