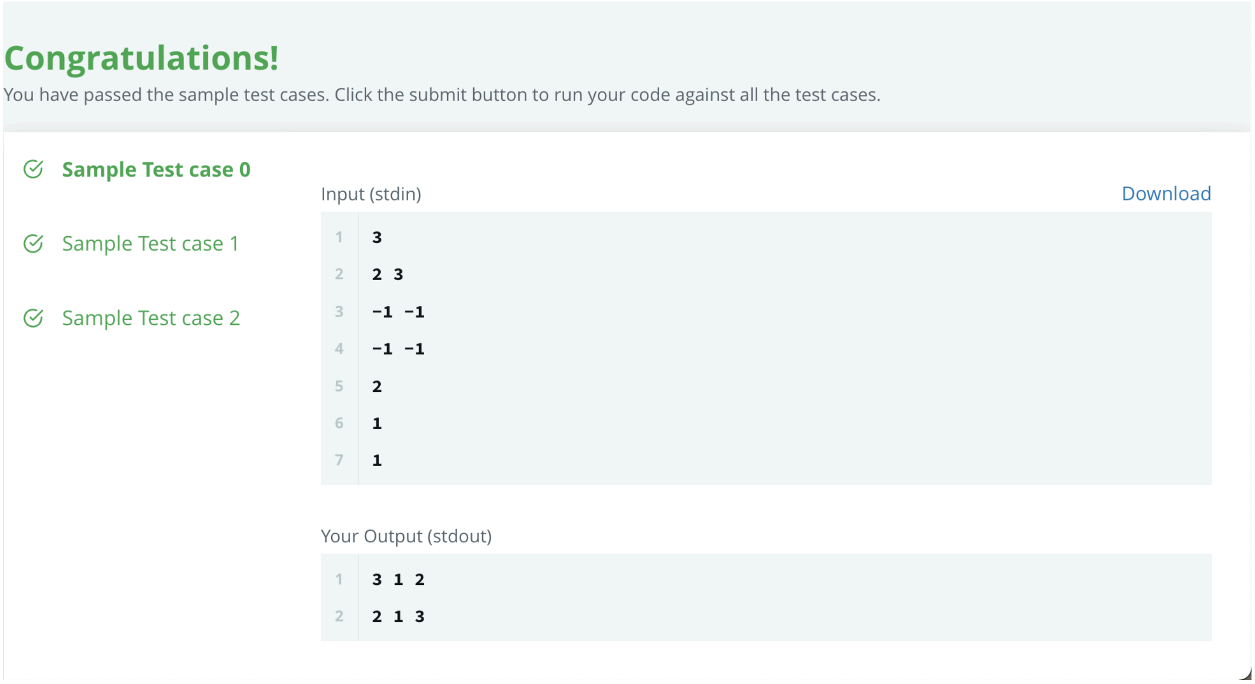
Hackerrank-Swapping of trees

```
#include <stdio.h>
#include <stdlib.h>

struct node
{
    int id;
    int depth;
    struct node *left, *right;
};

void
inorder(struct node* tree)
{
    if(tree == NULL)
        return;

    inorder(tree->left);
    printf("%d ",tree->id);
    inorder((tree->right));

}


int
main(void)
{
    int no_of_nodes, i = 0;
    int l,r, max_depth,k;
    struct node* temp = NULL;
    scanf("%d",&no_of_nodes);
    struct node* tree = (struct node *) calloc(no_of_nodes , sizeof(struct node));
    tree[0].depth = 1;
    while(i <  no_of_nodes )
    {
```

```
    tree[i].id = i+1;
    scanf("%d %d",&l,&r);
    if(l ==  -1)
        tree[i].left = NULL;
    else
         {
             tree[i].left = &tree[l-1];
             tree[i].left->depth = tree[i].depth + 1;
             max_depth = tree[i].left->depth;
         }

     if(r ==  -1)
        tree[i].right = NULL;
    else
         {
             tree[i].right = &tree[r-1];
             tree[i].right->depth = tree[i].depth + 1;
             max_depth = tree[i].right->depth+2;
         }

i++;
}

scanf("%d", &i);
while(i--)
{
    scanf("%d",&l);
    r = l;
    while(l <= max_depth)
    {
        for(k = 0;k < no_of_nodes; ++k)
        {
            if(tree[k].depth == l)
            {
                temp = tree[k].left;
                tree[k].left = tree[k].right;
```

```
                    tree[k].right = temp;
                }
            }
            l = l + r;
        }
        inorder(tree);
        printf("\n");
    }



    return 0;
}
```



Congratulations!

