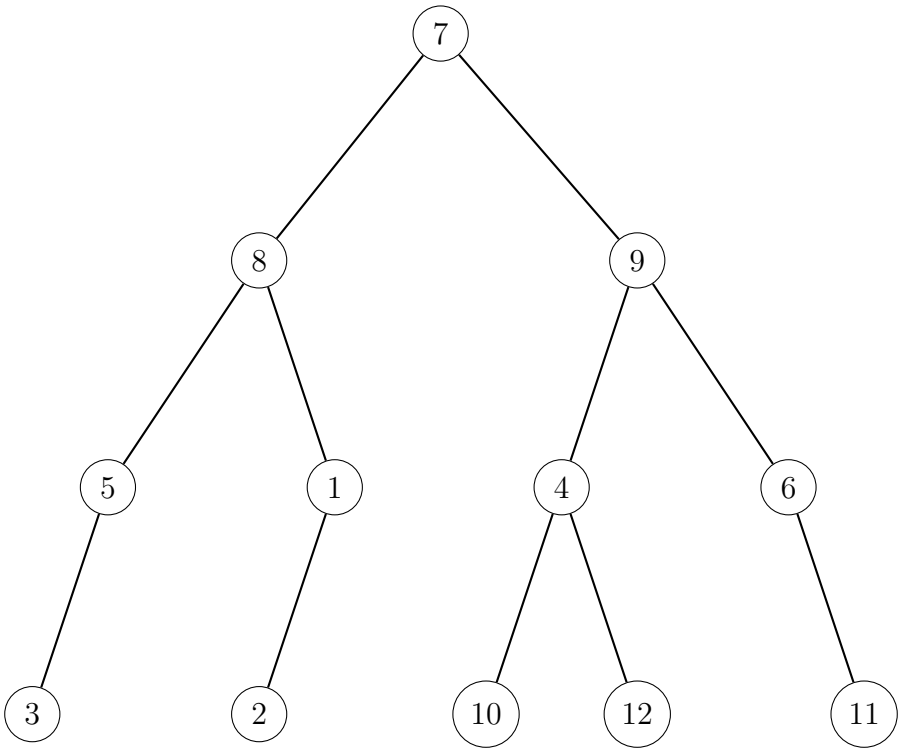
fiche 4

Arbres binaires

Nous reprendrons la construction de l'arbre binaire vu en cours.
Nous pourrons prendre en exemple l'arbre binaire suivant :



Exercice 1 : Définir l'arbre **exemple_1**.

Exercice 2 :

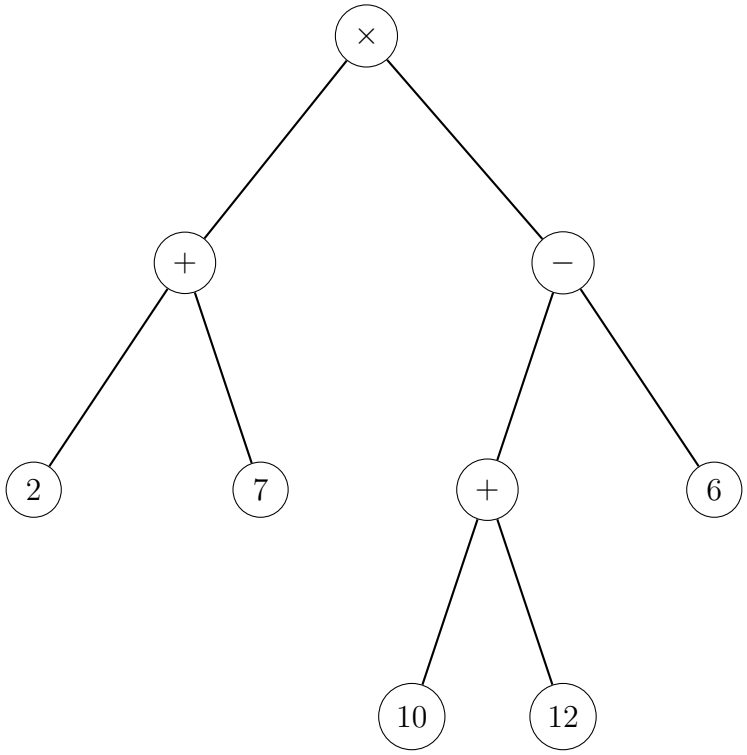
1. Écrire la fonction `recherche` qui prend en argument un arbre et un élément. La fonction renvoie le booléen correspondant à l'appartenance de l'élément dans l'arbre.
2. Écrire la fonction `mini_maxi` qui prend en argument un arbre et renvoie le minimum et le maximum élément de l'arbre.

Exercice 3 : On considère un liste. Le but est de construire simplement un arbre dont les étiquettes sont les éléments de la liste.
L'algorithme récursif est le suivant :

- Si la liste est vide, la fonction renvoie l'arbre vide.
- Sinon, la fonction divise la liste en deux.
  - La racine de l'arbre devient le centre de la liste.
  - Le sous arbre gauche est construit par la première partie de la liste.
  - Le sous arbre droit est construit par la deuxième partie de la liste.

Écrire la fonction `arbre_of_liste` qui convertit une liste en un arbre binaire.

Exercice 4 : On considère un arbre arithmétique suivant :



1. Calculer la valeur du résultat de l'exemple au dessus.
2. Déterminer les fonctions `fois`, `plus`, `moins` et `divise` qui prennent en arguments deux entiers, et renvoie le résultat du calcul.
3. Implémenter l'arbre exemple, à l'aide des fonctions précédentes.
4. Écrire la fonction `est_une_feuille`, qui prend en argument un arbre, et renvoie le booléen correspondant à l'état de feuille de l'arbre.
5. Écrire la fonction `calcul`, qui prend en argument un arbre arithmétique et renvoie le résultat du calcul de l'arbre.