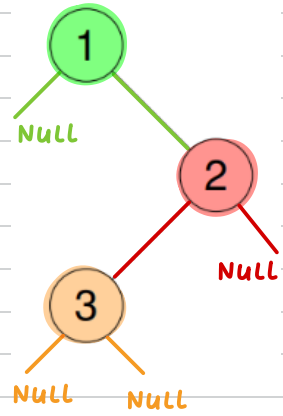
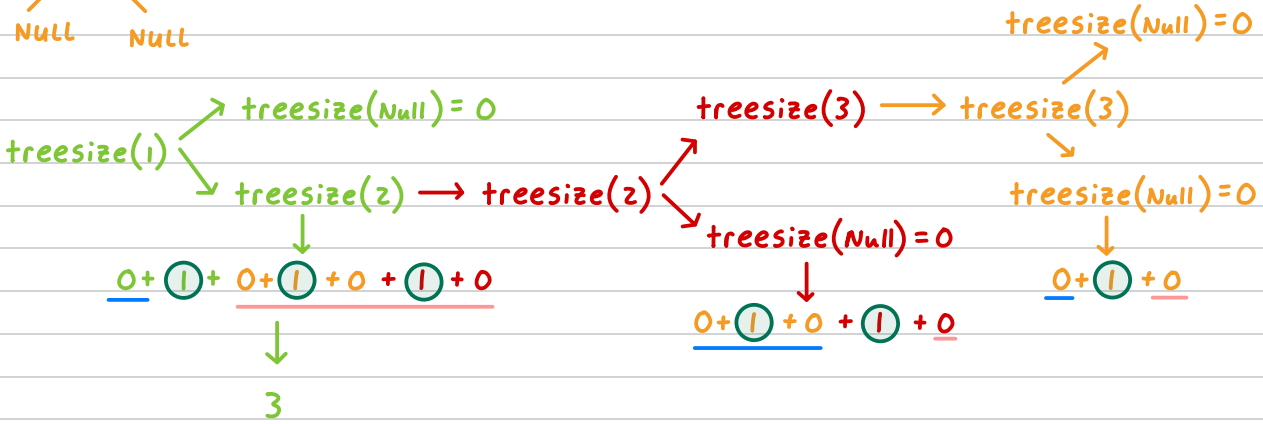
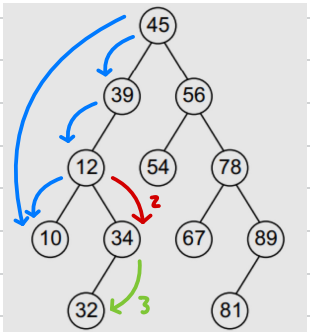
```c
int treeSize(struct TreeNode* root) {
    if (root == NULL) {
        return 0;
    }
    return treeSize(root->left) + 1 + treeSize(root->right);
}
```





```c
void preOrder (struct Node *tree) {
    if (tree != NULL) {
        printf("%i ", tree->data);
        preOrder(tree->left);
        preOrder(tree->right);
    }
}
```



---

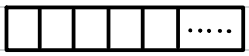
print ⟶ array[]

