INFO I31, partiel, 18 Nov 2014

RÉPONDEZ AUX 23 QUESTIONS DANS L'ORDRE ; MENTIONNEZ LE NUMÉRO DE LA QUESTION DEVANT CHAQUE REPONSE (EVENTUELLEMENT VIDE).

VOUS AVEZ DROIT A TOUS VOS DOCUMENTS PERSONNELS.

TOUT INSTRUMENT ELECTRONIQUE EST INTERDIT.

ECRIVEZ LISIBLEMENT. MERCI !

IL Y A 23 QUESTIONS, NUMEROTEES DE 1 A 23.

Question 1 – Déroulez l'algorithme d'Euclide pour calculer le PGCD de 325 et 240 (en utilisant la division euclidienne, et pas la soustraction). Dans la dernière ligne, a ou b est nul, et la case r est vide. Entourez la valeur du pgcd. Le tableau solution a 7 lignes (sans compter la ligne a, b, r).

a	b	r
325	240	

Question 2 – Soient $a \geq b > 0$ deux entiers donnés. L'algorithme d'Euclide étendu calcule g le PGCD de a et b ainsi que les plus petits entiers u, v tels que $au + bv = g$. Dans la programmation récursive, la fonction est récursivement appelée sur $a' = b$, et $b' = a \mod b$; elle rend g', u' et v' tels que $a'u' + b'v' = g' =$ PGCD(a', b'). Donnez, sans démonstration, les formules pour calculer g, u, v en fonction de g', u', v'. Vous pouvez utiliser q le quotient (entier) de a par b.

Question 3 – Déroulez l'algorithme d'Euclide étendu (ou Bézout) pour $a = 325$ et $b = 240$. u et v sont tels que $au + bv = g = PGCD(a, b)$; $r = a \mod b$, et $q = \lfloor \frac{a}{b} \rfloor$ est le quotient entier de la division de a par b Dans la dernière ligne, a ou b est nul, et les cases q et r sont vides.

a	b	r	q	g	u	v
325	240					

Question 4 – Citez les noms de 3 algorithmes optimaux de tri

Question 5 – Un étudiant programme une méthode de tri. Son programme met 1 seconde pour trier 100 mille éléments (10^5), et 4 secondes pour trier 200 mille éléments (2×10^5). Notons n le nombre d'éléments à trier. La complexité de son algorithme est en ?

☐ $O(n^2)$

☐ $O(n)$

(tournez la page s.v.p)

- ☐ $O(n \log n)$
- ☐ autre, laquelle ?

Question 6 – Quel est l'ordre de grandeur du nombre de comparaisons effectuées par un algorithme optimal de tri (qui n'utilise que des comparaisons entre 2 éléments)

Question 7 – Donnez les formules nécessaires pour le calcul récursif de a^n ($a \in \mathbb{R}, n \in \mathbb{N}$). N'oubliez pas le ou les cas terminaux.

Question 8 – Citez les noms de 3 algorithmes calculant les plus courts chemins dans un graphe

Question 9 – La difficulté d'un problème NP vient

- ☐ de ce qu'il y a beaucoup de candidats (de solutions potentielles)
- ☐ générer un candidat prend beaucoup de temps
- ☐ tester si un candidat est solution prend beaucoup de temps

Question 10 – Y a t-il des problèmes impossibles à résoudre en informatique ?

- ☐ oui ; proposez en deux.
- ☐ non. Il y a seulement des problèmes difficiles, qui nécessitent beaucoup de temps pour être résolus.

Question 11 – Un tableau contient n entiers triés par ordre croissant ; combien de comparaisons nécessite la méthode optimale (sans utiliser d'autre structure de données) pour décider si le tableau contient un entier donné

Question 12 – Quelle structure de données utiliser pour répondre en temps constant à la requête précédente ?

Question 13 – Un tableau, trié, contient n entiers ; combien de comparaisons nécessite la méthode optimale pour trouver le nombre maximum dans le tableau qui est inférieur (strictement) à un nombre donné, ou pour détecter qu'il n'existe pas

Question 14 – La méthode la plus rapide pour calculer le n ième terme de la suite de Fibonacci, F_n (rappel : $F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2}$ quand $n > 1$) nécessite

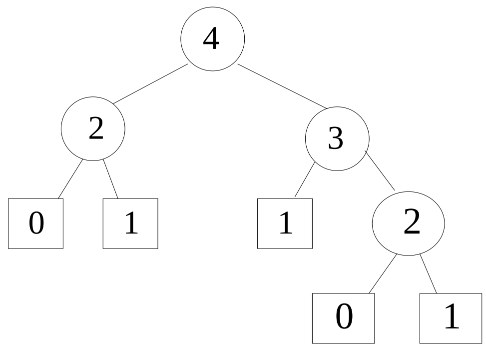
- ☐ $O(\log n)$ opérations sur des entiers
- ☐ $O(n)$ opérations sur des entiers
- ☐ $O(F_n)$ opérations sur des entiers

Question 15 – Le calcul rapide du membre droit de

$$(F_k, F_{k-1}) = (F_1, F_0) \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{k-1}$$

permet de calculer F_k, le k ième terme de la suite de Fibonacci. Pour calculer K_n, où K_0, K_1 sont donnés, et $K_n = aK_{n-1} + bK_{n-2}$ quand $n > 1$, avec a et b donnés, quelle formule matricielle utiliseriez-vous ?

Question 16 – Pour calculer G_n, où G_0, G_1 sont donnés, et $G_n = g_0 + g_1 G_{n-1} + g_2 G_{n-2}$ quand $n > 1$, avec g_0, g_1 et g_2 donnés, quelle formule matricielle utiliseriez-vous ?



Question 17 – L'arbre de Fibonacci T_4 est dessiné ci-dessus. T_0 est une feuille portant l'étiquette 0, T_1 est une feuille portant l'étiquette 1 ; pour $n > 1$, T_n est un noeud binaire, dont le fils gauche est T_{n-2} et dont le fils droit est T_{n-1}. Donnez une formule récursive pour le nombre total de noeuds (feuilles ou noeuds internes), noté t_n, de T_n, pour $n > 1$.

Question 18 – (suite) Donnez une formule récursive pour définir u_n le nombre de feuilles étiquetées 1 de T_n. Que constatez-vous ?

Question 19 – (suite) Donnez une formule récursive pour définir z_n le nombre de feuilles étiquetées 0 de T_n.

Question 20 – (suite) Remplissez le tableau suivant, où u_n est le nombre de feuilles étiquetées 1 de T_n, z_n le nombre de feuilles étiquetées 0 de T_n, et t_n le nombre de noeuds (feuilles ou non) de T_n.

n	0	1	2	3	4	5	6	7	8	9
t_n										
u_n										
z_n										

Question 21 – (suite) Donnez la formule matricielle pour calculer rapidement t_n.

Question 22 – Le temps d'exécution d'un algorithme est $T(n)$ pour une donnée de taille n, où $T(1) = 1$ et $T(n) = 3T(n/2) + n$. Prouvez en quelques lignes, par récurrence sur k, que $T(2^k) = 3^{k+1} - 2^{k+1}$. La formule est vraie pour $k = 0$, ce qui permet d'amorcer la récurrence.

Question 23 – (suite) La complexité d'un algorithme est $T(2^k) = 3^k, n = 2^k$. Déduisez-en la complexité de $T(n)$ en fonction de n ; vous la prouverez en une ligne ; les valeurs numériques de $\log_2 3$, ou autre, ne sont pas demandées.