HackerRack code:

```
#include <assert.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct Node {
   int data;
   struct Node* left;
   struct Node* right;
} Node;

Node* createNode(int data) {
   Node* newNode = (Node*)malloc(sizeof(Node));
   newNode->data = data;
   newNode->left = NULL;
   newNode->right = NULL;
   return newNode;
}

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

void swapAtLevel(Node* root, int k, int level) {
   if (root == NULL) return;
   if (level % k == 0) {
      Node* temp = root->left;
      root->left = root->right;
      root->right = temp;
   }
   swapAtLevel(root->left, k, level + 1);
   swapAtLevel(root->right, k, level + 1);
}

int** swapNodes(int indexes_rows, int indexes_columns, int** indexes, int queries_count, int* q
ueries, int* result_rows, int* result_columns) {

   Node** nodes = (Node**)malloc((indexes_rows + 1) * sizeof(Node*));
   for (int i = 1; i <= indexes_rows; i++) {
      nodes[i] = createNode(i);
```

```
    }

    for (int i = 0; i < indexes_rows; i++) {
        int leftIndex = indexes[i][0];
        int rightIndex = indexes[i][1];
        if (leftIndex != -1) nodes[i + 1]->left = nodes[leftIndex];
        if (rightIndex != -1) nodes[i + 1]->right = nodes[rightIndex];
    }


    int** result = (int**)malloc(queries_count * sizeof(int*));
    *result_rows = queries_count;
    *result_columns = indexes_rows;
    for (int i = 0; i < queries_count; i++) {
        swapAtLevel(nodes[1], queries[i], 1);
        int* traversalResult = (int*)malloc(indexes_rows * sizeof(int));
        int index = 0;
        inOrderTraversal(nodes[1], traversalResult, &index);
        result[i] = traversalResult;
    }

    free(nodes);
    return result;
}

int main() {
    int n;
    scanf("%d", &n);

    int** indexes = malloc(n * sizeof(int*));
    for (int i = 0; i < n; i++) {
        indexes[i] = malloc(2 * sizeof(int));
        scanf("%d %d", &indexes[i][0], &indexes[i][1]);
    }

    int queries_count;
    scanf("%d", &queries_count);

    int* queries = malloc(queries_count * sizeof(int));
    for (int i = 0; i < queries_count; i++) {
        scanf("%d", &queries[i]);
    }

    int result_rows;
    int result_columns;
```

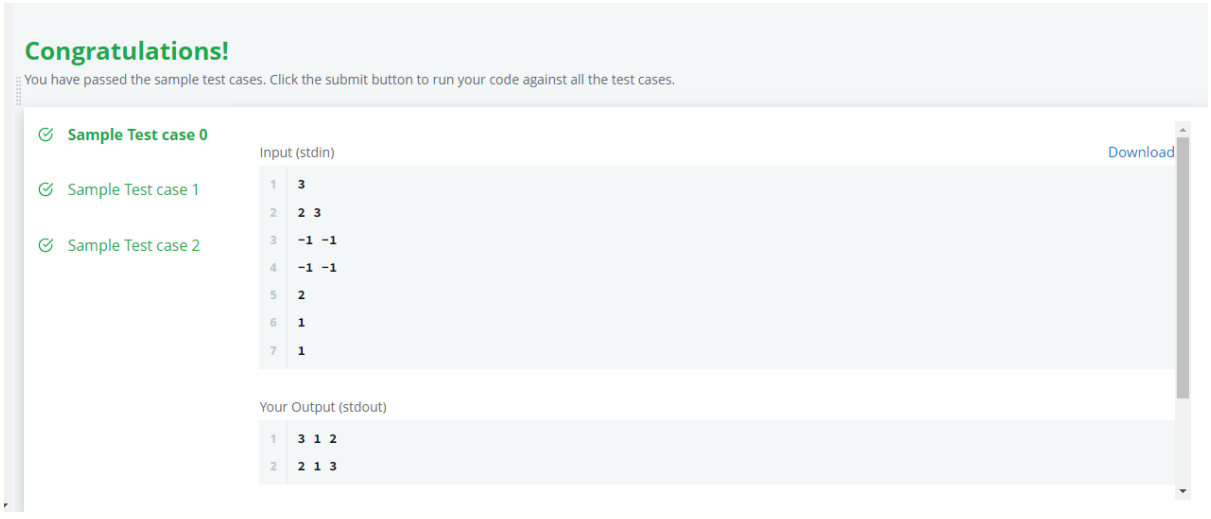
```
    int** result = swapNodes(n, 2, indexes, queries_count, queries, &result_rows, &result_column
s);

    for (int i = 0; i < result_rows; i++) {
        for (int j = 0; j < result_columns; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
        free(result[i]);
    }
    free(result);

    for (int i = 0; i < n; i++) {
        free(indexes[i]);
    }
    free(indexes);
    free(queries);

    return 0;
}
```

OUTPUT:



Congratulations!

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

- **Sample Test case 0**
- Sample Test case 1
- Sample Test case 2

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

Your Output (stdout)

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

Expected Output Download

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