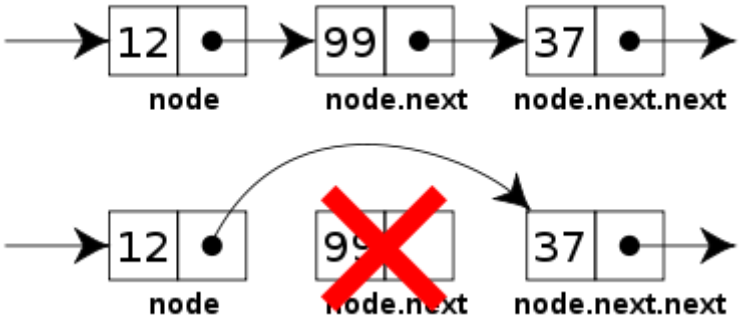
TP6: Structures avancées

Nizar Ouarti

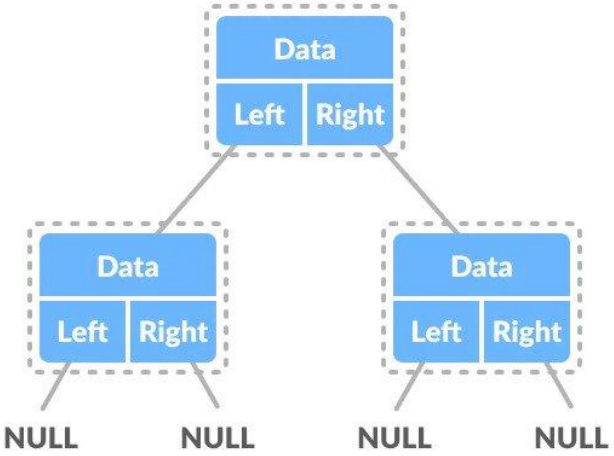
Listes chainées (linked lists)



Une liste chainée simple est une structure où chaques noeud est constituée d'une valeur et d'un pointeur vers l'élément suivant.

1. Créer une structure ListNode qui est un noeud composant une liste chaînée simple
2. Créer l'élément racine
3. Créer la fonction “insertEnd” qui ajoute un élément à la fin de la liste.
4. Créer la fonction “insertBeginning” qui insère au début
5. Créer la fonction “insertMiddle” qui insert un nouvel élément à l’intérieur de la liste, après un élément donné.
6. Créer la fonction “deleteNode” qui efface un noeud basé sur sa valeur.
7. Créer la fonction “printList” qui affiche la liste
8. Créer la fonction “searchNode” qui trouve si un noeud est présent en fonction de sa valeur
9. Bonus :Créer la fonction “sorterList” qui va trier les éléments de la liste en ordre croissant

Arbres Binaires de Recherche
(Binary Search Trees)



Un arbre binaire est un arbre ou chaque noeud contient une valeur et un enfant left et right. Dans un arbre binaire de recherche la valeur insérée doit être insérée à gauche si elle est inférieur au noeud courant (en partant de la racine) sinon elle est insérée à droite. Les feuiles sont les éléments terminaux de l'arbre initialisées à NULL.

1. Créer une structure TreeNode.
2. Créer une fonction newNode qui crée un nouveau noeud
3. Créer la fonction insert qui ajoute un élément dans l'arbre de manière ordonné (la valeur de gauche est inférieure à à la racine pour toutes les sous-parties). Si l'élément se trouve à une feuille alors on crée un node avec la valeur. Sinon si la valeur est inférieur à la valeur du noeud courant on va au noeud de gauche suivant, si c'est supérieur ou égale, au noeud de droite. Utiliser la récursivité.
4. Créer la fonction inorder qui va lire les valeurs en ordre croissant. Pour cela elle va partir du noeud le plus à gauche, puis on affiche la valeur et ensuite les noeud à droite de manière récursive. On s'arrête à une branche quand on atteint une feuille.
5. Trier les nombres [101, 8, 14 129, 4, 99,13, 55] grace à cet arbre.
6. Bonus: Écrire la fonction qui efface 55. Sachant qu'elle parcours l'arbre pour trouver la valeur, gauche ou droite de manière récurcive tant qu'elle n'a pas trouvé la valeur précise. Une fois la valeur trouvée
    a. Si le noeud n'à pas d'enfant, il suffit de l'effacer.
    b. S'il n'a qu'un enfant, son enfant va le remplacer.
    c. S'il à deux enfant, on trouve le noeud qui est le suivant dans l'ordre de l'arbre qui va le remplacer et être à son tour effacé de manière récurcive. Cet élément est le plus petit de la branche droite (part rapport à l'élément à effacer).