COMP 285 (NC A&T, Spr '22) Weekly Quiz 4

1

Assume you have a function called 3Sort that takes as input an array of size 3 of comparable objects and returns them in sorted order. You want to sort an array of size $n \geq 3$ objects only using callsto 3Sort. How many function calls of 3Sort are required to sort an array of size n? Choose the asymptotically largest applicable lower bound.

Solution

$\Omega(n \log n)$

It's just similar with sorting the array directly (Using a Comparison Sort with Merge Sort). The only difference is that we compare 3 elements each time instead of 2.

2

I claim to have a data structures that can store comparable objects and supports the following operations:

1. insert new objects in $O(1)$ time per insertion
2. remove objects in $O(1)$ time per removal
3. return the smallest object in the data structures in $O(1)$ time.

Is it possible to have such a data structure? (Hint: Recall that sorting n comparable elements must take at least $O(n \log n)$ time. Can you sort faster if the data structure I described above exists?)

Solution

Impossible

It is possible to insert/remove objects in average O(1) time, however, it's impossible to return the smallest object in O(1) time. If we could find the smallest element is O(1) time, then the whole array could be sort in O(n) time.
Recall that sorting n comparable elements must take at least $O(n \log n)$.

3

Assume you have an array A of size n with positive integer element with all elements in the range [1, n^3). What is the runtime of Radix Sort using base 10 run on A?

Hint: The runtime of radix sort is $O(d(n+r))$, where n is the size of your input array, r is the number of buckets, and d is the number of digits.

Solution

$\Theta(nlogn)$

$d = \lfloor log_10(n) \rfloor + 1$, so time $= O(nd) = O(nlog(n))$.

4

The runtime of Radix Sort is $O(d(n+r))$ where d is the number of digits in our base, n is the number of elements we're sorting, and r is the number of buckets we're using to sort (our base). When sorting integers, what is a "good" choice of r?

Solution

n

Compared with 2, 10, n^2 this gives the best running time.

5

Which of the following describes the height of a red-black tree on n nodes?

Solution

- $\Theta(\log n)$ because a red-black tree is balanced, which means its height is $\Theta(\log n)$.
- $O(\log n)$ because if the height is $\Theta(\log n)$ then it is both an upper and lower bound.
- $\Omega(\log n)$ because if the height is $\Theta(\log n)$ then it is both an upper and lower bound.

6

If the length of a path from the root of a red-black tree to one of the leaf NIL nodes is 100, what could be the length of another path from the root to some other NIL node?

Solution

180

Compared with 30 and 45, both of which are far too small. Even if they consist of all black-nodes, the longest other path could be at most 60 and 90 (respectively). Red-Black Trees always have height at most $2 * \log(n+1)$, since path can be at most twice as long another if we pad it with red nodes.

7

What is the worst-case runtime of operations INSERT/DELETE/SEARCH on a red-black tree storing n nodes?

Solution

$$\Theta(\log n)$$

As with general Binary Search Trees, all operations are O(height). So all operations with RBTrees are $O(\log(n))$.

8

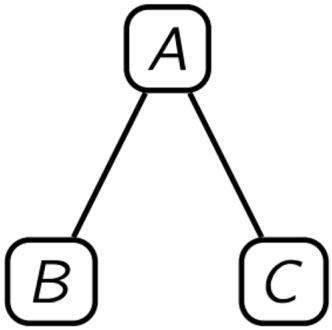
Hash tables with universal hash families guarantee an expected runtime of $O(1)$ for the INSERT,SEARCH, and DELETE operations.

Solution

True. This is the definition of a hash table.

9

Suppose that the nodes A,B,C in a binary search tree are arranged as follows. (The tree has no duplicate nodes.). What must be true?



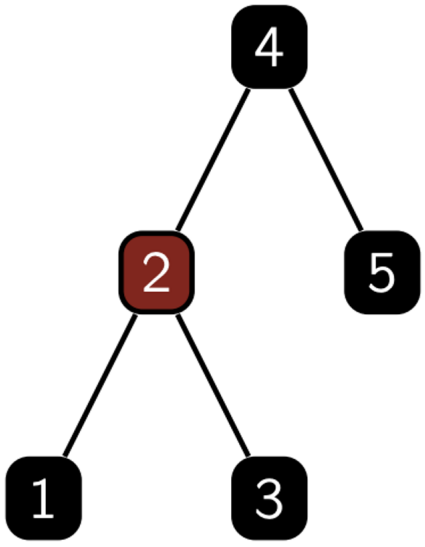
Solution

$$B < A < C$$

By definition of the binary search tree.

10

Is the following a valid red-black tree?



Solution

Yes

By definition of the red-black tree.