

# LOG3430 – Méthodes de test et de validation du logiciel

---

## Laboratoire 2

Tests de partition de catégorie et de flot de données

Département de génie informatique et de génie logiciel  
École Polytechnique de Montréal



Hiver 2020

## **1. Mise en contexte théorique**

Tel que présenté dans le cours, il existe plusieurs familles de critères d'adéquation pour réaliser des tests en boîte noire. Ces critères visent à réduire le nombre de valeurs à tester pour chaque variable, ainsi qu'à réduire le nombre de combinaisons de ces valeurs entre les différentes variables. Ce TP vise le critère partition de catégorie, puis revient aux critères de tests en boîte blanche basés sur le flot des données pour illustrer le focus différent des tests en boîte noire et blanche.

En particulier, pour le critère partition de catégorie, on regardera le critère AC (All Choice; chaque choix doit être utilisé avec tous les choix des autres catégories) et EC (Each Choice; chaque valeur de chaque choix pour chaque catégorie doit être utilisée au moins dans un cas de test). Comme critère de tests en boîte blanche, on utilisera all-P-uses/some-C-uses.

## **2. Objectifs**

- Mettre en pratique les connaissances théoriques acquises sur les tests fonctionnels (boîte noire);
- Comprendre les critères de couverture (flot de données);
- Appliquer les concepts de catégorie-partition;
- Être en mesure d'identifier des critères AC (All Choice) et EC (Each choice).

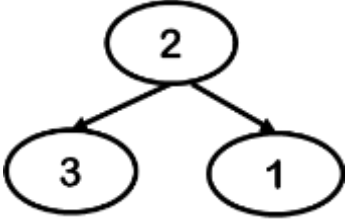
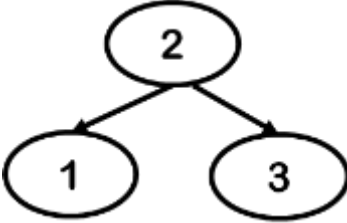
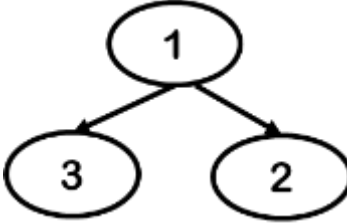
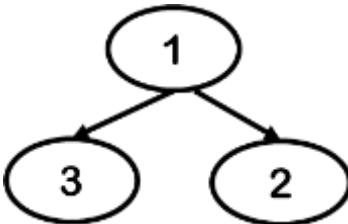
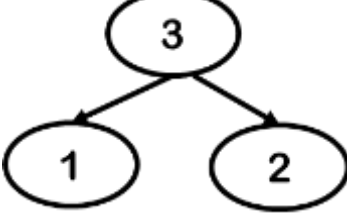
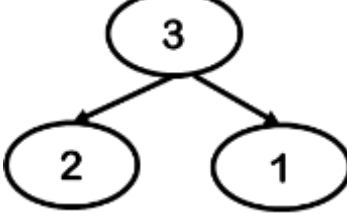
## **3. Mise en contexte pratique**

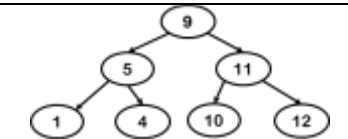
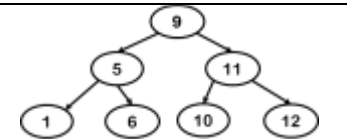
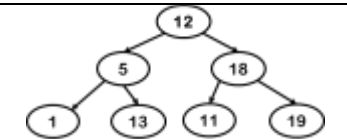
Il s'agit de tester fonctionnellement la structure de données BST (Binary Search Tree). Les spécifications de cet algorithme sont fournies dans le fichier PowerPoint ci-joint. Un code basic en python est aussi fourni pour vous permettre de tester certains cas.

Propriété du BST :

- ❖ Chaque nœud du sous-arbre droit doit être plus grand que le nœud actuel et chaque nœud du sous-arbre gauche doit être plus petit que le nœud actuel. Si cette propriété n'est pas vérifiée, il ne s'agit pas de BST.