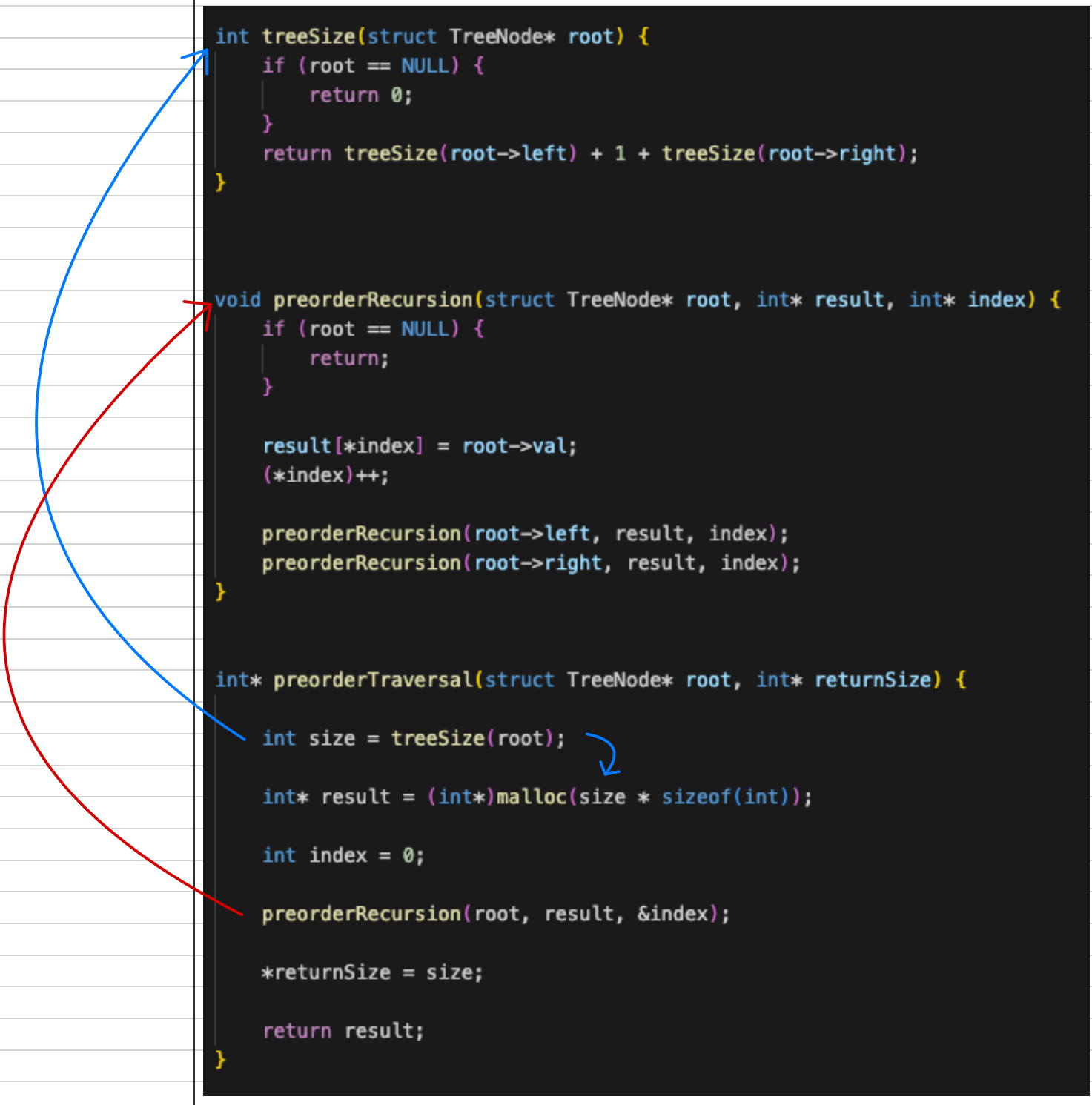
```c
void preOrder (struct Node *tree) {
    if (tree != NULL) {
        printf("%i ", tree->data);
        preOrder(tree->left);
        preOrder(tree->right);
    }
}
```

print → array[]

```c
void preorderRecursion(struct TreeNode* root, int* result, int* index) {
    if (root == NULL) {
        return;
    }

    result[*index] = root->val;
    (*index)++;

    preorderRecursion(root->left, result, index);
    preorderRecursion(root->right, result, index);
}
```

```c
int* preorderTraversal(struct TreeNode* root, int* returnSize) {

    int size = treeSize(root);

    int* result = (int*)malloc(size * sizeof(int));

    int index = 0;

    preorderRecursion(root, result, &index);

    *returnSize = size;

    return result;
}
```

result →  size = # nodes ↔ tree size



```c
int treeSize(struct TreeNode* root) {
    if (root == NULL) {
        return 0;
    }
    return treeSize(root->left) + 1 + treeSize(root->right);
}



void preorderRecursion(struct TreeNode* root, int* result, int* index) {
    if (root == NULL) {
        return;
    }

    result[*index] = root->val;
    (*index)++;

    preorderRecursion(root->left, result, index);
    preorderRecursion(root->right, result, index);
}


int* preorderTraversal(struct TreeNode* root, int* returnSize) {

    int size = treeSize(root);

    int* result = (int*)malloc(size * sizeof(int));

    int index = 0;

    preorderRecursion(root, result, &index);

    *returnSize = size;

    return result;
}
```