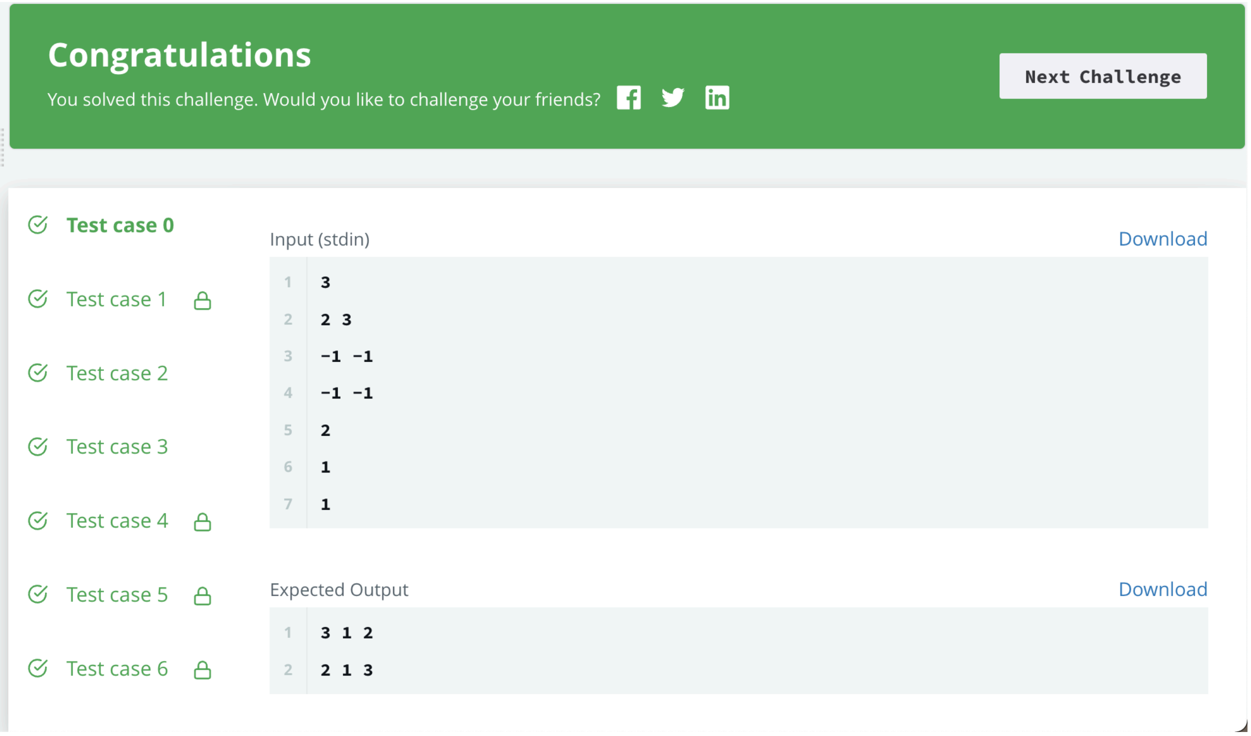
You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
3
2 3
-1 -1
-1 -1
2
1
1
```

Your Output (stdout)

```
3 1 2
2 1 3
```



Congratulations

You solved this challenge. Would you like to challenge your friends?

- ✓ **Test case 0**
- ✓ Test case 1 🔒
- ✓ Test case 2
- ✓ Test case 3
- ✓ Test case 4 🔒
- ✓ Test case 5 🔒
- ✓ Test case 6 🔒

Input (stdin)

```
3
2 3
-1 -1
-1 -1
2
1
1
```

Expected Output

```
3 1 2
2 1 3
```