Assignment 8.

Haume Kizza.

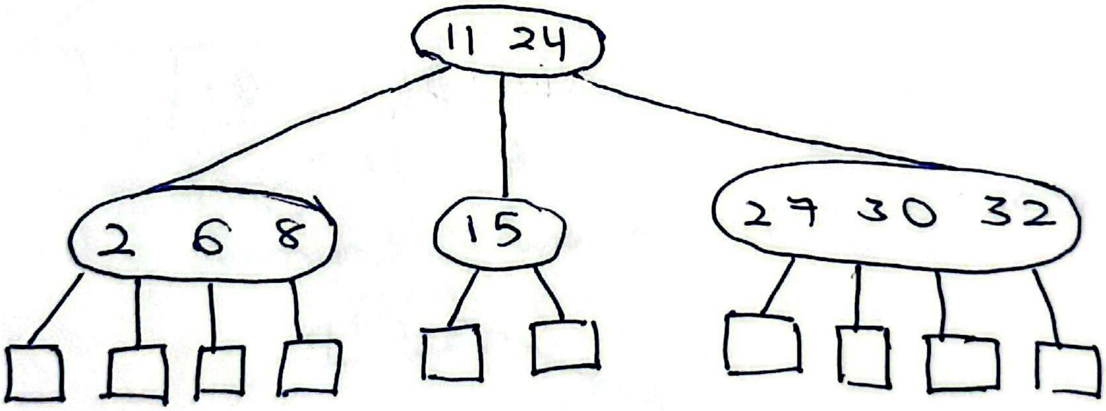
1

The multi-way search tree for lecture 8 slide 39 isn't a valid (2,4) tree because.

① its leaves / external nodes are not at the same depth.

② [27, 32] is not fully shown if it an external node or an internal node. if its an external node then the children are not properly positioned.

To make it into a valid tree we could move 30 up to the middle of the 27 and 32 nodes so the tree is balanced.

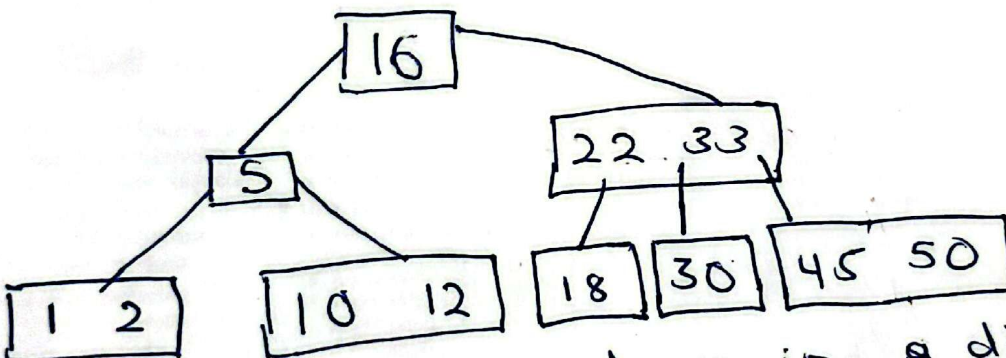


R.3.10 n

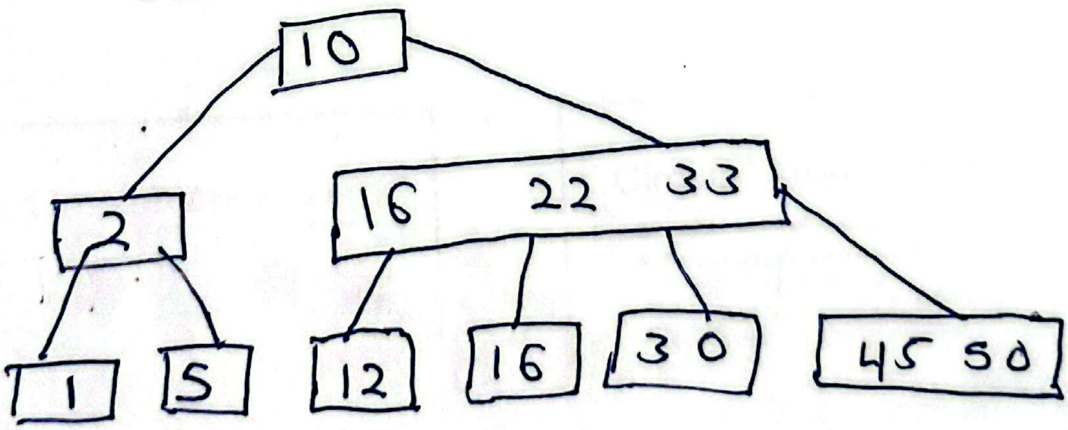
17 (2,4) tree does not always have the same structure when inserting the same items in different orders. The timings of splits and promotions depend on insertion order, which leads to different tree shapes. therefore, the claim is false.

inserting the items in this order will

(16, 5, 22, 45, 2, 10, 18, 30, 50, 12, 1, 33)



while inserting the items in a different order
ie sorted elements would be: (1, 2, 5, 10, 12, 16, 18, 22, 30, 33, 45, 50



Assignment 8

C-4.11 Election Winner in O(n log k) Time

```
Algorithm FindWinner(S, n, k):
    votes = empty balanced BST

    for v in S:
        if votes.contains(v):
            votes[v] = votes[v] + 1
        else:
            votes.insert(v, 1)

    winner = null
    maxVotes = 0
    for candidate, count in votes:
        if count > maxVotes:
            maxVotes = count
            winner = candidate

    return winner
```

Time Complexity

Updating counts for n votes: $O(n \log k)$ (since each insert/update takes $O(\log k)$ and we do it n times).

Finding the maximum: $O(k)$ (negligible compared to $n \log k$ when $n >> k$).
Total = $O(n \log k)$.

C-4.22 Pair Sum Check in O(n log n) Time

```
Algorithm PairSum(A, B, n, x):
    sort(B)  // O(n log n)

    for a in A:  // O(n log n)
        target = x - a
        if binarySearch(B, target) == true:
            return true

    return false
```

Time Complexity

Sorting B: $O(n \log n)$.

n binary searches: $n \times O(\log n) = O(n \log n)$.
Total = $O(n \log n)$.