Rappel : Pour toute paire de fonctions $f(n)$ et $g(n)$ sur $\mathbb{N}$ asymptotiquement positives ou nulles, par définition $f \in O(g)$ si $\exists c > 0\ \exists n_0 \in \mathbb{N}\ \forall n > n_0$, $f(n) \leq cg(n)$. Aussi par définition, $f \in \Omega(g)$ si $g \in O(f)$.

Questionnaire

Répondre pour chaque question **V** (vrai) ou **F** (faux). Aucune justification n'est demandée mais les réponses fausses seront pénalisées.

1 Pour toute paire de fonctions $f(n)$ et $g(n)$ sur $\mathbb{N}$ asymptotiquement positives ou nulles, on a : Si $g \in O(f)$ alors $f(n) + g(n) \in \Omega(f(n))$.

2 Toute solution de la récurrence $T(n) = 3T(n/3) + O(\sqrt{n})$ appartient à $\Omega(n \log(n))$.

3 Dans le pire des cas, le temps d'exécution de *Quicksort* sur un tableau de taille n est $\Omega(n^2)$.

4 Trier par comparaisons un tableau de n valeurs prises dans $0 \ldots 9$ requiert $\Omega(n \log n)$ comparaisons.

5 Dans un graphe, l'arbre de recouvrement de poids minimal est unique lorsque toutes les arêtes portent des poids différents.

6 Un arbre binaire de recherche sur n entiers peut être construit en temps $O(n)$.

7 Dans une version probabiliste de *Quicksort*, on applique au tableau fourni en entrée une permutation aléatoire (tirée uniformément dans l'ensemble des permutations sur n éléments). Dans le pire des cas, l'espérance du temps de calcul de *Quicksort* probabiliste est $\Omega(n^2)$.

8 Dans un graphe orienté arbitraire, on peut trouver un tri topologique en temps linéaire.

9 On dispose de trois listes triées à trois éléments, on veut trouver le $3^{\text{ème}}$ plus petit élément dans l'ensemble constitué par la réunion de ces trois listes. On peut faire cela en 3 comparaisons.

10 En réduisant le problème du tri à la construction d'un tas (*heap*), on peut prouver que construire un tas sur n éléments requiert $\Omega(n \log n)$ comparaisons.

Toutes les questions sont indépendantes.

Problème 1

Donner un algorithme efficace pour résoudre le problème suivant.

Donnée : Un ensemble de n paires de skis est offert à un groupe de m skieurs ($m \leq n$). L_i désigne la hauteur de la $i^{\text{ème}}$ paire de skis, et ℓ_j désigne la hauteur du $j^{\text{ème}}$ skieur. A priori, on souhaiterait que chaque skieur reçût une paire de skis de même hauteur que lui. Comme ce n'est pas toujours possible, on se contente de minimiser les écarts en répondant à la question suivante.

Question : Trouver une affectation des skis aux skieurs (i.e. une fonction injective ϕ de $\{1 \ldots m\} \to \{1 \ldots n\}$) telle que $\sum_i |L_{\phi(i)} - \ell_i|$ soit minimale.

Remarque : Chaque skieur doit recevoir une paire de skis.

Suggestions : Résoudre d'abord le problème lorsque $m = n$; puis analyser les sous-problèmes $A(k, h)$ où on considère les h plus petits skieurs et les k plus courts skis.

Problème 2

Dans une graphe orienté sans boucle, à n sommets, une *source* est un sommet de degré entrant nul, et de degré sortant $n - 1$.

Donner un algorithme qui examine $O(n)$ entrées de la matrice d'adjacence pour décider si un graphe sans boucle possède une source.
Est-il nécessaire d'examiner $\Omega(n)$ entrées?

Problème 3

Dans un graphe orienté G, un *circuit* est une suite de sommets $v_0, v_1 \ldots v_k$ telle que $v_0 = v_k$ et $\forall i, 0 \leq i < k \quad (v_i, v_{i+1})$ est un arc de G. Le circuit est *impair* si k est impair. Donner un algorithme efficace (polynomial) pour résoudre le problème suivant:

Donnée : Un graphe orienté décrit par sa matrice d'adjacence.
Question : Le graphe contient-il un circuit impair ?

Remarque : on se demandera quelle est la relation entre l'existence de chemins de longueur k entre deux points et les puissances de la matrice d'adjacence.

Problème 4

Un algorithme détermine le maximum dans un tableau d'éléments non triés. Pour cela il utilise une boîte noire qui effectue les comparaisons entre éléments et lui retourne le résultat. Lors de la conception de l'algorithme, on apprend que la boîte noire peut retourner un résultat faux (mais pas plus d'une fois au cours du déroulement de l'algorithme).

Montrer que $2n - 1$ appels à la boîte noire sont nécessaires et suffisants si on veut garantir que l'algorithme détermine toujours le maximum, quelque soit le tableau d'entrée et la défaillance de la boîte noire.

Problème 5

Dans un graphe non orienté, $G = (V, E)$, un *cycle* est un ensemble d'arêtes du graphe $\{(v_0, v_1), (v_1, v_2) \ldots (v_{k-1}, v_0)\}$. (Cette définition est équivalente dans un sens très fort à celle du problème 3, mais elle est plus adaptée aux questions qui suivront). Un cycle est *simple* si $|\{v_0, v_1, \ldots, v_{k-1}\}| = k$. Le cycle est *pair* s'il contient un nombre pair d'arêtes (k est pair). L'ensemble vide désigne le cycle vide.

Deux cycles sont *arête-disjoints* s'ils sont disjoints en tant qu'ensembles d'arêtes. La *différence symétrique* de deux ensembles est formée par leur réunion moins leur intersection.

On note $\mathcal{P}(E)$ l'ensemble des ensembles d'arêtes. Par définition soit $\mathcal{C}$ le sous-ensemble de $\mathcal{P}(E)$ constitué par les ensembles d'arêtes formés par des réunions de cycles arête-disjoints. On admettra que $\mathcal{C}$ muni de la différence symétrique et de la multiplication par $\{0, 1\}$ est un espace vectoriel sur $\{0, 1\}$ (c'est l'espace des cycles, sous-espace de $\mathcal{P}(E)$ muni des mêmes opérations).

1) Donner un algorithme polynomial en $|E|$ qui construit une base de l'espace des cycles.

2) Donner un algorithme polynomial qui détermine si un graphe non orienté contient un cycle simple pair.

Problème 6

La médiane d'un ensemble $\{x_1, x_2 \ldots x_n\}$ de n éléments distincts d'un ensemble totalement ordonné est l'indice i, $1 \leq i \leq n$, tel que

$$|\{j \quad ; \quad x_j < x_i\}| = \lfloor n/2 \rfloor \qquad \text{et} \qquad |\{j \quad ; \quad x_j \geq x_i\}| = \lceil n/2 \rceil.$$

Quel est l'ordre de grandeur du nombre de comparaisons realisées par un algorithme optimal pour trouver la médiane de l'union de deux ensembles

déjà triés de taille commune n?

Vous devez donner un algorithme calculant cette médiane et réalisant un nombre de comparaisons de cet ordre de grandeur et un argument qui explique pourquoi cet ordre de grandeur est optimal.