# Travaux Dirigés d'algorithmique n°8

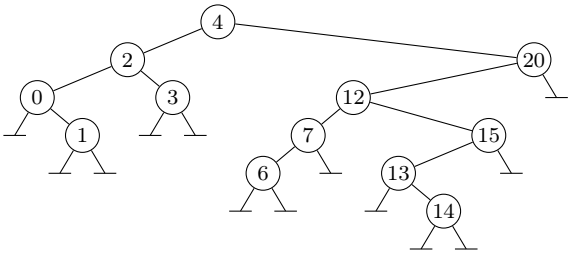
**Cours d'algorithmique**

*—Licence CFA / Troisième Année—*

---

▸ **Exercice 1. (Rotations dans les arbres binaires)**

On considère l'arbre binaire de recherche $T$ suivant :

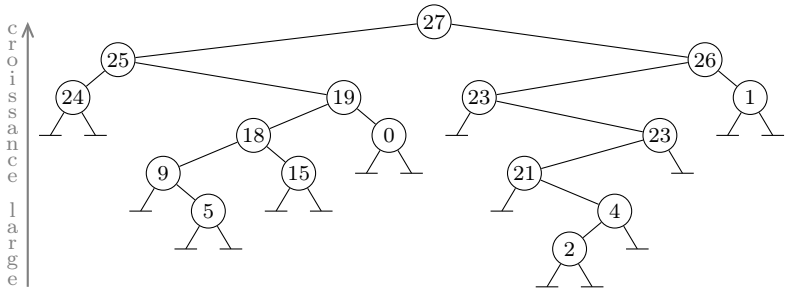



1. Vérifier que $T$ est bien un arbre binaire de recherche.
2. Effectuer une rotation de l'arête entre les noeuds étiquetés 2 et 4, même question pour 7 et 12.
3. Calculer pour chaque noeud quelle est la valeur de déséquilibre. L'arbre $T$ est-il équilibré ?
4. Vérifier que l'on peut équilibrer tous les arbres à trois noeuds en moins de deux rotations.
5. En effectuant le minimum de rotations, équilibrer l'arbre $T$.

▸ **Exercice 2. (Arbres tournois)**

On dit qu'un arbre binaire valué est *tournoi* s'il est vide ou si
- les deux sous-arbres gauche et droit sont eux-mêmes tournois ;
- l'étiquette de la racine est supérieure ou égale aux étiquettes des sous-arbres.

Voici un exemple de tournoi :




1. Écrire un algorithme qui retourne le plus grand élément d'un arbre tournoi non vide. Quelle est la complexité de cet algorithme ?

On veut un algorithme qui supprime la plus grande étiquette d'un arbre tournoi $a$ tout en maintenant la propriété de tournoi. Pour ceci, si les deux sous-arbres sont non vides, on peut la remplacer par la plus grande étiquette des racines des deux sous-arbres ; cette plus grande étiquette doit alors être elle-même supprimée récursivement du sous-arbre.

2. Écrire un algorithme `suppr_min` qui supprime la plus grande étiquette d'un arbre tournoi $a$ tout en maintenant la propriété de tournoi. Quelle est la complexité de cet algorithme ?

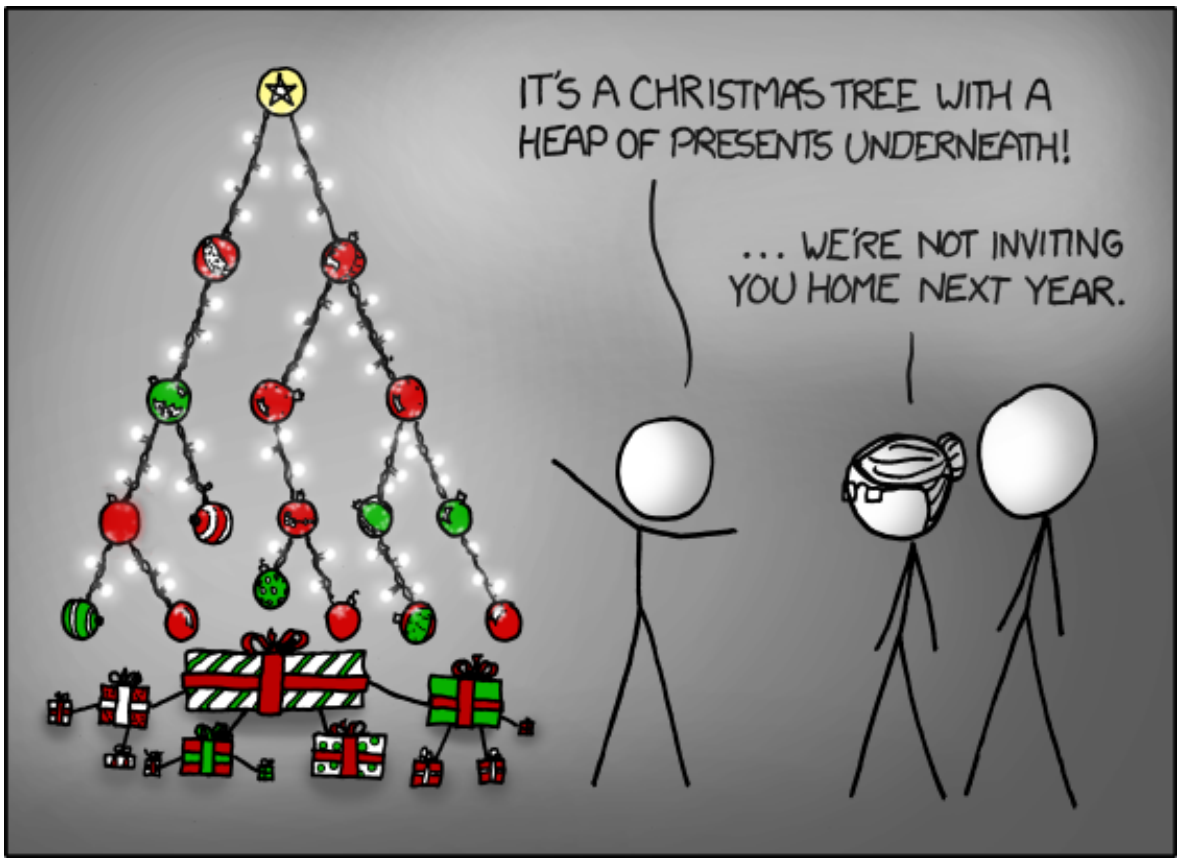
▸ **Exercice 3. (Tas)**

Un arbre binaire $T$ est dit *quasi parfait* s'il contient tous les noeuds dont les numéros sont entre 1 et $n$ où $n$ est la taille de $T$.

1. Quelle peut-être la forme d'un arbre quasi parfait à $n$ noeuds ?

On code un arbre quasi parfait par un tableau où le contenu du noeud de numéro $i$ est dans la case $i$ du tableau. Un *tas* est un arbre binaire qui est a la fois tournoi et quasi parfait.

1. Donner un algorithme qui insère un élément dans un tas. Le résultat doit être encore un tas. Quelle est la complexité ?
2. Donner un algorithme qui supprime le plus petit élément d'un tas. Le résultat doit être encore un tas. Quelle est la complexité ?
3. En déduire un algorithme de tri. Quelle est la complexité ?




C'est un arbre de Noël avec un tas de cadeaux en dessous !
... L'an prochain on t'invite pas chez nous.
Source : `http://xkcd.com/835/`