Exemples :

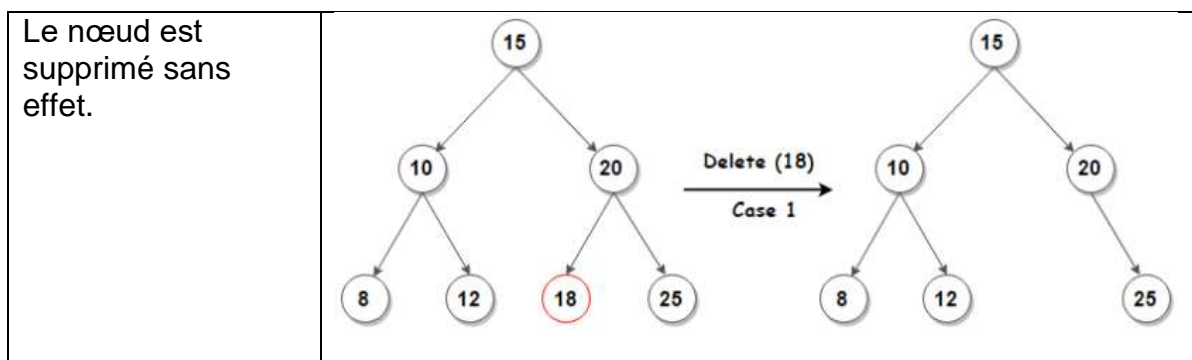
		
✗	✓	✗
		
✗	✗	✗

		
✗	✓	✗

### ❖ Suppression de nœuds

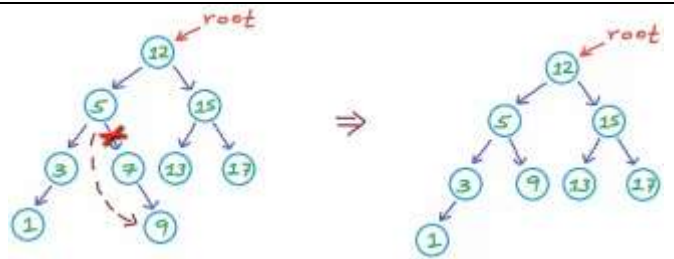
La suppression d'un nœud est l'opération la plus complexe. Afin de préserver la propriété du BST, il faut traiter les trois cas de figure :

- 1<sup>er</sup> cas : nœud feuille (pas d'enfants)



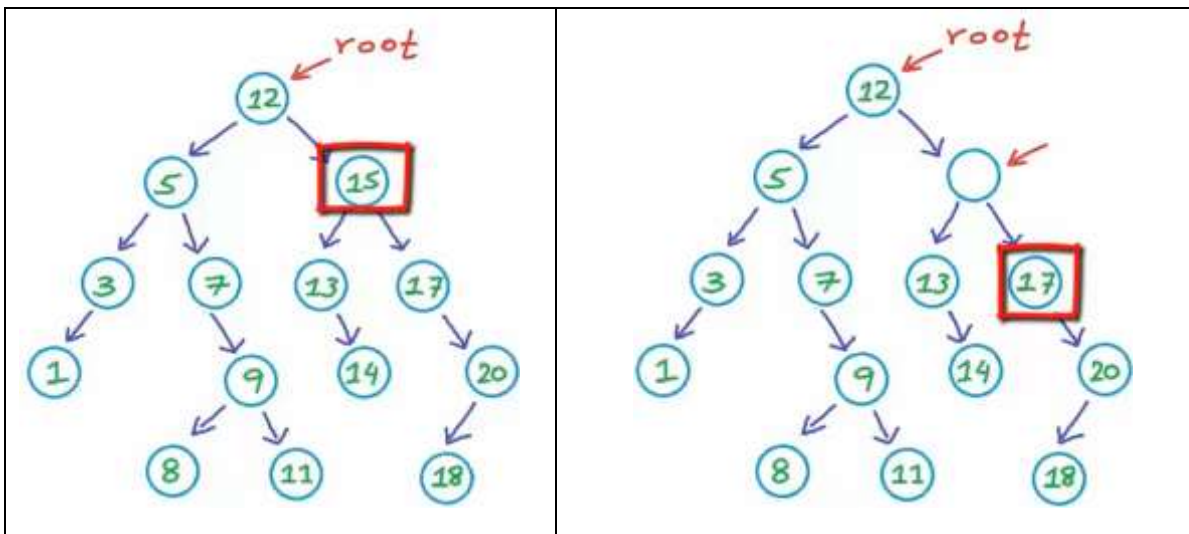
- 2e cas : un seul enfant

On connecte le père du nœud à son fils (même si le fils a des enfants).



