JOHNSON TROTTER

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

#define LEFT -1
#define RIGHT 1

typedef struct {
    int value;
    int dir;
} Element;

void printPerm(Element *a, int n) {
    for (int i = 0; i < n; i++)
        printf("%d ", a[i].value);
    printf("\n");
}

int getMobile(Element *a, int n) {
    int mobile = 0;
    int mobileIndex = -1;
    for (int i = 0; i < n; i++) {
        if (a[i].dir == LEFT && i != 0 && a[i].value > a[i - 1].value && a[i].value > mobile) {
            mobile = a[i].value;
            mobileIndex = i;
        }
        if (a[i].dir == RIGHT && i != n - 1 && a[i].value > a[i + 1].value && a[i].value >
mobile) {
            mobile = a[i].value;
```

```
            mobileIndex = i;
        }
    }
    return mobileIndex;
}

void generatePermutations(int n) {
    Element *a = malloc(n * sizeof(Element));
    for (int i = 0; i < n; i++) {
        a[i].value = i + 1;
        a[i].dir = LEFT;
    }

    printPerm(a, n);

    for (int i = 1; i < tgamma(n + 1); i++) {
        int mobileIndex = getMobile(a, n);
        if (mobileIndex == -1) break;

        int dir = a[mobileIndex].dir;
        int swapIndex = mobileIndex + dir;
        Element temp = a[mobileIndex];
        a[mobileIndex] = a[swapIndex];
        a[swapIndex] = temp;

        mobileIndex = swapIndex;
        for (int j = 0; j < n; j++) {
            if (a[j].value > a[mobileIndex].value)
                a[j].dir = -a[j].dir;
```

```
        }
        printPerm(a, n);
    }

    free(a);
}

int main() {
    int n;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    generatePermutations(n);
    return 0;
}
```

OUTPUT: