Jerome Schmidt 70497605 HW#2 Written

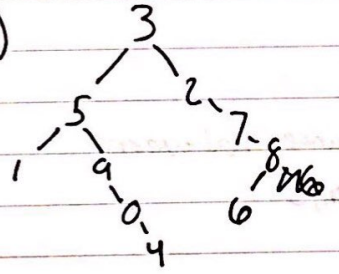
① The algorithm needs to return a one instead of a zero

Input: Binary search tree T

Output: Number of leaves in T

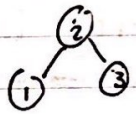
```
if T = Θ
    return 0
else if T_L = Θ and T_R = Θ
    return 1
else
    return LeafCounter(T_L) + LeafCounter(T_R)
```

② a.)



b.) n=2 and n=3

↳ for (0, 1, 2) and (1, 0, 2)



no tree exists that satisfies this

c.) Input: Array $I[i_0, i_1 \dots i_{n-1}]$ for inorder list

Array $P[p_0, p_1 \dots p_{n-1}]$ for postorder list

Output: -1 if no binary tree that satisfies conditions is found or prints elements if tree can be constructed

```
BinaryTree(I[i_0 ... i_{n-1}], P[p_0 ... p_{n-1}])
    flag = -1
    for i = 0 to n-1
        if (I[i] == P[n-1])
            // prints root
            Output I[i]
            flag = 1
        else
            flag = -1
    if (flag == -1)
        return flag
    // recursively call function
    BinaryTree(I[i+1, ... n-1]; P[k, ... n-2])
```


Scanned with CamScanner

③ if $n = 10^5$

mergesort = $n \log n$

binary search = $\log n$

$n \log n + k \log n \leq k \frac{n}{2}$

$k \geq \frac{n \log n}{\frac{n}{2} - \log n}$

plug in $n = 10^5$

$k_{min} = 23$

if $n = 10^8$

$k_{min} = \frac{n \log n}{\frac{n}{2} - \log n}$

plug in $n = 10^8$

$k_{min} = 36$

④ a.) Input: Array A[ ] of numbers

Output: Array D[ ] that contains the distances between two consecutive elements

```
mergesort(A[])
for 0 → i, i<n-1
    D[i] = (A[i+1] - A[i])
return D[]
```

mergesort running time: $O(n \log n)$ / Difference running time: $O(1)$

so running time is $O(n \log n)$

b) Brute force running time: $O(n^2)$

since $O(n \log n)$ running time is much less than $O(n^2)$ the presorted algorithm performed significantly better

Scanned with CamScanner

⑤ Input: Array of numbers $A[n_1 \dots n_k]$

output: Array of sorted numbers $A[n_i \dots n_{i+k}]$

int flag = 0

~~while(flag = 0)~~

place element i in box i

While(flag == 0)

~~if all inequalities are satisfied~~ for(int i = 0 ~~for~~ ~~if~~ while i < k ; i++)

~~flag = 1~~

if(i and i+1 don't satisfy inequality between them)
swap i and i+1

end for loop ~~else~~

if(all inequalities are satisfied

flag = 1

return A[ ]

⑥ multiplications → $\sum_{i=0}^{n} \sum_{j=1}^{i+i} 1 \Rightarrow \frac{(n+1)(n+2)}{2} \Rightarrow \Theta(n^2)$

additions → $\sum_{i=0}^{n} 1 = n+1$

Scanned with CamScanner