- 3e cas : deux enfants

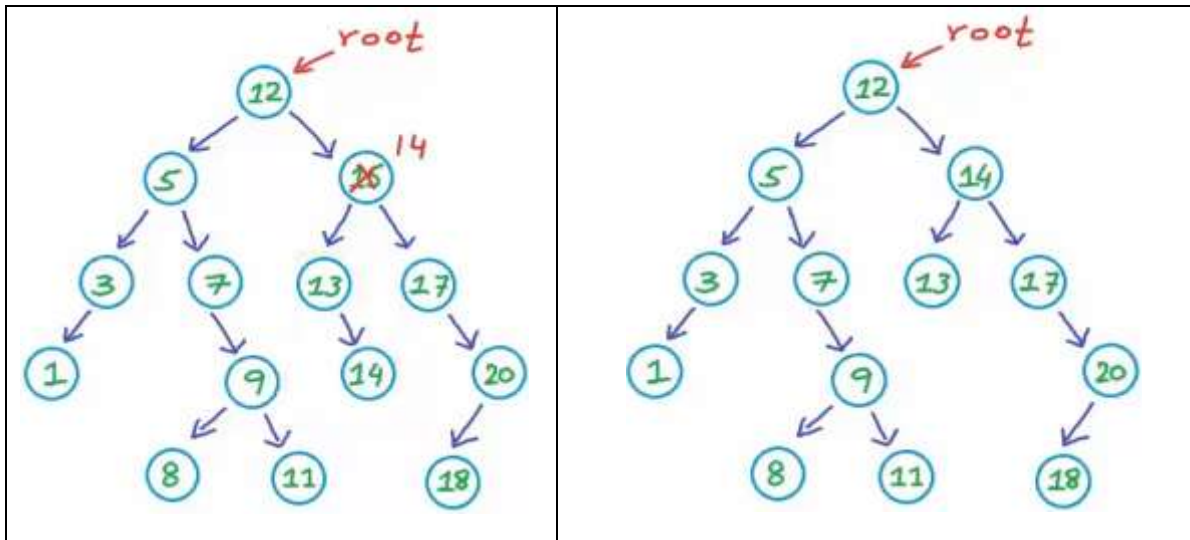
Deux cas de figure tels que présentés dans Tab1 et 2.



**Tab1.** On réduit ce cas au 1<sup>er</sup> ou 2<sup>e</sup> cas :

Trouver le **minimum** dans le sous-arbre **de droite**, ici **17**. Quand on remplace **15** par **17** la propriété demeure préservée. Notez que le nœud **17** a un seul enfant donc on sait comment le supprimer en utilisant le 2<sup>e</sup> cas.

Pourquoi le minimum dans le sous-arbre droit et pas une autre valeur ? On doit conserver la propriété que les nœuds à gauche doivent être plus petits et à droite plus grands.

