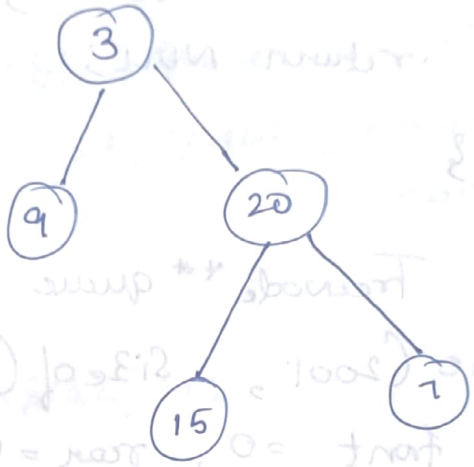
09/May/24

lab - 2

Binary Tree level order Traversal.
of its node's values.

Example 1:



Input : root = [3, 9, 20, null, null, 15, 7]

output: [[3], [9, 20], [15, 7]]

Example 2:

Input : root = [1]

output : [[1]]

Example 3:

Input : root = [ ]

Output : [ ]

```
int** levelOrder(struct TreeNode* root,
    int* returnSize, int** returnSize
    int** returnColumnSize){
    if(!root){
    *returnSize = 0;
    return NULL;
    }
struct TreeNode** queue = (struct TreeNode**)
calloc(2001, sizeof(struct TreeNode*));
int front =0, rear = 0;
queue[rear++] = root;
int** result = (int**) calloc(2001,
        sizeof(int*));
*returnColumnSize = (int*) calloc(2001,
        sizeof(int));
*returnSize =0;
while(front< rear){
    int levelSize = rear - front;
    (*returnColumnSizes)[*returnSize]
        = levelSize;
result-[*returnSize]= (int*)calloc
```

```
(levelSize, sizeof(int));

for (int i = 0; i < levelSize; i++) {
    struct TreeNode* node = queue[front++];
    result[*returnSize][i] = node->val;

    if (node->left) queue[rear++] = node->left;
    if (node->right) queue[rear++] =
                        node->right;
  }
  (*returnSize)++;
 }
 free(queue);
  return result;
 }
```

Case: 1

Input:

root = [3, 9, 20, null, null, 15, 7]

output : [[3], [9, 20], [15, 7]]

Case: 2 : Input : root = [1]

output = [[1]]

Case 3 : Input : root = []

output : []