



Année 2022

Nom et prénom :

```
public abstract class LList<T> {
    public T head();
    public LList<T> tail();
    public boolean isEmpty();
    public LList<T> prepend(T elt);
    ...
    public LList<T> mystere(int b, int c) {
        LList<T> cur = this;
        LList<T> temp = new EmptyLList();
        for (int i = 0; i < b; i++) {
            temp = temp.prepend(cur.head());
            cur = cur.tail();
        }
        for (int i = 0; i < c; i++) {
            cur = cur.tail();
        }
        while (!temp.isEmpty()) {
            cur = cur.prepend(temp.head());
            temp = temp.tail();
        }
        return cur;
    }
}
```

FIGURE 1 – LList est une liste simplement chaînée immutable, identique à celle décrite dans le chapitre 2 du cours.

Question 1 Soit une classe représentant une liste simplement chaînée immutable de type `LList<T>`, comme celle du listing 1. Donner le code d’une méthode publique `LList<T> distinct()` de la classe `LList<T>`, renvoyant une liste contenant exactement un exemplaire de chaque valeur de la liste `this` (les doublons sont supprimés).

L’ordre des objets contenu dans `this` doit être conservé : si `this` contient des doublons, seul la première occurrence du doublon est conservée.

Votre méthode doit avoir une complexité *linéaire*. Utilisez le `HashSet` de l’API Java pour obtenir un code concis et efficace.

Répondez sur la copie séparée.

..... ☐1 ☐2 ☐3 ☐4 ☐5 ☐6

Question 2

Observez l’algorithme décrit sur le listing 1. Quelle est la complexité de l’algorithme ?

- | | | | |
|---------------------------------------------------|----------------------------------------------------|----------------------------------------------|------------------------------------------------|
| <input type="checkbox"/> $\Theta(b + c)$ | <input type="checkbox"/> $\Theta(c \cdot \log b)$ | <input type="checkbox"/> $\Theta(b)$ | <input type="checkbox"/> $\Theta(\log(b + c))$ |
| <input type="checkbox"/> $\Theta(b \cdot \log c)$ | <input type="checkbox"/> $\Theta(\log(b \cdot c))$ | <input type="checkbox"/> $\Theta(b \cdot c)$ | <input type="checkbox"/> $\Theta(c)$ |

Question 3 Soit la fonction : $f(n) = \frac{n \cdot (n + \log n)}{10}$. Est-ce que cette fonction est dans :

- | | | |
|----------------------------------------|-------------------------------------------------|---------------------------------------------------|
| <input type="checkbox"/> $\Theta(n)$ | <input type="checkbox"/> $\Theta(\frac{1}{10})$ | <input type="checkbox"/> $\Theta(n^n)$ |
| <input type="checkbox"/> $\Theta(2^n)$ | <input type="checkbox"/> $\Theta(n^2)$ | <input type="checkbox"/> $\Theta(n \cdot \log n)$ |



Question 4 ♣ Pour une table de hash (c'est-à-dire un tableau de seaux *tab*), lesquelles des affirmations suivantes sont vraies :

- ☐ Une table de hash permet d'ordonner des données dans l'ordre croissant en temps $O(n)$.
- ☐ Pour deux objets o et p , une fonction de hash valide doit avoir la propriété que $h(o) = h(p) \implies o = p$.
- ☐ Pour mon application, j'ai besoin de pouvoir lister des objets dans l'ordre dans lequel je les ai insérés. Je peux utiliser une simple table de hash.
- ☐ Aucune de ces réponses n'est correcte.

Question 5 ♣ Pour un tableau auto-redimensionnable, lesquelles des affirmations suivantes sont vraies :

- ☐ On peut retrouver le i^{e} plus grand élément du tableau (et donc le plus petit et le plus grand) en $O(1)$.
- ☐ On peut savoir si un élément est présent dans le tableau en $O(\log n)$.
- ☐ Je peux obtenir une copie du tableau à laquelle on aura ajouté un élément à la fin en temps $O(1)$.
- ☐ Ajouter un élément au début du tableau n'en placez se fait en temps linéaire.
- ☐ Aucune de ces réponses n'est correcte.

Question 6 ♣ Pour une liste simplement chaînée immuable de longueur n , lesquelles des affirmations suivantes sont vraies :

- ☐ Je peux concaténer une liste de longueur m devant une liste de longueur n en temps $O(m)$.
- ☐ On peut obtenir une mise à jour de la liste, dans laquelle on aura remplacé la valeur placée à la i^{e} position de la liste par une autre en temps $O(i)$.
- ☐ Je peux trier la liste en temps $O(n \cdot \log n)$.
- ☐ On peut obtenir une sous-liste comportant les m premiers éléments de la liste d'origine (opération **take(m)**) en temps $O(m)$.
- ☐ Aucune de ces réponses n'est correcte.