Algorithmes et Structures de données | 2021

Journée 4 – Vendredi 12 mars

Exercice 1 – Implémentation de la recherche séquentielle en Python

Le but de cet exercice est d'implémenter et d'évaluer les performances de l'algorithme de recherche séquentielle en utilisant le langage Python. Pour ce faire, il s'agit de traduire en Python l'algorithme donné en pseudo-code dans les slides, en s'inspirant de la version donnée en Java. Pour l'évaluation de performances, inspirez-vous du code Java fourni en annexe.

Exercice 2 – Implémentation de la recherche binaire en Python

Le but de cet exercice est d'implémenter et d'évaluer les performances de l'algorithme de recherche binaire en utilisant le langage Python. Pour ce faire, il s'agit de traduire en Python l'algorithme donné en pseudo-code dans les slides, en s'inspirant de la version donnée en Java. Pour l'évaluation de performances, inspirez-vous du code Java fourni en annexe.

Exercice 3 – Propriétés des arbres binaires de recherche

Supposons que nous ayons des nombres entre 1 et 1000 dans un arbre binaire de recherche, et que nous voulions rechercher le nombre 363. Lesquelles des séquences suivantes **ne** pourraient **pas** être une séquence des nœuds examinés avant de trouver 363 ?

a. 2, 252, 401, 398, 330, 344, 397, 363.

b. 924, 220, 911, 244, 898, 258, 362, 363.

c. 925, 202, 911, 240, 912, 245, 363.

d. 2, 399, 387, 219, 266, 382, 381, 278, 363.

e. 935, 278, 347, 621, 299, 392, 358, 363.

Exercice 4 – Implémentation des arbres binaires de recherche en Python

Le but de cet exercice est d'implémenter la structure de données des arbres binaires de recherche en Python. Pour ce faire, il s'agit de créer une classe `Node` qui implémente la structure de données présentée dans les slides. Pour simplifier, on admet que la clé et la valeur sont le même champ.

A ce stade, on ne vous demande pas d'implémenter les algorithmes d'insertion et de suppression d'éléments dans l'arbre. En revanche, une fois que votre structure est définie, on vous demande d'effectuer les tâches suivantes :

a. Créer manuellement l'arbre binaire de recherche donné dans les slides en utilisant la classe `Node`.

b. Implémenter les algorithmes INORDER-TREE-WALK(x), TREE-MINIMUM(x) et TREE-MAXIMUM(x) en Python, en utilisant la classe `Node`.

Exercice 5 – Implémentation de l'insertion et la suppression en Python

Le but de cet exercice est d'implémenter en Python les algorithmes d'insertion et de suppression d'éléments dans l'arbre binaire de recherche, en utilisant la classe `Node`. Pour ce faire, il s'agit de s'inspirer des algorithmes correspondants donnés dans les slides.

Exercice 6 – Trois autres algorithmes sur les arbres binaires de recherche

Le but de cet exercice est d'écrire en pseudo-code trois algorithmes utilisant les arbres binaires de recherche, à savoir TREE-PRECESSOR(x), ainsi que des versions récursives de TREE-MINIMUM(x) et TREE-MAXIMUM(x). Pour ce faire, il s'agit de s'inspirer des algorithmes donnés dans les slides.

Exercice 7 – L'opération de suppression est-elle commutative ?

L'opération de suppression est-elle *commutative* dans le sens où la suppression de x puis de y dans un arbre de recherche binaire laisse le même arbre que la suppression de y puis de x ? Expliquer pourquoi c'est le cas ou donner un contre-exemple.