Chimzim Ogbondah

Com S 311

9 March 2021

Exam 1

1. K
   1. = O(n)

f(n) ≤ c \* g(n), f(n) = 16n

≤

= c \* g(n)

Let c = 16 N = 1 for all n ≥ N

f(n) ≤ c \* g(n), f(n) =

16n

n ≤ c \*16

n > c \* 16 since c \*16 will be a constant value eventually n will be greater than it hence **n ≠**

1. Alg1(A)
   1. n = array.length c1 = 1
   2. constant number of operations c2 = 1
   3. for(i=n; i ≥ 1; i = i/2) c3 = log(n)
   4. for(j=1; j ≤ n; j = j+1) c4 = n

c1 + c2+ (c3log(n) \* c4(n) )

c1 + c2 + c5nlog(n) where c5 = c3\*c4

**O(nlog(n))**

1. findIndex(int[] A, int index)

if(A.length == 1)

if(index == A[1])

found = true //Note found would be a boolean variable used to dictate in main if an index is found

return index //prints out the index

else()

return null //prints out null

bool found = false

int a =

int b = a +1

left = findIndex(A[1->a], a)

right = findIndex(A[b->n], b)

if(!found)

return null //prints out null or that nothing is found

/\* where A[] is the given array and n is equal to array.length \*/

T(n) = 2T(n/2) + 1

O(log(n))

1. T(n) = 2T(n/4) + 16 ; T(1) = O(1)

Using case 2 of the Master Theorem

If f(n) =

16 + 1 = Ɵ( /\*square both sides\*/

256n +1 = Ɵ(n)

T(n) = Ɵ(

1. Since the parent of in a heap is equal to taking the last leaf **(N)** of the heap knowing that it can be stored inside of an array its parent would be . This makes it tells us that its parent cannot be . Because of this statement the leaf nodes will be indexed in the array from to n. by subtracting these values you can find the max number of leaves which would . <= (n-