

15-351 / 15-650 / 02-613: Exercises 2a

1. (a) Given a sorted doubly linked list L of n numbers S , give an $O(1)$ method to extract a valid binary search tree T from the linked list. *Hint: in a doubly linked list, each object has pointers to the left and to the right. What pointers do objects in a binary search tree have? You may reinterpret the meaning of existing pointers.*
(b) Given a sorted doubly linked list L of n numbers S , give an $O(n)$ algorithm for constructing a balanced binary search tree S . You may assume for simplicity that the number of items is exactly one less than a power of 2.
2. The skip list introduced in lectures supports several expected $O(\log n)$ -time operations, such as **find**, **insert**, and **delete**. Please modify the skip list to support another operation **index** which takes an integer number k as input and outputs the k -th smallest element in the skip list. The expected runtime should also be $O(\log n)$ (so it doesn't depend on k). Show how to handle this query, and how to modify other operations if necessary.
Hint: Try to store an extra piece of information at every node and update it when necessary.
3. An array is a contiguous reserved block of memory used to store a list of numbers. Consider a dynamic array A of size n with reserved space for $m \geq n$ items. Then, indexing into it to find $A[i]$ is $O(1)$ time, whereas inserting items is $O(n)$. However, pushing an item onto the end of the array is $O(1)$ time if we keep track of the current size n in a counter $C = n$. Then, pushing an item x simply requires setting $A[C] = x$ (we are 0-indexing here, so $A[n]$ is the $(n+1)$ th item) and then $C = C + 1$, but this only works if $m \geq n + 1$. If $m < n + 1$, then we need to reallocate a larger block of memory and copy all the items over to the new memory space, which may take $O(n)$ time. Design and analyze a version of a dynamic array where pushing an item onto the end of the array takes amortized $O(1)$ time.
4. Suppose you inserted the integers 3, 11, 7, 20, 9 into a splay tree in that order. Draw the structure of the splay tree following the last insertion.
5. Given a binary search tree of size n , we can recover a sorted list of all keys by doing an in-order traversal of the nodes. Design a sorting algorithm using splay trees and the above property. What is the runtime of the sorting algorithm?

Some sorting algorithms (like naive quicksort) perform poorly on already sorted data. Prove that the splay tree sorting algorithm actually performs better on an already sorted list than it does on an unsorted list. What is the runtime?

For the purposes of this problem, please be careful to distinguish between amortized analysis, average-case performance, and worst-case performance.