29/2/24

HackerRank

```
typedef struct TreeNode {
    int data;
    struct TreeNode * left;
    struct TreeNode * right;
} TreeNode;

void inOrderTraversal (TreeNode * root, int *
    result, int * index){
  if (root == NULL)
      return;
  inOrderTraversal (root -> left , result, index);
  result [(*index) ++] = root -> data;
  inOrderTraversal (root -> right , result, index);
}

void swapSubtrees (TreeNode * root , int k,
    int depth) {
  if (root == NULL)
      return;
  if ( depth % k == 0){
    TreeNode * temp = root -> left ;
    root -> left  = root -> right;
    root -> right = temp;
  }

  swapSubtrees (root -> left , k, depth+1);
  swapSubtrees ( root -> right , k, depth+1);
}
```

```
TreeNode * buildTree (int indexes-rows,
    int indexes-columns, int ** indexes) {
  TreeNode * root = (TreeNode *) malloc
                (sizeof (TreeNode));
  root -> data = 1;
  root -> left = NULL;
  root -> right = NULL;

  TreeNode * nodes [indexes-rows + 1];
  nodes [1] = root;

  for(int i = 0; i < indexes-rows ; i++) {
      TreeNode * curr = nodes [i+1];

      if (indexes [i][0] != -1) {
          curr -> left = (TreeNode *) malloc
                (sizeof (TreeNode));
          curr -> left -> data = indexes[i][0];
          curr -> left -> left = NULL;
          curr -> left -> right = NULL;
          nodes [indexes [i][0]] = curr -> left;
      }
  }
  return root;
}
```

```
int ** swapNodes (int indexes_rows, int
indexes_columns, int ** indexes, int queries_
count, int * queries, int * result_rows,
int * result_columns)
{
    int ** result = (int **) malloc (queries_
            count * sizeof (int *));
    *result_rows = queries_count;
    *result_columns = indexes_rows;

    TreeNode * root = buildTree (indexes_rows,
            indexes_columns, indexes);

    for (int i=0 ; i< queries_count ; i++)
    {
        int k = queries[i];
        swapSubtrees (root, k, k);

        int * traversal = (int *) malloc
            (indexes_rows * sizeof (int));
        int index=0;
        inOrderTraversal (root, traversal, &index);
        result[i] = traversal;
    }

    return result;
}
```

~~OUTPUT~~ :-

INPUT -

3

2 3

-1 -1

-1 -1

2

1

1

OUTPUT

3 1 2

2 1 3

29/12/24