Analyse Asymptotique du Laboratoire 1

Structure de Données : BiblioTab (Array)

**Méthode ajouterOuvrage**

Instruction baromètre 🡪 

Petite explication du bloc d’instruction dans laquelle se trouve l’instruction baromètre :

Une clé auto-générée (this.compteur+1) est auto-incrémentée si elle ne passe pas le test

de la méthode contains du sorted set (treeset) ‘cles’.

Supposons que la clé est le dernier élément du treeset ‘cles’, ce dernier étant par nature un tableau d’entiers automatiquement trié en ordre croissant, la méthode contains aurait donc itérée d’un ordre de log2 de n si n est la taille du treeset donc **bigO = O(log n).**

**Méthode supprimerOuvrage()**

Instruction baromètre 🡪 

Supposons que l’ouvrage est le dernier élément de notre tableau (Array), la méthode rechercherOuvrage, faisant parti de notre code pour les méthodes supprimerOuvrage(), aurait donc itérée d’un ordre de n si n est la taille du tableau donc **bigO = O(n).**

**Méthode rechercherOuvrage**

Instruction baromètre 🡪 

Supposons que l’ouvrage est le dernier élément de notre tableau (Array), la méthode rechercherOuvrage aurait donc itérée d’un ordre de n si n est la taille du tableau donc **bigO = O(n).**

Structure de Données : BiblioList (LinkedList)

**Méthode ajouterOuvrage**

Instruction baromètre 🡪 

Supposons que la clé est le dernier élément du treeset ‘cles’, ce dernier étant par nature un tableau d’entiers automatiquement trié en ordre croissant, la méthode contains aurait donc itérée d’un ordre de log2 de n si n est la taille du treeset donc **bigO = O(log n).**

**!!!!!!! SANS CETTE CONTRAINTE !!!!!!!**

Instruction baromètre 🡪 ****

Insérer dans une LinkedList de Java place l’élément à la fin de la liste. Par conséquence, notre méthode aurait ici été effectué d’un ordre de temps constant peu importe la taille de notre LinkedList donc **bigO = O(1).**

**!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

**Méthode supprimerOuvrage()**

Instruction baromètre 🡪 

Supposons que l’ouvrage est le dernier élément de notre liste (LinkedList), la méthode rechercherOuvrage, faisant parti de notre code pour les méthodes supprimerOuvrage(), aurait donc itérée d’un ordre de n si n est la taille du tableau donc **bigO = O(n).**

**!!!!!!! SANS CETTE CONTRAINTE !!!!!!!**

Instruction baromètre 🡪 ****

Supposons que notre ouvrage à supprimer n’est ni la tête ni la queue de notre LinkedList, une boucle présente dans la méthode remove de LinkedList aurait été itérée d’un ordre de n – 2 si n est la taille de notre LinkedList donc **bigO= O(n – 2) = O(n**).

**!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

**Méthode rechercherOuvrage**

Instruction baromètre 🡪 

Supposons que l’ouvrage est le dernier élément de notre liste la méthode rechercherOuvrage aurait donc itérée d’un ordre de n si n est la taille du tableau donc **bigO = O(n).**

Structure de Données : BiblioListPerso(ListeChainee)

**Méthode ajouterOuvrage**

Instruction baromètre 🡪 

Supposons que la clé est le dernier élément du treeset ‘cles’, ce dernier étant par nature un tableau d’entiers automatiquement trié en ordre croissant, la méthode contains aurait donc itérée d’un ordre de log2 de n si n est la taille du treeset donc **bigO = O(log n).**

**!!!!!!! SANS CETTE CONTRAINTE !!!!!!!**

Instruction baromètre 🡪 **** (.ajouter)

Insérer dans une ListeChainee (personnelle) place l’élément à la fin de la liste. Par conséquence, notre méthode aurait ici été effectué d’un ordre de temps constant peu importe la taille de notre LinkedList donc **bigO = O(1).**

**!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

**Méthode supprimerOuvrage()**

Instruction baromètre 🡪 

Supposons que l’ouvrage est le dernier élément de notre liste (LinkedList), la méthode rechercherOuvrage, faisant parti de notre code pour les méthodes supprimerOuvrage(), aurait donc itérée d’un ordre de n si n est la taille du tableau donc **bigO = O(n).**

**!!!!!!! SANS CETTE CONTRAINTE !!!!!!!**

Instruction baromètre 🡪 **** (.retirer)

Supposons que notre ouvrage à supprimer n’est ni la tête ni la queue de notre LinkedList, une boucle présente dans la méthode remove de LinkedList aurait été itérée d’un ordre de n – 2 si n est la taille de notre LinkedList donc **bigO= O(n – 2) = O(n**).

**!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

**Méthode rechercherOuvrage**

Instruction baromètre 🡪 

Supposons que l’ouvrage est le dernier élément de notre liste la méthode rechercherOuvrage aurait donc itérée d’un ordre de n si n est la taille du tableau donc **bigO = O(n).**