Nom : ..... **Prénom :** .....

INFO I31, partiel 2012

Pour chaque question, répondez directement sur la feuille. Quand vous avez le choix, il n'y a qu'une seule bonne réponse par question. Lisez et comprenez les questions avant d'y répondre !

Question 1 – Déroulez l'algorithme d'Euclide pour calculer le PGCD de 156 et 180 :

Question 2 – Un étudiant programme une méthode de tri. Son programme met 1 seconde pour trier 100 mille éléments, et 4 secondes pour trier 200 mille éléments. La complexité de son algorithme est :

- ☐ $O(n^2)$, où n est le nombre d'éléments à trier
- ☐ $O(n)$, où n est le nombre d'éléments à trier
- ☐ $O(n \log n)$, où n est le nombre d'éléments à trier
- ☐ aucune des réponses précédentes n'est correcte.

Question 3 – Un étudiant programme une méthode de calcul de l'enveloppe convexe de n points en 2D. Son programme met 1 seconde pour $n = 100$ mille points, et 8 secondes pour $n = 200$ mille points. Son algorithme est en :

- ☐ $O(n^3)$; il s'agit vraisemblablement de la méthode naïve qui teste toutes les paires de points pour décider si elle donne une arête de l'enveloppe convexe.
- ☐ $O(n^2)$
- ☐ $O(n \log n)$
- ☐ aucune des réponses précédentes n'est correcte.

Question 4 – Un de ces problèmes est difficile. Lequel :

- ☐ trouver un chemin qui passe par tous les sommets d'un graphe
- ☐ trouver un chemin qui passe par toutes les arêtes d'un graphe

Question 5 – Quand un problème est dans NP, alors :

- ☐ il est difficile de décider si une proposition de solution est une solution
- ☐ il est facile de décider si une proposition de solution est une solution

Question 6 – Y a t-il des problèmes impossibles à résoudre en informatique ? :

- ☐ oui, en voici deux :
- ☐ non, il n'y a pas de problème insoluble, seulement des problèmes difficiles, qui exigent beaucoup de temps pour être résolus

Question 7 – Un algorithme optimal de tri, n'utilisant que des comparaisons entre 2 éléments, nécessite :

- ☐ $O(n \log n)$ comparaisons pour trier n éléments
- ☐ $O(n^2)$ comparaisons
- ☐ $O(n)$ comparaisons
- ☐ un nombre exponentiel de comparaisons

Question 8 – Calculer le produit scalaire entre 2 vecteurs de n nombres flottants $u = (u_i)$ et $v = (v_i)$ nécessite :

- ☐ $O(\log n)$ opérations flottantes

(tournez la page s.v.p)

- ☐ $O(n)$ opérations flottantes
- ☐ $O(n \log n)$ opérations flottantes
- ☐ $O(n^2)$ opérations flottantes

Question 9 – Calculer le produit entre une matrice carrée quelconque de taille $n \times n$, contenant n^2 nombres flottants, et un vecteur colonne de n éléments (n nombres flottants) nécessite :

- ☐ $O(n \log n)$ opérations flottantes
- ☐ $O(n)$ opérations flottantes
- ☐ $O(n^2)$ opérations flottantes
- ☐ ce n'est pas toujours possible

Question 10 – Calculer le produit entre deux matrices carrées de taille $n \times n$, en appliquant la formule : $C_{lc} = \sum_{k=1}^{n} A_{lk} \times B_{kc}$, nécessite :

- ☐ $O(n \log n)$ opérations flottantes
- ☐ $O(n)$ opérations flottantes
- ☐ $O(n^2)$ opérations flottantes
- ☐ $O(n^3)$ opérations flottantes
- ☐ ce n'est pas toujours possible (la matrice doit être inversible, par exemple).

Question 11 – La méthode de puissance rapide pour calculer M^k (M est une matrice carrée de taille $n \times n$, et k est un entier naturel) nécéssite :

- ☐ inévitablement $k-1$ (donc $O(k)$) multiplications de matrices carrées de taille $n \times n$
- ☐ $O(\log k)$ multiplications de matrices carrées de taille $n \times n$
- ☐ $O(k \log k)$ multiplications de matrices carrées de taille $n \times n$
- ☐ ce n'est pas toujours possible (la matrice doit être inversible, par exemple).

