Programmation 2 | **Time: 30 minutes** | Spring 2026

Quiz 3

Numéro d'étudiant : **NOM :** **Prénom :**

Téléphones, écouteurs et montres connectées obligatoirement dans le sac. Tout appareil électronique trouvé sur vous entraîne un 0 immédiat. Vous devez tout écrire sur le sujet.

1. Écrivez une fonction récursive find qui reçoit une liste `A` en paramètre et retourne un `set` contenant tous les éléments qui apparaissent *plus* de `len(A)/3.0` fois dans `A`. Si aucun élément ne satisfait cette condition, la fonction retourne un ensemble vide.

   Votre algorithme doit avoir complexité $O(n \log_2 n)$.

   Contrainte : Vous n'avez pas le droit d'utiliser d'autres fonctions que `len`, `set`, `find` et `add`.

   Donnez une justification de pourquoi votre algorithme calcule la réponse correcte.

   Exemples :

   `find( [1, 2, 3, 1, 2, 3, 1] )` $\longrightarrow$ retourne `{1}` car 1 apparaît 3 fois sur 7 ($3 > 7/3.0 \approx 2.33$).

   `find( [1, 2, 3, 4, 5] )` $\longrightarrow$ retourne `{}` car aucun élément n'apparaît plus de $5/3.0 \approx 1.66$ fois.

   `find( [1, 1, 1, 2, 2, 2, 3] )` $\longrightarrow$ retourne `{1, 2}` car 1 apparaît 3 fois sur 7 ($3 > 2.33$) et 2 apparaît 3 fois sur 7 ($3 > 2.33$).

   (10 points)

   Solution. C'est presque le même algorithme que pour l'élément majoritaire.

   Si un élément apparaît plus d'1/3 du temps dans `A`, alors il apparaît plus d'1/3 du temps dans au moins une des deux moitiés (relativement à la taille de cette moitié).

   Ainsi, il sera retourné par l'un des deux appels récursifs; ensuite, nous pourrons vérifier, avec une boucle, s'il apparaît effectivement plus d'un tiers du temps dans `A`.

   Pour la complexité : chaque appel récursif retourne au plus 2 éléments, donc la boucle de vérification parcourt au plus $4n$ fois par niveau. Avec $\log_2 n$ niveaux, on obtient $O(n \log_2 n)$.

```python
def find(A):
    n = len(A)

    if n == 0: return set()
    if n == 1: return { A[0] }

    B = find(A[:n//2]) | find(A[n//2:])
    C = set()
    for x in B:
        count = 0
        for i in range(0, n):
            if A[i] == x:
                count += 1
        if count > n/3.0:
            C.add(x)
    return C

print( find( [1, 2, 3, 1, 2, 3, 1] ) )
print( find( [1, 2, 3, 4, 5] ) )
print( find( [1, 1, 1, 2, 2, 2, 3] ) )
```