On my honor, I pledge that I have neither recieved nor provided any improper assistance in the completion of this assignment.

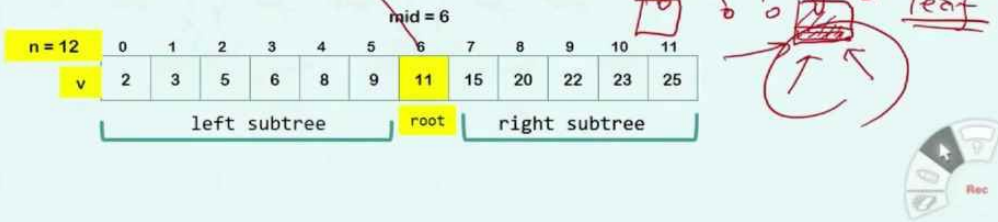
—2190012 김석원

## Homework

- When you rebuild an AVL tree from BST using recycling, how the left and right of the leaf nodes are to set to **`nullptr`**. Explain how they are being set in your code.
- You may add new pages of ppt to answer the question.

### Building AVL tree from BST in O(n) – recycling method

```
// rebuilds an AVL tree using a list of nodes sorted, no memory allocations
// v - an array of nodes sorted, n - the array size
tree buildAVL(tree* v, int n) {
  if (n <= 0) return nullptr;
  int mid = n / 2;

  tree root = v[mid];                    // mid becomes the root; don't call new TreeNode.

  // recursive buildAVL() calls for left & right, return it to root->left & root->right



  return root;
}
```



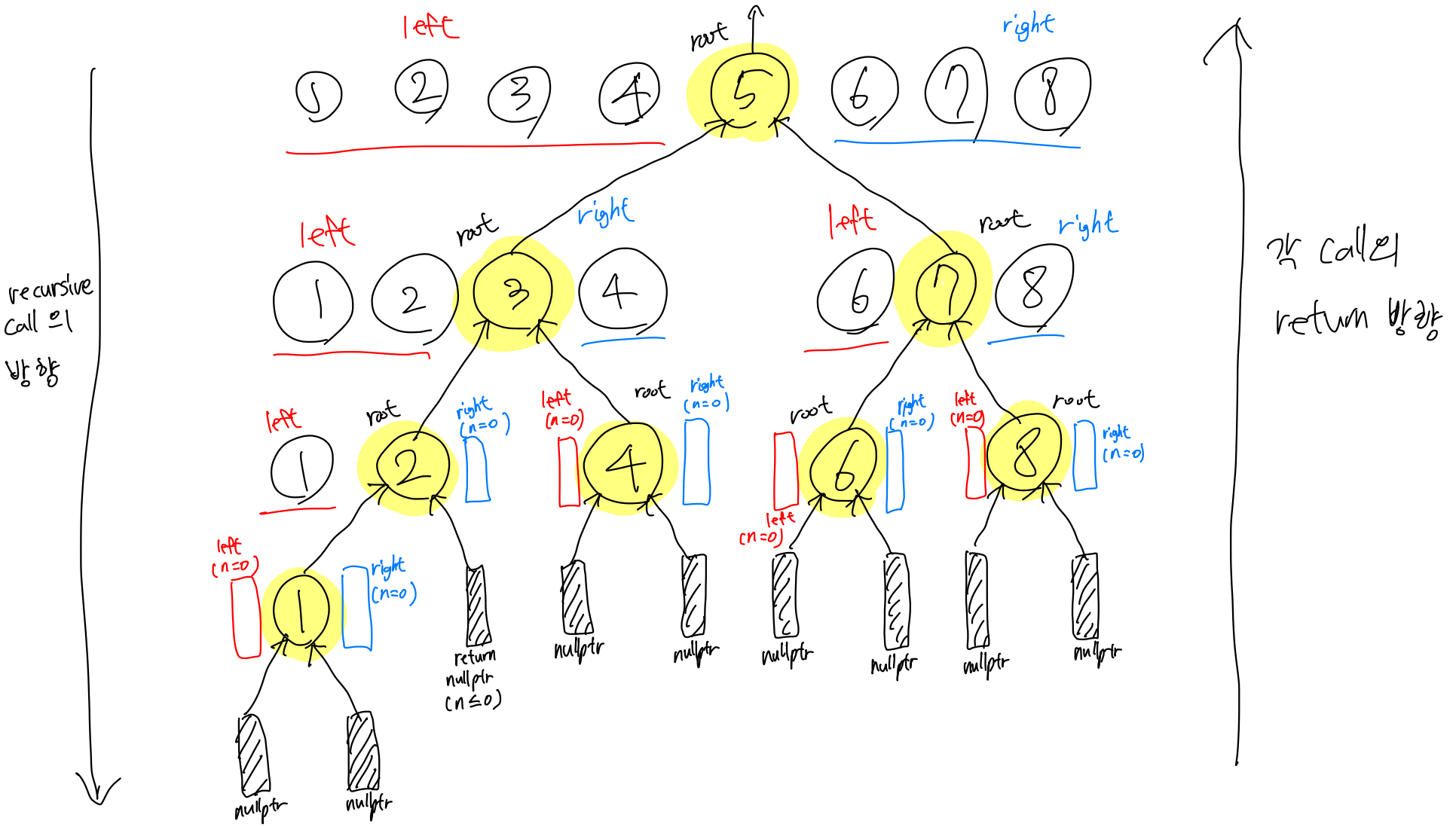
```
tree buildAVL(tree* v, int n) {  // recycling me
    if (n <= 0) return nullptr;
    DPRINT(cout << ">buildAVL v[0]=" << v[0] <<
    int mid=n/2;

    tree root=v[mid];

    root->left=buildAVL(v,mid);
    root->right=buildAVL(&v[mid+1],n-mid-1);

    DPRINT(cout << "<buildAVL" << n << endl;);
    return root;
}
```

윈쪽의 코드가 직접 구현한 코드이다. recycle 하기 위해 key값이 아니라 node 자체로 구성된 vector를 .data() method를 사용하여 생성된 array를 reconstruct 로부터 받게 되고, 이를 parameter로 받아야 한다. array의 중간 노드를 root로 정하고, root의 left와 right는 root를 제외한 배열의 나머지 왼쪽과 오른쪽 부분을 알맞는 index를 지정해주고, 이를 두번째 parameter로 사용하여 각각 recursive call 하게 된다.

여기서 base case를 살펴본다면, n≤0, 즉 array의 size가 0보다 작거나 같으면 nullptr을 return 한다. 배열의 길이가 0인 경우는 int n 값이 0인 경우, 즉 이전 recursive call 단계에서 n=1 인 경우이다. 이 경우, n이 1이기에 mid는 0이 되고, 따라서 root→left에서 n 값은 0, root→right에서 n 값은 n-mid-1, 즉 0이 된다.

n=0 이 parameter로 전달된다면 base case에 의해 각각 nullptr이 리턴되고, system stack에 저장되어 있던 이전 노드들에게 차례로 return 되면서 tree가 완성되게 된다.

ex)




이런 식으로, 모든 leaf node의 left와 right가 nullptr로 초기화된다.