Manuela Dorado Novoa

ArrayList time: 616.8889999979291
Linkedlist time: 2.375999999999939
Binary search tree time: 157.96100000011276

ArrayList is faster than LinkedList if I randomly access its elements. I think random access means "give me the nth element".

Access time for ArrayList: O(1). Access time for LinkedList: O(n).In an array, you can access to any element by using `array[index]`, while in a linked list you must navigate through all the list starting from `first` until you get the element you need.

LinkedList is faster than ArrayList for deletion. I understand this one. ArrayList's slower since the internal backing-up array needs to be reallocated.

Deletion time for ArrayList: Access time + O(n). Deletion time for LinkedList: Access time + O(1).The ArrayList must move all the elements from array[index] to array[index-1] starting by the item to delete index. The LinkedList should navigate until that item and then erase that node by decoupling it from the list.

LinkedList is faster than ArrayList for deletion. I understand this one. ArrayList's slower since the internal backing-up array needs to be reallocated.

Insertion time for ArrayList: O(n). Insertion time for LinkedList: O(1).

Why the ArrayList can take O(n)? Because when you insert a new element and the array is full, you need to create a new array with more size (you can calculate the new size with a formula like 2 * size or 3 * size / 2). The LinkedList just add a new node next to the last.