Rapport TP 4 d'informatique C++

Exercice 1 : Classe d'arbre (d'entiers) :

Exercice 2 : Affichage d'un arbre :

Ce parcours d'arbre correspond à un parcours en profondeur d'abord (depth first), ce qui s'implémente de manière récursive.

Exercice 3 : Gestion d'erreurs :

Liste de tous les cas d'erreurs pour les fonctions de IntTree :
Tout d'abord à chaque fois que l'on a un « return » dans une fonction, il faut vérifier que le type de la valeur retournée est le bon.
Plus précisément, fonction par fonction :
-Le constructeur : il s'agit de s'assurer que « d » est bien un entier.
-Le destructeur : il faut veiller à ce qu'il n'y ait pas de fuite mémoire, que l'on n'effectue pas qu'une destruction partielle de l'arbre.
-GetData : il faut s'assurer que lors de l'appel le noeud n'est pas vide.
-SetData : vérifier que la valeur d est bien un entier.
-getSon : il est possible qu'il n'y ait rien à l'endroit pos. Donc il convient de s'assurer que pos est compris entre 0 et nbSon.
-RemoveLastSon : il s'agit de s'assurer que les vecteurs « sons » ne sont pas vides afin de pouvoir « pop » un élément.
-Display : Cette fonction est récursive donc il faut vérifier qu'elle se termine, qu'elle ne tourne pas en boucles à l'infini. Il faut également s'assurer que la fonction « print » dans le bon ordre les éléments de l'arbre et avec les bonnes indentations pour la seconde fonction display.
Les deux sources d'erreurs principales sont donc les dépassements de taille et le fait que le vecteur sons soit vide ou non.

L'utilisation des méthodes '.at' à la place des crochets pour l'accès aux éléments d'un vecteur permet de signaler les erreurs de 'out of range' possiblement causées par 'pos'.

La méthode choisie pour gérer les erreurs est la méthode des 'try, catch et throw'.
Cette méthode permet de gérer les erreurs de manière indépendante vis-à-vis du reste du code, c'est pour cette raison que j'ai retenu cette technique.

Exercice 4 : La classe Template :

Le fichier Tree<T> ne peut pas être séparé en Tree.h et Tree.cpp pour la compilation séparée.
En effet, l'ordre de compilation rend impossible cette séparation.
En C++, les fichiers cpp sont d'abord compilés puis le main donc le template ne sera pas utilisé.
Enfin, la dernière raison est le fait que la séparation en plusieurs fichiers induirait une dépendance entre ces fichiers que C++ ne pourrait pas gérer.