
Quick 1 – 3 octobre 2025 – Durée 1 h

Une feuille A4 manuscrite recto-verso autorisée, ainsi que les dictionnaires bilingues.
 Les dispositifs électroniques sont interdits, à l'exception des montres non connectées.
 Le barème est indicatif.

Vous êtes vivement encouragés à illustrer vos recherches et vos réponses par des schémas.

1 Structure de données (7 points)

On s'intéresse dans cet exercice à la structure de données *Bloc* décrite par le type abstrait ci-dessous.

Initialiser(B)

{ **Donnée-résultat** : un Bloc B
Post-condition : B est initialisé avec un contenu vide
Effet de bord : B est modifié }

EstVide(B)

{ **Donnée** : un Bloc B
Résultat : un booléen vrai ssi B est vide }

Déposer(B,x)

{ **Donnée-résultat** : un Bloc B
Donnée : un élément x
Post-condition : x est ajouté dans B
Effets de bord : B est modifié }

Extraire(B)

{ **Donnée-résultat** : un Bloc B
Résultat : l'élément x le plus récent de B
Pré-condition : B est non vide
Post-condition : supprime l'élément x le plus récent de B
Effets de bord : B est modifié }

Effacer(B,x)

{ **Donnée-résultat** : un Bloc B
Donnée : un élément x
Post-condition : supprime l'élément de B le plus récent dont la valeur est égale à x
Effets de bord : B est modifié, sauf si aucun élément de B n'est égal à x }

- (1 pt) a) Illustrez le fonctionnement de cette structure en choisissant une suite d'opérations qui montre tout ce qui peut se passer, et en décrivant le contenu du *Bloc* par un schéma après chaque étape.
- b) Proposez une implémentation de ce type, autrement dit :
- (2 pts) (i) Décrivez une représentation concrète des objets de type *Bloc*, à l'aide d'autres types de données classiques (entiers, booléens, tableaux, listes chaînées).
 Vous accompagnerez vos explications de schémas.
Note : Si vous utilisez un tableau, vous ne tiendrez pas compte des cas où on essaye de stocker plus d'éléments que le tableau ne le permet.
- (3 pts) (ii) Écrivez un ensemble de fonctions qui réalisent les opérations spécifiées ci-dessus.
- (1 pt) (iii) Donnez (**sans justification**) le coût de chacune de vos opérations.

2 Écriture d'algorithme (6 points)

L'objectif de cet exercice est l'écriture et l'analyse d'algorithmes permettant de déterminer l'élément apparaissant le plus de fois dans un tableau donné.

Par exemple, dans $T = [E, B, E, E, I, C, B, C]$ l'élément E apparaît 3 fois alors que les autres n'apparaissent que 1 ou 2 fois, c'est donc E la valeur cherchée.

En cas d'égalité, parmi les « premiers *ex æquo* », on choisira de renvoyer celui qui apparaît le plus à gauche dans T . Par exemple pour $[R, E, B, E, I, C, B, C]$, les valeurs B, C et E apparaissent 2 fois chacune, on renvoie E car parmi les plus fréquents c'est lui qui est situé le plus à gauche dans T .

a) Algorithme naïf

(2 pts) (i) Écrivez un algorithme naïf qui résout ce problème.

(1 pt) (ii) Déterminez et justifiez le coût au pire de votre algorithme.

b) On admet pour cette partie disposer d'un algorithme qui trie un tableau de taille n en effectuant au pire $n \log n$ comparaisons.

On propose alors l'algorithme suivant :

```

MAJORITAIRE_AVEC_TRI( $T$ )
  Trier( $T$ )
   $max := 0$ 
   $i := 0$ 
  tant que  $i < n$  faire
     $candidat := T[i]$ 
     $occurrences := 0$ 
    tant que  $i < n$  et  $T[i] = candidat$  faire
       $occurrences := occurrences + 1$ 
       $i := i + 1$ 
    si  $occurrences > max$  alors
       $max := occurrences$ 
       $valeur := candidat$ 
  renvoyer  $valeur$ 

```

(2 pts) (i) Quel est le coût total de cet algorithme ?

Attention à bien détailler vos calculs et vos justifications.

(1 pt) (ii) Dans certains cas, cet algorithme ne répond pas au problème posé.

Lesquels et pourquoi ?

3 Analyse de coût (7 points)

Les deux algorithmes étudiés dans cet exercice prennent chacun en entrée un tableau T de n entiers dont les valeurs sont toutes comprises entre 0 et $n - 1$.

Précisons qu'il est possible que deux cellules distinctes de T possèdent des valeurs identiques.

Dans tout l'exercice, les mesures de coûts sont le nombre exact d'appels à la fonction **Traiter**, et **uniquement** ces appels.

a) On s'intéresse d'abord à l'algorithme :

```
ALGO_EGAL(T)
pour  $i := 0$  à  $n-1$  faire
  si  $T[i]=i$  alors
    Traiter T[i]
```

(1 pt) (i) Quel est son coût au pire ?
Justifiez et exhibez un cas défavorable.

(1 pt) (ii) Quel est son coût au mieux ?
Justifiez et exhibez un cas favorable.

(1 $\frac{1}{2}$ pts) (iii) Quel est son coût en moyenne ?
On suppose que chaque cellule de T contient un entier choisi au hasard uniformément entre 0 et $n - 1$. La valeur de chaque cellule est indépendante des autres.

b) Répondez ensuite aux mêmes questions pour ce second algorithme :

```
ALGO_INFERIEUR(T)
pour  $i := 0$  à  $n-1$  faire
  si  $T[i] \leq i$  alors
    Traiter T[i]
```

(1 pt) (i) Quel est son coût au pire ?
Justifiez et exhibez un cas défavorable.

(1 pt) (ii) Quel est son coût au mieux ?
Justifiez et exhibez un cas favorable.

(1 $\frac{1}{2}$ pts) (iii) Quel est son coût en moyenne ?
On suppose que chaque cellule de T contient un entier choisi au hasard uniformément entre 0 et $n - 1$. La valeur de chaque cellule est indépendante des autres.