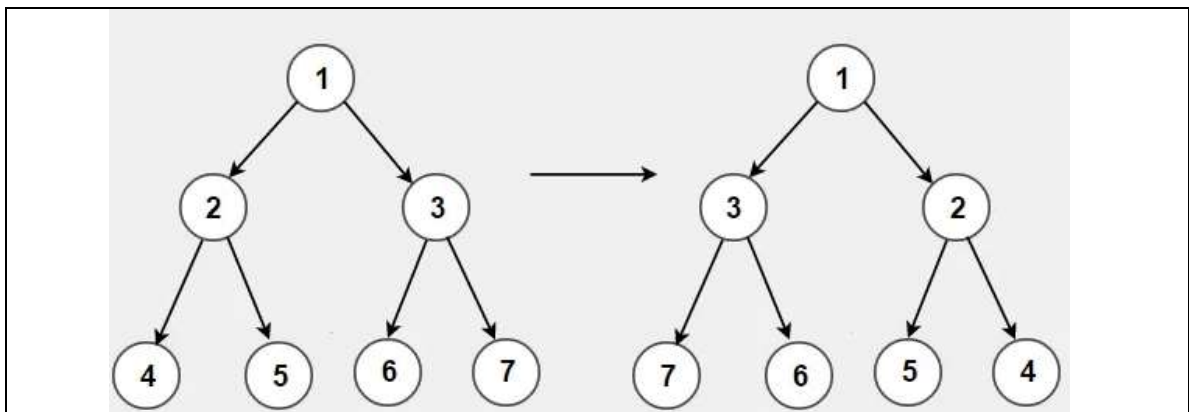


**Tab2.** On réduit ce cas au 1<sup>er</sup> ou 2<sup>e</sup> cas. Un autre choix serait :

Trouver le **maximum** dans le sous-arbre **de gauche**, ici **14**. Quand on remplace **15** par **14** la propriété demeure préservée. Notez que parce qu'on prend une valeur à gauche la valeur est toujours plus petite que le nœud. Le nœud **14** n'a pas d'enfants donc on sait comment le supprimer en utilisant le 1<sup>er</sup> cas.

#### ❖ L'inversion d'un arbre

Pour pouvoir traverser un arbre avec la même méthode *Printtree()* mais en ordre inverse, nous avons une méthode *inverttree()* dont le fonctionnement est illustré sur la figure suivante :



C'est l'une des questions d'entrevue les plus connues qui peut être facilement résolue récursivement. Nous traversons l'arbre de manière *preorder* et pour chaque nœud rencontré, nous échangeons son enfant gauche et droit avant d'inverser récursivement son sous-arbre gauche et son sous-arbre droit. On peut aussi parcourir l'arbre en mode *postorder*.

## 4. Travail à effectuer

- 4.1. Pour afficher la liste des nœuds en ordre (inorder traversal), on exploite la propriété de base d'un BST pour lire les nœuds de l'arbre de **gauche à droite**. Utilisez l'approche de partition de catégories EC pour tester les opérations d'insertion et d'affichage. Astuce : considérez la propriété « le BST est trié correctement » comme une des catégories. Dans votre rapport, il faut aller jusqu'au niveau des cas de test, **\*\*pas\*\*** l'écriture des tests avec l'échafaudage unittest. [4 points]
- 4.2. Utilisez de nouveau l'approche de partition de catégories, mais cette fois avec AC au lieu de EC, pour l'opération de suppression de nœuds (méthode **delete\_node()**) dans le fichier **BST.py**. Allez jusqu'au niveau de l'écriture des tests avec unittest. [6 points]
- 4.3. Utilisez maintenant l'approche boîte blanche all-P-uses/some-C-uses (flot de données) sur la même opération (méthode **delete\_node()**) dans le fichier **BST.py** que 4.2 (suppression de nœuds) en regardant le code source joint à cet énoncé. Allez de nouveau jusqu'au niveau de l'écriture des tests avec unittest. Comparez vos résultats avec ceux de 4.2, qu'est-ce que vous remarquez? [6 points]
- 4.4. Utilisez de nouveau l'approche boîte blanche all-P-uses/some-C-uses (flot de données), cette fois-ci sur la méthode **invertTree()** dans le fichier **BST.py** (inversion de l'arbre). Dans votre rapport, il faut aller jusqu'au niveau des cas de test, **\*\*pas\*\*** l'écriture des tests avec l'échafaudage unittest. [4 points]

## 5. Livrables attendus

Les livrables suivants sont attendus :

- Un rapport pour le laboratoire avec des réponses sur les questions.
- Le dossier COMPLET contenant le projet.

Le tout à remettre dans une seule archive zip avec pour nom **matricule1\_matricule2\_lab1.zip** à téléverser sur Moodle.

Le rapport doit contenir le titre et numéro du laboratoire, les noms et matricules des coéquipiers ainsi que le numéro du groupe.

Consultez le site Moodle du cours pour la date et l'heure limite de remise des fichiers. **Un retard de ]0,24h] sera pénalisé de 10%, de ]24h, 48h] de 20% et de plus de 48h de 50%.**