

Tri par tas

Motivation: Tri simple et optimal en temps.

Tas \rightarrow peut être représenté par un arbre binaire complet

Exemple:



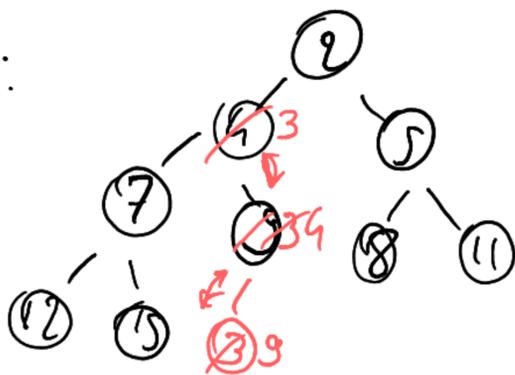
Remarque: peut être représenté par un tableau (voir indices)
hauteur logarithmique et le nombre de nœuds

Opérations

\hookrightarrow Création: renvoyer un tableau dynamique vide

\hookrightarrow Ajout: . Ajouter l'élément en fin de tableau $\rightarrow O(1)$ amorti
. Faire remonter l'élément tant que la propriété de tas n'est pas vérifiée.

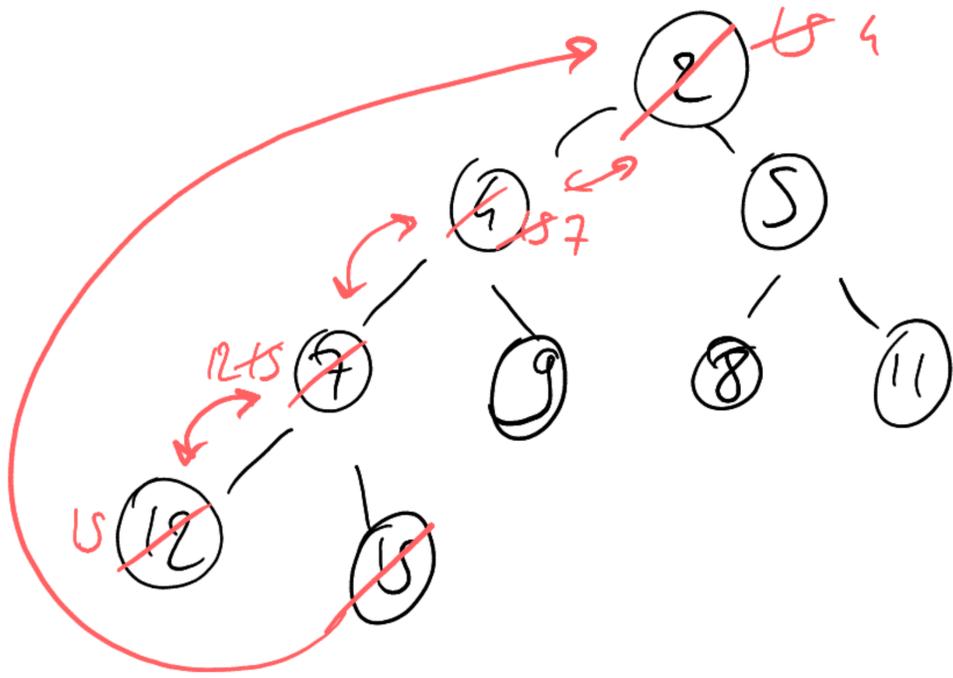
Exemple: ajout 3 :



Complexité: $O(\log n)$

↳ Suppression de la racine

↳ On remplace la racine par le dernier élément et on le fait descendre par échanges avec le fils le plus petit à chaque étape.



