Correction

NSI - 2021 Amérique (21-NSIJ1AN1)

Exercice 4 - Arbre de compétition

1. a. La racine est "Lea".
Les feuilles sont "Marc", "Lea", "Claire", "Theo", "Marie", "Louis", "Anne" et "Kevin".

1. b.

```
def vainqueur(arb) :
    """ Renvoie le nom du vainqueur du tournoi. """
    return racine(arb)
```

1. c.

```
def finale(arb) :
    """ Renvoie les noms des deux compétiteurs finalistes."""
    return [racine(gauche(arb)), racine(droite(arb))]
```

2. a. Il faut écrire une fonction récursive :

```
def occurrences(arb, nom, total=0) :
    """ Donne le nombre d'occurrences du joueur. """
    if est_vide(arb):
        return 0
    else:
        if racine(arb) == nom :
            total = 1
        else :
            total = 0
        return total + occurrences(gauche(arb), nom, total) + occurrences(droit(arb), nom, total)
```

2. b.

```
def a_gagne(arb, nom):
    """ détermine si le joueur a gagné au moins un match dans la compétition """
    return occurrences(arb, nom) > 1
```

3. a. Le vainqueur de la compétition voit son nom apparaître dans la racine de l'arbre. Pour le vainqueur de la compétition, la valeur sera erronée car elle comptera une fois de trop son nom.

3. b. Code corrigé :

```
def nombre_matchs(arb, nom):
    """ arbre_competition, str -> int """
    total = occurrences(arb, nom)
    if racine(arb) == nom:
        return total - 1
    else:
        return total
```

Correction

NSI - 2021 Amérique (21-NSIJ1AN1)

4. Code complété :

```
def liste_joueurs(arb):
    """ arbre_competition -> tableau """
    if est_vide(arb):
        return []
    elif est_vide(gauche(arb)) and est_vide(droit(arb)) :
        return [racine(arb)]
    else:
        return liste_joueurs(gauche(arb)) + liste_joueurs(droit(arb))
```