**ADA LAB-7**

* **JOHNSON TROTTER ALGORITHM TO GENERATE PERMUTATIONS**
  + **PROGRAM**

#include <bits/stdc++.h>

using namespace std;

bool LEFT\_TO\_RIGHT = true;

bool RIGHT\_TO\_LEFT = false;

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

{

int i;

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

if (a[i] == mobile)

return (i + 1);

return 0;

}

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++)

cout << a[i];

cout << " ";

return 0;

}

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;

cout << a[i];

}

cout << endl;

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;

cout << "enter the no.of terms: "<<endl;

cin >> n;

cout << "permutations: "<<endl;

printPermutation(n);

return 0;

}

* + **OUTPUT**

