Sujet TP noté, Licence 2, module I31

Elisabeth Gavignet, Dominique Michelucci, Université de Bourgogne

Activité : Travail individuel sur machine (ordinateur de l'université ou ordinateur personnel en wifi)

Durée : 1 heure 45 + temps pour l'envoi par messagerie ou récupération sur clé USB éventuellement

Documents autorisés : CM, TD et TP de I31 exclusivement

Modalités d'évaluation : à chaque question résolue, vous demandez au surveillant de venir l'évaluer.

Remarques :

- Il est préférable de traiter les questions dans l'ordre où elles sont proposées. Ne pas savoir répondre à une question ne vous empêche pas de faire la suivante car soit elle ne sont pas dépendantes, soit il existe une solution de secours proposée dans le sujet.

- Si vous constatez un disfonctionnement sur un ordinateur de l'université, vous devez immédiatement avertir votre surveillant

- Des sauvegardes régulières, y compris sur un support externe, sont largement conseillées afin de ne pas perdre le fruit du travail réalisé.

- Inutile de passer trop de temps sur une partie que vous n'arrivez pas à réaliser. Le barème est là pour vous aider à gérer votre temps et par conséquent la note obtenue

Objectif du TP noté : Calculer la longueur de la séquence croissante la plus longue d'éléments stockés dans un tableau non trié d'entiers E[0], . . ., E[n−1] donné. Dans cette séquence, tout élément (sauf le dernier) est inférieur ou égal à son élément suivant, non forcément consécutif. Pour ce TP noté, vous supposerez que tous les éléments du tableau E sont différents.

Par exemple, si E = [0; 300; 100; 200; 1000; 400; 500;1100; 900; 800; 600; 700;−100], alors la séquence croissante la plus longue possède 7 éléments. Il s'agit de [0; 100; 200; 400; 500; 600; 700].

Question 1 (2 pts) : Construire un nouveau projet nommé « TPxxx » où xxx est votre nom de famille et recopier dans le dossier « src » de votre projet, le fichier « Liste.java » (celui utilisé dans les TP précédents) et le fichier « Sequence.java » qui se trouvent sur le serveur à l'endroit indiqué par votre surveillant. Le fichier « Sequence.java » contient le programme principal qui comporte déjà des instructions (certaines sont placées en commentaire dans la version téléchargée) utiles pour tester votre programme au fur et à mesure que vous réaliserez les questions.

Question 2 (2 pts) : Compléter la méthode « afficheTab(int [] t, String esp) » pour qu'elle affiche sur une même ligne tous les éléments du tableau passé en paramètre. Chaque valeur sera séparée de la précédente par le contenu de « esp » passé en paramètre. Un retour à la ligne sera toujours réalisé à la fin de l'affichage. Le tableau comporte toujours autant d'éléments que d'espace réservé.

Question 3 (1 pt) : L'appel à la méthode « remplirTab(lp) », mis en commentaire dans le programme principal, permet d'initialiser le tableau E à partir des éléments stockés dans la liste passée en paramètre. Modifier l'initialisation de cette liste afin qu'elle permette de remplir le tableau E avec les valeurs suivantes :

i	0	1	2	3
E[i]	0	300	100	200

Retirer les commentaires qui conviennent pour tester que votre programme affiche bien le tableau proposé ci-dessus.

Question 4 (2 pts) : Après avoir pris connaissance de la méthode (voir encadré ci-dessous) qui permet d'instancier LT, compléter la méthode « initLT(int n) » proposée dans le fichier « Sequence.java » dont l'objectif est de calculer, pour chacun des éléments du tableau, la longueur de la séquence croissante la plus longue qui se termine à chacun d'eux. La valeur passée en paramètre, à la méthode « initTL », représente la taille du tableau.

Il existe une méthode en temps polynomial (O(n2)) qui permet de calculer la longueur de la séquence croissante la plus longue qui se termine (et utilise) E[i] récursivement à partir de celles qui se terminent en E[0], E[1], …, E[i-1].

