EXAMEN I31 SESSION 2, Licence 2, 14 Juin 2017

Tous vos documents sont autorisés, mais pas la copie de votre voisin. N'écrivez aucun programme. Ecrivez lisiblement. Répondez aux questions dans l'ordre, en indiquant le numéro de chaque question. Attention : IL FAUT SAVOIR LIRE, ET IL FAUT LIRE L'ENONCE.

1. Citez un problème que nous avons résolu par programmation dynamique (en cours, TD ou TP).

2. Quelle est la solution de cette équation de récurrence : $T(n) = 2T(n/2) + O(n)$? (Indication : ce n'est pas $O(n^2)$). Citer un algorithme avec cette complexité.

3. Quelle est la solution de cette équation de récurrence : $T(n) = T(n/2) + O(n)$? (Indication : ce n'est pas $O(n^2)$). Citer un algorithme avec cette complexité.

4. Que fait la méthode de Dijkstra ? (2 lignes au plus)

5. Que fait la méthode de Newton ? (2 lignes au plus)

6. Un programme est en $O(n^2)$. Il met 1 seconde pour $n = 1000$. Combien de temps lui faut-il pour $n = 2000$?

7. On mesure le temps $t(n)$ mis par un programme, en faisant varier n, le nombre de données. Un ensemble de k paires $(n_1, t_1 = t(n_1))$, $(n_2, t_2 = t(n_2))$, $\ldots (n_k, t_k = t(n_k))$ est obtenu. Ensuite, les points $(x_i = \log n_i, y_i = \log t_i)$ sont affichés. En supposant que le programme met Cn^d unités de temps pour n données (C est une constante), donnez la relation entre les x_i et les y_i. Que peut-on dire sur les points (x_i, y_i) ? Se trouvent-ils sur une parabole, par exemple ?

8. Pour le tri par tas, quelle est l'équation récursive donnant le temps (ou

le nombre d'instructions élémentaires) $T(n)$ nécessaire pour trier n éléments ? Vous prendrez $T(0) = 1$.

9. Citez deux algorithmes qui trient des entiers sans les comparer.

10. Quelle est la complexité du tri naïf ?

11. (Sur 2 points) Un tableau de n valeurs entières (toutes différentes) : V[1], V[2], ... V[n] est donné. Vous devez calculer la longueur de la séquence croissante la plus longue. Les éléments de la séquence ne sont pas forcément consécutifs (comme dans l'exemple ci-dessous). Soit $M[k]$ la longueur de la séquence croissante la plus longue dans les k premières cases du tableau : V[1], V[2], ... V[k] ; cette séquence croissante la plus longue peut utiliser V[k] ; elle peut aussi ne pas l'utiliser. Voici un exemple :

i	1	2	3	4	5	6	7	8
V	100	40	50	200	60	80	300	90
M	1	1	2	3	3	4	5	5

où les séquences les plus longues sont de longueur 5, par exemple : $40, 50, 60, 80, 90$, qui utilise V[2], V[3], V[5], V[6], et V[8]. Donnez une formule mathématique (et pas un programme) pour calculer M[k+1]. Dans la formule, vous avez le droit d'utiliser $\max, \min, \max_{i=1}^{k}, +, <, >$ et : si ... alors ... sinon. Voici une formule fausse mais syntaxiquement correcte : M[k+1] := si (V[k] $<$ V[k+1]) alors 1 + M[k] sinon M[k]. Vous pouvez utiliser la notation V_i au lieu de V[i], M_k au lieu de $M[k]$, etc. Répondre en 2 lignes au plus.

12. (suite) Quelle est la complexité de l'algorithme précédent (pour remplir le tableau) ? Avec la formule fausse, il serait en $O(n)$.

13. (suite, sur 2 points) Pour l'exemple, dessinez un graphe orienté, dont tout chemin critique donne une séquence croissante de longueur maximale. Dessinez les sommets de gauche à droite : V[1], V[2], ... V[8]. Représentez chaque sommet par un cercle, entourant sa valeur V. N'oubliez pas de tracer les arcs. Vous pouvez ne pas tracer les arcs inutiles (de transition), pour alléger la figure. Ajoutez un sommet source et un sommet puits. La durée de tous les arcs est 1, sauf pour les arcs issus de la source ou arrivant au puits. Indiquez les dates au plus tôt et au plus tard de chaque sommet.

15. (Sur 2 points) En utilisant l'algorithme d'Euclide étendu, calculez le pgcd de 144 et 233 ainsi que les coefficients de Bezout. Utilisez la représentation habituelle en tableau.

16. Résoudre l'équation de récurrence $T(n) = 3T(n/2)$.