Question 12 – Trouver l'élément le plus petit dans une liste non ordonnée de taille $n > 0$ nécessite :

- ☐ $O(1)$ comparaisons
- ☐ $O(n^2)$ comparaisons
- ☐ $O(n)$ comparaisons
- ☐ $O(\log n)$ comparaisons

Question 13 – Trouver l'élément le plus petit dans une liste triée par ordre croissant et de taille $n > 0$ nécessite :

- ☐ $O(1)$ comparaisons
- ☐ $O(n^2)$ comparaisons
- ☐ $O(n)$ comparaisons
- ☐ $O(\log n)$ comparaisons

Question 14 – Un étudiant programme le tri rapide ("quicksort") d'une liste L ainsi : si L contient moins de 2 éléments, alors elle est déjà triée. Sinon, l'étudiant choisit un élément p dans L. Il partitionne L en 3 sous listes, la liste L_1 des éléments dans L inférieurs à p, la liste L_2 des éléments dans L égaux à p, la liste L_3 des éléments dans L supérieurs à p. Il trie récursivement les 3 listes, puis concatène les résultats. Qu'en pensez-vous :

- ☐ la méthode est correcte mais la complexité est modifiée
- ☐ la méthode est correcte et la complexité est inchangée
- ☐ la méthode est incorrecte ; elle boucle quand L contient deux (ou davantage) éléments égaux à p.
- ☐ la méthode est incorrecte car L_2 n'est jamais vide, puisqu'elle contient p.

Question 15 – Pour le calcul de l'arbre couvrant minimum d'un graphe connexe, un étudiant propose l'algorithme suivant : si le graphe est un arbre, alors insérer cet arbre dans l'arbre couvrant minimum ; sinon décomposer le graphe G en 2 graphes de taille à peu près moitié, G_1, et G_2, tous deux connexes ; calculer l'arbre

couvrant minimum de G_1 ; calculer l'arbre couvrant minimum de G_2 ; joindre ces deux arbres par l'arête de coût minimum entre G_1 et G_2 (cette arête a un sommet dans G_1 et un sommet dans G_2). Que pensez-vous de cet algorithme :

- ☐ il est correct
- ☐ il est difficile de partitionner un graphe connexe en deux sous graphes connexes ayant (à peu près) deux fois moins de sommets ; c'est pourquoi cet algorithme n'est pas utilisé
- ☐ il est incorrect, et voici un contre exemple simple (au plus 4 sommets !) :

Question 16 – Une matrice de Vandermonde a la structure suivante :

$$M = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w & w^2 & \dots & w^{n-1} \\ 1 & w^2 & w^4 & \dots & w^{2(n-1)} \\ \dots & \dots & \dots & \dots & \\ 1 & w^{n-1} & w^{2n-2} & \dots & w^{(n-1)^2} \end{pmatrix}$$

où n est une puissance de 2. Si de plus w est une racine n ième de l'unité, alors le produit avec un vecteur colonne quelconque de taille n :

- ☐ peut se faire en $O(n \log n)$ (avec l'algorithme de la transformée rapide de Fourier)
- ☐ ne peut pas se faire en moins que $O(n)$
- ☐ ne peut pas se faire en moins que $O(n^2)$

Question 17 – Pour $n = 8$, écrire la ligne de la matrice de Vandermonde commençant par 1 puis w^3, en simplifiant (c'est à dire en éliminant les puissances plus grandes que 7) :

Question 18 – Idem, pour $n = 8$, écrire la ligne de la matrice de Vandermonde commençant par 1 puis w^6 :

Question 19 – Dans le problème de la somme, un ensemble E de n entiers positifs $e_1 \geq e_2 \geq \ldots \geq e_n$ est donné, ainsi qu'un entier $0 < S < \sum_1^n e_i$. Il faut trouver le sous ensemble X de E dont la somme des éléments est maximum, mais inférieure ou égale à S. Pour tout ensemble K, on note $s(K)$ la somme des éléments de K. L'algorithme glouton calcule $X_0 = \emptyset$ (donc $s(X_0) = 0$) ; puis, pour i de 1 à n, il calcule $X_i = X_{i-1} \cup \{e_i\}$ si $e_i + s(X_{i-1}) \leq S$, et $X_i = X_{i-1}$ sinon. Avec $E = 100, 25, 20, 10, 1, 1, 1$ et $S = 40$, donnez les valeurs de $X_0, X_1, \ldots X_7$, et les $S(X_i)$ correspondants :

Question 20 – Cet algorithme donne-t-il un sous ensemble X_n :

- ☐ correct ($s(X_n) \leq S$), mais pas forcément optimal : donnez un exemple simple ($n < 5$) où X_n n'est pas optimal
- ☐ correct, et optimal
- ☐ pas toujours correct, mais optimal : donnez un exemple simple ($n < 5$)

- ☐ souvent correct, et souvent optimal
- ☐ ni correct ni optimal : donnez un exemple simple ($n < 5$)
- ☐ toutes les réponses précédentes sont fausses

Question 21 – Arthur conjecture que si, dans un graphe non orienté, tous les sommets distincts soit sont voisins, soit ont un voisin en commun, alors il existe au moins un sommet qui est voisin de tous les autres. Que pensez vous de cette conjecture :

- ☐ elle est vraie
- ☐ elle est fausse et vous en dessinez un contre exemple simple