

## Objectifs

À la fin de cette séance, vous devriez être capable de :

- utiliser des algorithmes de parcours d’arbre binaire pour résoudre des problèmes simples ;
- déterminer le coût d’un algorithme de parcours d’arbre.

## Exercice 1 : Échauffement

- Écrivez une fonction qui calcule le nombre de feuilles d’un arbre binaire.
- Écrivez une fonction qui renvoie l’arbre « miroir » d’un arbre binaire (dans lequel les fils gauche et droit de chaque nœud ont été inversés).

Pour écrire ces fonctions, vous avez besoin de manipuler un type abstrait `arbre`. Vous utiliserez les primitives d’accès définies en cours.

## Exercice 2 : Feuille haute

Pour un arbre binaire  $A$ , écrivez une fonction qui calcule la hauteur de la feuille la plus haute (la feuille la plus proche de la racine)

1. par un parcours en profondeur d’abord récursif,
2. par un parcours en largeur d’abord.
3. Justifiez pourquoi ces deux algorithmes (récursif et itératif) renvoient bien la valeur demandée.

## Exercice 3 : Feuilles à la profondeur $p$

Pour un arbre binaire  $A$  et une profondeur  $p$ , écrivez une fonction qui calcule le nombre de feuilles à la profondeur  $p$

1. par un parcours en profondeur d’abord récursif,
2. par un parcours en largeur d’abord.

Pour chacun de ces deux algorithmes (récursif et itératif), que pouvez vous dire de son coût ? Y-a-t-il un algorithme moins coûteux que l’autre ?

## Membres du groupe

- 
- 
- 
- 

## Feuille haute

Pour un arbre binaire  $A$ , écrivez une fonction qui calcule la hauteur de la feuille la plus haute (la feuille la plus proche de la racine)

— par un parcours en profondeur d’abord récursif,

— par un parcours en largeur d’abord.

Justifiez pourquoi ces deux algorithmes (récursif et itératif) renvoient bien la valeur demandée.