

TD 1 - Complexité, Listes

Conseil : conservez précieusement la correction des TDs, ils seront utiles en TPs.

Exercice 1. Complexité

```
entrées: tableau d'entiers tab et sa longueur n
s:=0;
pour i allant de 1 à n faire
    pour j allant de 1 à tab[i] faire
        s:=i*j;
```

Quelle est la complexité de cet algorithme...

1. ... si tous les entiers contenus dans `tab` sont plus petits que 10 ?
2. ... si tous les entiers contenus dans `tab` sont plus petits que n ?
3. ... si la valeur maximale m des entiers contenus dans `tab` est un paramètre ?
4. ... si n est fixé et m est un paramètre ?
5. ... si tous les entiers contenus dans `tab` sont plus petits que n et la multiplication de deux entiers a et b coûte $O(M)$, avec $M = \max(a, b)$?

Exercice 2. Premières fonctions sur les listes

Écrire les fonctions suivantes, et donner leur complexité. On n'utilisera pas de parcours itératif (uniquement récursif).

1. `Element thirdElement(Liste l)`; Renvoie le 3^e élément de l .
2. `Element ithElement(int i, Liste l)`; Renvoie le i^e élément de l .
3. `int contains(Element e, Liste l)`; Teste si la liste l contient l'élément e .
4. `Liste copie(Liste l)`; Renvoie une copie de l (une liste avec un contenu identique mais stockée à un emplacement mémoire différent).
5. `Liste deleteAll(Element e, Liste l)`; Renvoie une copie de l où les éléments égaux à e sont retirés.
6. `Liste mirror(Liste l)`; Renvoie une copie de l en ordre inversé.
Indice : Utiliser une fonction annexe qui reçoit deux listes A et B , et qui renvoie une liste composée du miroir de A suivi de B .

Exercice 3. Listes d'entiers

On suppose dans cet exercice que `Element` est le type `int`. Écrire les fonctions suivantes, et donner leur complexité.

1. `Element maxList(Liste l)`; Renvoie le maximum de l .
2. `int isIncreasing(Liste l)`; Teste si l est croissante.
3. `Liste makeRange(int a, int b)`; Renvoie la liste des éléments de l'intervalle $[a..b]$.

Exercice 4. Fonctions de Pile (LIFO)

Écrire les fonctions suivantes, et donner leur complexité.

1. `int empty(Liste p)` teste si la liste est vide

2. `void push(Element e, Liste* p )` ajoute `e` en tête de `p`
3. `Element pull(Liste* p)` retire et renvoie le premier élément de `p`. Quitte avec un message d'erreur si la liste est vide.

Quelle serait la complexité d'une fonction qui insère un élément en dernière position ?

Exercice 5. Fonctions de File (FIFO)

Écrire les fonctions suivantes à partir du type `ListeDC`, et donner leur complexité.

1. `int empty(ListeDC f)` teste si la liste doublement chaînée est vide
2. `void append(Element e, ListeDC* f)` ajoute `e` à la fin de `f`
3. `Element pull(ListeDC* f)` retire et renvoie le premier élément de `f`