Soit LT[i], la longueur de la séquence croissante la plus longue qui se termine (et utilise) E_i.
La méthode pour calculer LT[i] est la suivante

```
pour chaque i Є [0..n ]
    LT [i] = 1
    pour k Є [0..i-1 ]
        si   E[k] < E[i]   et   LT[i] < LT[k] + 1   alors  LT[i] = LT[k] + 1
```

Exemple de référence :

i	0	1	2	3	4	5	6	7	8	9	10	11	12
E[i]	0	300	100	200	1000	400	500	1100	900	800	600	700	-100
LT[i]	1	2	2	3	4	4	5	6	6	6	6	7	1

Question 5 (2 pts) : Modifier le programme principal pour que ce dernier exécute la méthode « initLT » et affiche sur une première ligne le contenu de E avec 3 espaces entre chaque valeur et sur la ligne suivante le contenu de LT avec 5 espaces entre chaque valeur. Si vous n'avez pas réussi à rédiger la méthode « initLT » à la question 4, vous pouvez faire appel à la méthode « initLTsecours » pour pouvoir réaliser cette question et continuer le travail pratique demandé.

Question 6 (3 pts) : Rédiger la méthode récursive « rechDicho (int val, int [] t, int inf, int sup) qui réalise la recherche dichotomique de la valeur val dans le tableau t entre les bornes inf et sup. Cette méthode retourne vrai si la valeur recherchée existe et faux sinon.

Question 7 (2 pts) : La méthode « affichSeq(int lg) » affiche la première séquence croissante de longueur lg (valeur passée en paramètre) si elle existe. Elle fournit aussi un résultat dans le cas où la valeur transmise en paramètre n'existe pas. Modifier cette méthode pour que, dans ce cas, le message soit « aucune séquence de cette longueur ». Vous utiliserez pour cela la méthode rechDicho de la question précédente ou, si vous n'avez pas réussi à traiter la question 6, la méthode privée max présente dans le fichier « Sequence ».

Exemple1 à tester : affichSeq(6) donnera l'affichage suivant : 0 100 200 400 500 1100

i	0	1	2	3	4	5	6	7	8	9	10	11	12

E[i]	0	300	100	200	1000	400	500	1100	900	800	600	700	-100
LT[i]	1	2	2	3	4	4	5	6	6	6	6	7	1

qui correspond aux valeurs grisées ci-dessus 1, 2, 3, 4, 5 et 6 de LT en choisissant toujours la dernière occurrence en cas de valeurs multiples.

Exemple2 à tester : affichSeq(8) devrait donner : aucune séquence de cette longueur

Question 8 (2 pts) : La méthode « affichSeqRec (int li, int lg) » est une méthode récursive dont on souhaite avoir une trace. Ajouter aux bons endroits les éléments utiles afin de pouvoir afficher d'une part le nombre d'appels récursifs à affichSeqRec exécutés lors d'un appel à « affichSeq » et d'autre part les étapes de la constitution de la chaîne « res » qui est retournée par la méthode « affichSeqRec ».

Question 9 (3 pts) : En vous inspirant de la méthode « affichSeqRec (int li, int lg) », rédiger la méthode récursive « String affichRecTous(int li, int lg, String res) » qui affiche toutes les séquences croissantes de longueur maximale lg. Cette méthode est appelée par la méthode « String affichSeqTous(int l) » donnée ci-dessous :

```
public String affichSeqTous(int lg)
{   assert (lg>=0 && lg<E.length);
    return affichRecTous(1, lg, "sequence de taille "+lg+"\n");    }
```

Exemple à tester : affichSeqTous(6) donnera les affichages successifs suivants (aux espaces près) :

i	0	1	2	3	4	5	6	7	8	9	10	11	12
E[i]	0	300	100	200	1000	400	500	1100	900	800	600	700	-100
LT[i]	1	2	2	3	4	4	5	6	6	6	6	7	1

sequence de taille 6
0 100 200 400 500 1100

LT[i]	1	2	2	3	4	4	5	6	6	6	6	7	1

sequence de taille 6
0 100 200 400 500 900

LT[i]	1	2	2	3	4	4	5	6	6	6	6	7	1

sequence de taille 6
0 100 200 400 500 800

LT[i]	1	2	2	3	4	4	5	6	6	6	6	7	1

sequence de taille 6
0 100 200 400 500 600

Question 10 (1 pt) : Constituer une archive au format zip de votre dossier « TPxxx » que vous nommerez « TPxxx » où xxx est votre nom de famille. Envoyer un message à dominique.michelucci@u-bourgogne.fr en joignant votre archive et mettre en copie le surveillant (son adresse électronique est notée au tableau).