**LAB-3**

#include <stdio.h>

#define RIGHT\_TO\_LEFT 0

#define LEFT\_TO\_RIGHT 1

int searchArr(int a[], int n, int mobile)

{

for (int i = 0; i < n; i++)

if (a[i] == mobile)

return i + 1;

return -1; // Element not found

}

int getMobile(int a[], int 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;

}

}

}

return mobile;

}

void swap(int\* a, int\* b)

{

int temp = \*a;

\*a = \*b;

\*b = temp;

}

void printOnePerm(int a[], int 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("\n");

}

int fact(int n)

{

int res = 1;

for (int i = 1; i <= n; i++)

res = res \* i;

return res;

}

void printPermutation(int n, int dir[])

{

int a[n];

for (int i = 0; i < n; i++) {

a[i] = i + 1;

printf("%d ", a[i]);

}

printf("\n");

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);

int dir[n];

for (int i = 0; i < n; i++) {

printf("Enter the direction (0 for RIGHT\_TO\_LEFT, 1 for LEFT\_TO\_RIGHT) of element %d: ", i + 1);

scanf("%d", &dir[i]);

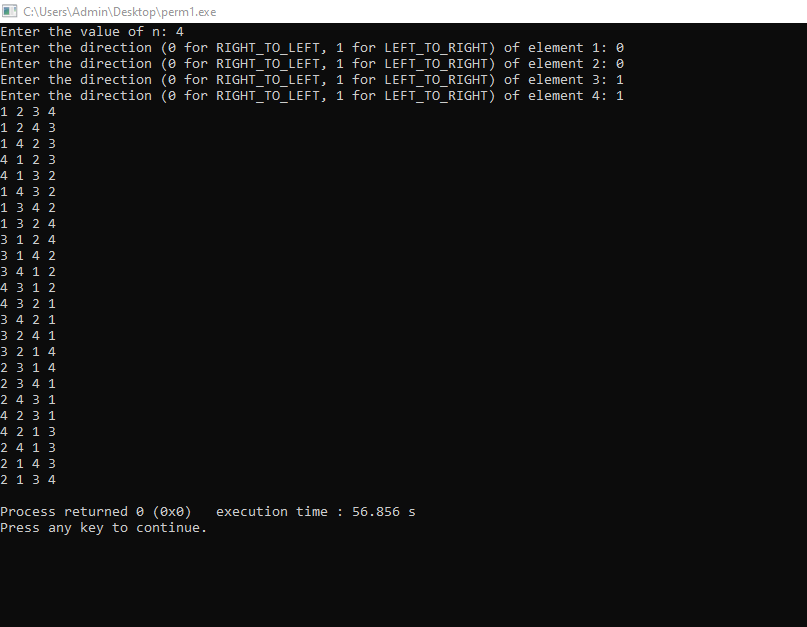
}

printPermutation(n, dir);

return 0;

}

**OUTPUT:**

****