Algorithmique L3 LI

13 Mars 2008

Devoir surveillé numéro 1

Durée : 2h.

Le barême est donné à titre indicatif.

Mots Palindrômes et Carrés - 9 points

Dans cet exercice on supposera que les mots sont représentés par des chaînes de caractères (tableau de caratères). L'accès à un caractère du mot pourra se faire au choix par s[i] où par s.charAt(i).

1. Écrire une méthode booléenne qui prend en argument un mot et qui retourne vrai si le mot est un palindrôme et faux sinon. Un palindrôme est un mot qui peut se lire dans les deux sens (ex : "elle", "radar", "kayak" ou plus généralement "abbcacbba" sont des palindrômes). L'algorithme doit procéder sans utilisation de tableau supplémentaire.
2. Écrire une méthode booléenne qui prend un mot en argument et qui retourne vrai si le mot est un carré, faux sinon. Un mot carré est un mot de la forme "uu" (ex : "abbcabbc" est le mot carré de "abbc"). L'algorithme doit procéder sans utilisation de tableau supplémentaire.
3. Quelles sont les compléxités des méthodes que vous avez écrites en 1) et 2) (en fonction de la longueur du mot) ? Donner une explication succinte.

Le tri CombSort - 5 points

En avril 1991 paraît un article de Stephen Lacey et Richard Box montrant qu'une modification simple de l'algorithme de tri par remontée des bulles permet de le rendre quasiment aussi efficace que les algorithmes de tri en tas ou de tri rapide. Dans le tri par remontée des bulles chaque élément est comparé au suivant, et éventuellement échangé si les deux éléments ne sont pas dans l'ordre. Apparaissent alors ce que les auteurs appellent des "tortues", c'est à dire des éléments qui sont loin de leurs places, et qui y vont lentement par échanges successifs. Leur idée est d'éliminer les tortues : au lieu de comparer des éléments successifs, **on compare des éléments éloignés de tailleSaut**, et on les échange s'ils ne sont pas dans l'ordre. Alors un élément devrait aller plus vite à sa place. **A chaque étape, la valeur de tailleSaut est divisée par 1.3**. La valeur 1.3 a été obtenue expérimentalement. L'algorithme se termine, lorsqu'on a fait un passage avec **tailleSaut égal à 1** et **sans faire d'échanges** : alors tous les éléments sont à leur place.

Écrire une methode qui est une variante du tri à bulle et qui ordonne le tableau passé en argument en implémentant le tri CombSort décrit ci-dessus.

On déclarera la constante `final double coeff = 1.3 ;` et une variable `int tailleSaut ;`

Pour vous aider voici un rappel de l'implémentation du tri à bulle :

```
void triBulles(int t[]){
  boolean echange;
  int taille = t.length();
  echange = true;
  int i = 0;
  while(echange || i<taille){
        echange = false;
        for (int j=taille-1; j>i; j--){
          if (t[j+1] < t[j]) { // comparer deux voisins
            echanger(t[j+1],t[j]); // les echanger si necessaire
            echange=true;
          }
        }
i++;
    }
}
```

Listes chaînées - 6 points

Dans cet exercice, vous illustrerez vos raisonnements avec de petits graphiques représentant les opérations sur les listes et vous écrirez vos méthodes en pseudo-code sans rentrer dans les détails d'implémentation.

1. Rappeler les avantages et les inconvénients d'une implémentation par chaînage par rapport à une implémentation contigüe.
2. On a une liste chaînées d'entiers. On veut savoir si elle contient (au moins) un doublon, c'est-à-dire s'il existe un entier qui apparaisse deux fois. Écrire une méthode booléenne `doublon` qui prend en argument la liste chaînée et retourne vrai si un tel doublon existe, faux sinon. Pouvez-vous imaginer une version plus rapide si l'on suppose que la liste passé en argument est triée ?
3. Écrire une méthode `circulaire` prend en argument une liste chaînée et la rend circulaire. C'est-à-dire que le dernier élément de la liste pointe sur le premier élément.
4. Écrire une méthode `miroir` qui prend en argument une liste chaînée et la renverse. C'est-à-dire que le dernier élément devient le premier élément de la liste, l'avant-dernier le deuxième élément... et ainsi de suite la tête de la liste devenant la queue.
5. Quelles sont les compléxités respectives des méthodes `doublon`, `circulaire` et `renverse` ? Donner une explication succinte.