Complexité : $O(\log(n))$

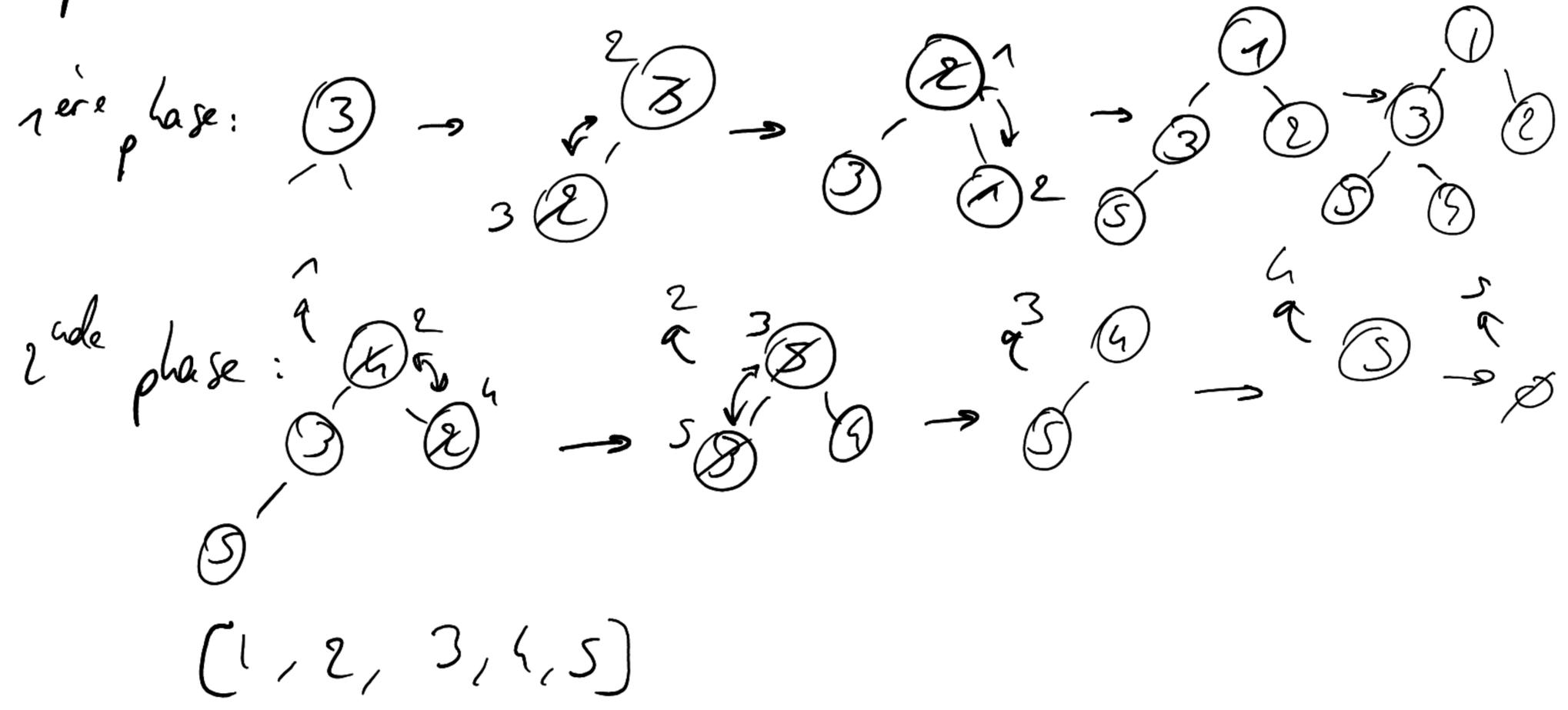
Tri par tas :

↳ On dispose d'un tableau avec tous les éléments.

↳ On les insère 1 à 1 dans un nouveau tas

↳ On les retire 1 à 1 dans le tableau initial

Exemple: Trier [3, 2, 1, 5, 4]



Terminaison: Immédiate

Correction: Immédiate car tous les éléments sont présents dans le tas et la suppression de la racine renvoie bien le plus petit élément

Complexité: $O(n \log n)$. → Optimal en temps

Possibilité de le faire en place ($O(1)$ en mémoire) avec un tas max:

