

OUTPUT FOR JOHNSON TROTTER

The image shows two side-by-side screenshots of the OnlineCCompiler interface. Both windows have the title "Online C Compiler - online editor".

Top Window (Left):

```
main.c .swp ;  
1 #include<stdio.h>  
2 #include<stdlib.h>  
3 bool LEFT_TO_RIGHT = true;  
4 bool RIGHT_TO_LEFT = false;  
5 void swap(int *a, int *b) {  
6     int temp = *a;  
7     *a = *b;  
8     *b = temp;  
9 }  
10 int searchArr(int a[], int n, int mobile) {  
11     for (int i = 0; i < n; i++) {  
12         if (a[i] == mobile) {  
13             return i + 1;  
14         }  
15     }  
16     int getMobile(int a[], bool dir[], int n) {  
17         int mobile_prev = 0, mobile = 0;  
18         for (int i = 0; i < n; i++) {  
19             if (dir[a[i] - 1] == RIGHT_TO_LEFT)  
20             {  
21                 if (a[i] > a[i - 1] && a[i] > mobile_prev)  
22                 {  
23                     mobile = a[i];  
24                     mobile_prev = mobile;  
25                 }  
26                 if (dir[a[i] - 1] == LEFT_TO_RIGHT && i != n - 1)  
27                 {  
28                     if (a[i] > a[i + 1] && a[i] > mobile_prev)  
29                     {  
30                         mobile = a[i];  
31                         mobile_prev = mobile;  
32                     }  
33                 }  
34             if (mobile == 0 && mobile_prev == 0)  
35                 return 0;  
36             else  
37                 return mobile;  
38         }  
39         int printOnePerm(int a[], bool dir[], int n) {  
40             int mobile = getMobile(a, dir, n);  
41             if (mobile == 0)  
42                 return;  
43             printf("%d", a[0]);  
44             for (int i = 1; i < n; i++)  
45                 if (dir[a[i] - 1] == RIGHT_TO_LEFT)  
46                     swap(&a[i - 1], &a[pos - 1]);  
47                 else if (dir[a[i] - 1] == LEFT_TO_RIGHT)  
48                     swap(&a[i], &a[pos]);  
49                 pos = (int) i + 1; i < n; i++)  
50                 if (a[i] > mobile)  
51                 {  
52                     if (dir[a[i] - 1] == LEFT_TO_RIGHT)  
53                         dir[a[i] - 1] = RIGHT_TO_LEFT;  
54                     else if (dir[a[i] - 1] == RIGHT_TO_LEFT)  
55                         dir[a[i] - 1] = LEFT_TO_RIGHT;  
56                     else if (dir[a[i] - 1] == RIGHT_TO_LEFT)  
57                         dir[a[i] - 1] = LEFT_TO_RIGHT;  
58                 }  
59             for (int i = 0; i < n; i++)  
60                 printf("%d", a[i]);  
61             printf(" ");  
62         }  
63         int fact(int n) {  
64             int res = 1;  
65             for (int i = 1; i < n; i++)  
66                 res = res * i;  
67             return res;  
68         }  
69         void printPermutation(int n) {  
70             int a[n];  
71             bool dir[n];  
72             for (int i = 0; i < n; i++)  
73                 a[i] = i + 1;  
74             for (int i = 0; i < n; i++)  
75                 dir[i] = RIGHT_TO_LEFT;  
76             printOnePerm(a, dir, n);  
77         }  
78         int main() {  
79             int n;  
80             printf("Enter the value of n: ");  
81             scanf("%d", &n);  
82             printf("All the permutations are : \n");  
83             printPermutation(n);  
84         }  
85     }
```

Bottom Window (Right):

```
main.c .swp ;  
1 int pos = searchArr(a, n, mobile);  
2 if (dir[a[pos - 1] - 1] == RIGHT_TO_LEFT)  
3     swap(&a[pos - 1], &a[pos - 2]);  
4 else if (dir[a[pos - 1] - 1] == LEFT_TO_RIGHT)  
5     swap(&a[pos], &a[pos - 1]);  
6 pos = (int) i + 1; i < n; i++)  
7 if (a[i] > mobile)  
8 {  
9     if (dir[a[i] - 1] == LEFT_TO_RIGHT)  
10         dir[a[i] - 1] = RIGHT_TO_LEFT;  
11     else if (dir[a[i] - 1] == RIGHT_TO_LEFT)  
12         dir[a[i] - 1] = LEFT_TO_RIGHT;  
13     else if (dir[a[i] - 1] == RIGHT_TO_LEFT)  
14         dir[a[i] - 1] = LEFT_TO_RIGHT;  
15 }  
16 for (int i = 0; i < n; i++)  
17     printf("%d", a[i]);  
18 printf(" ");  
19 for (int i = 0; i < n; i++)  
20     dir[i] = RIGHT_TO_LEFT;  
21 for (int i = 1; i < fact(n); i++)  
22     printOnePerm(a, dir, n);  
23  
24 int main() {  
25     int n;  
26     printf("Enter the value of n: ");  
27     scanf("%d", &n);  
28     printf("All the permutations are : \n");  
29     printPermutation(n);  
30 }
```

The screenshot shows two instances of the Online C Compiler interface. Both instances have the same code displayed in the editor:

```

main.c
49 swap(&a[pos], &a[pos - 1]);
50 for (int i = 0; i < n; i++)
51 {
52     if (a[i] > mobile)
53     {
54         if (dir[a[i] - 1] == LEFT_TO_RIGHT)
55             dir[a[i] - 1] = RIGHT_TO_LEFT;
56         else if (dir[a[i] - 1] == RIGHT_TO_LEFT)
57             dir[a[i] - 1] = LEFT_TO_RIGHT;
58     }
59     for (int i = 0; i < n; i++)
60     {
61         print("%d", a[i]);
62         print(" ");
63     }
64     int fact(int n)
65     {
66         int res = 1;
67         for (int i = 1; i <= n; i++)
68             res = res * i;
69         return res;
70     }
71     void printPermutation(int n)
72     {
73         int a[n];
74         bool dir[n];
75         for (int i = 0; i < n; i++)
76         {
77             a[i] = i + 1;
78             getint("%d", a[i]);
79         }
80         print(" ");
81         for (int i = 0; i < fact(n); i++)
82             printOnePerm(a, dir, n);
83     }
84     int main()
85     {
86         int n;
87         print("Enter the value of n: ");
88         scanf("%d", &n);
89         print("All the permutations are : \n");
90         printPermutation(n);
91         print("\n");
92     }
}

```

In the bottom console window, the user enters the value of n (4) and the program outputs all 24 permutations of the array [1, 2, 3, 4]. The output is as follows:

```

Enter the value of n: 4
All the permutations are :
1234 1243 1423 4123 4132 1432 1342 1324 3124 3142 3412 4312 4321 3421 3241 3214 2314 2341 2431 4231 4213 2413 2143 2134
...Program Finished with exit code 0
Press ENTER to exit console.

```

CODE FOR JOHNSON TROTTER

```
#include<stdio.h>

#include<stdbool.h>

bool LEFT_TO_RIGHT = true;

bool RIGHT_TO_LEFT = false;
```

```

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int searchArr(int a[], int n, int mobile) {
    for (int i = 0; i < n; i++) {
        if (a[i] == mobile) {
            return i + 1;
        }
    }
}

int getMobile(int a[], bool dir[], int n) {
    int mobile_prev = 0, mobile = 0;
    for (int i = 0; i < n; i++) {
        if (dir[a[i] - 1] == RIGHT_TO_LEFT && i != 0) {
            if (a[i] > a[i - 1] && a[i] > mobile_prev) {
                mobile = a[i];
                mobile_prev = mobile;
            }
        }
        if (dir[a[i] - 1] == LEFT_TO_RIGHT && i != n - 1) {
            if (a[i] > a[i + 1] && a[i] > mobile_prev) {
                mobile = a[i];
            }
        }
    }
}

```

```

mobile_prev = mobile;
}
}
}

if (mobile == 0 && mobile_prev == 0)
return 0;
else
return mobile;
}

int printOnePerm(int a[], bool dir[], int n) {
int mobile = getMobile(a, dir, n);
int pos = searchArr(a, n, mobile);
if (dir[a[pos - 1] - 1] == RIGHT_TO_LEFT)
swap(&a[pos - 1], &a[pos - 2]);
else if (dir[a[pos - 1] - 1] == LEFT_TO_RIGHT)
swap(&a[pos], &a[pos - 1]);
for (int i = 0; i < n; i++)
{
if (a[i] > mobile)
{
if (dir[a[i] - 1] == LEFT_TO_RIGHT)
dir[a[i] - 1] = RIGHT_TO_LEFT;
else if (dir[a[i] - 1] == RIGHT_TO_LEFT)
dir[a[i] - 1] = LEFT_TO_RIGHT;
}
}
for (int i = 0; i < n; i++)
printf("%d", a[i]);
printf(" ");

```

```
}

int fact(int n) {
    int res = 1;
    for (int i = 1; i <= n; i++)
        res = res * i;
    return res;
}

void printPermutation(int n) {
    int a[n];
    bool dir[n];
    for (int i = 0; i < n; i++)
    {
        a[i] = i + 1;
        printf("%d", a[i]);
    }
    printf(" ");
    for (int i = 0; i < n; i++)
        dir[i] = RIGHT_TO_LEFT;
    for (int i = 1; i < fact(n); i++)
        printOnePerm(a, dir, n);
}

int main() {
    int n;
    printf("Enter the value of n: ");
    scanf("%d", &n);
    printf("All the permutations are : \n");
    printPermutation(n);
    printf("\n");
    return 0;
}
```