Table of Contents

[1. Permutari de n elemente varianta iterativa 2](#_Toc126909337)

[2. Sa se genereze si sa numere toate permutarile multimii {1,2,3,...,n}. 3](#_Toc126909338)

[3. Sa se genereze si sa numere toate permutarile multimii {1,2,3,...,n} care incep cu valoarea 1. 4](#_Toc126909339)

[4. Se citeste un numar natural n si apoi o multime cu n elemente numere naturale. Folosind interschimbari de elemente generati si afisati permutarile multimii citite. 5](#_Toc126909340)

[5. Se citeste un numar natural n. Afisati permutarile multimii 1,2,3...n in care elementele pare sunt puncte fixe (se afla pe pozitie egale cu valoarea lor). 6](#_Toc126909341)

[6. Se citeste un numar natural n. Afisati permutarile multimii 1,2,3...n in care pana la jumatate elementele sunt in ordine descrescatoare, iar de la jumatate pana la final in ordine crescatoare. 7](#_Toc126909342)

[9.Sa se afiseze toate permutarile multimii {1,2,...,n} care au proprietatea ca pentru orice element x din permutare (exceptandu-l pe primul) exista un element generat anterior care sa aiba valoare cu unu mai mica sau mai mare ca x. 10](#_Toc126909343)

[10. Se citeste un numar natural n>=4. Sa se afiseze toate permutarile multimii {1, 2, ... n} care au proprietatea ca diferenta absoluta a oricaror 2 elemente alaturate este cel putin egala cu 2. 11](#_Toc126909344)

[11. Se citeste un numar natural n (n<10) si apoi o multime A cu n elemente naturale ordonate crescator. Afisati in ordine lexicografica toate permutarile multimii A in care elementele pare sunt puncte fixe. 12](#_Toc126909345)

# Permutari de n elemente varianta iterativa

#include <iostream>

using namespace std;

int n , st[100];

void afisare(int k){

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

cout << st[i] << " ";

cout << endl;

}

bool OK(int k){

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

if(st[i] == st[k])

return 0;

return 1;

}

void back()

{

int k = 1;

st[1] = 0;

while(k > 0)

{

if(st[k]<n)

{

st[k]++;

if(OK(k))

{

if(k==n) afisare(k);

else st[++k]=0;

}

}

else

k--;

}

}

int main(){

cin >> n;

back();

return 0;

}

# Sa se genereze si sa numere toate permutarile multimii {1,2,3,...,n}.

#include<iostream>

using namespace std;

int x[100],pus[100],n,nr=0;

void Write()

{ for(int i=1;i<=n;i++) cout<<x[i]<<" ";

cout<<endl;

nr++;

}

void Perm(int k)

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

if(!pus[i])

{ x[k]=i;

pus[i]=1;

if(k==n) Write();

else Perm(k+1);

pus[i]=0;

}

}

int main()

{ cin>>n;

Perm(1);

cout<<nr;

system("pause");

return 0;

}

# Sa se genereze si sa numere toate permutarile multimii {1,2,3,...,n} care incep cu valoarea 1.

#include<iostream>

using namespace std;

int x[100],pus[100],n,nr=0;

void Write()

{ for(int i=1;i<=n;i++) cout<<x[i]<<" ";

cout<<endl;

nr++;

}

void Perm(int k)

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

if(!pus[i])

{ x[k]=i;

pus[i]=1;

if(k==n) Write();

else Perm(k+1);

pus[i]=0;

}

}

int main()

{ cin>>n;

x[1]=1;

pus[1]=1;

Perm(2);

cout<<nr;

system("pause");

return 0;

}

# 4. Se citeste un numar natural n si apoi o multime cu n elemente numere naturale. Folosind interschimbari de elemente generati si afisati permutarile multimii citite.

#include <iostream>

using namespace std;

int x[20],n;

void afis()

{

for (int i = 1; i <= n; i++) cout<<x[i]<<" ";

cout<<endl;

}

void inter(int &x, int &y)

{

int aux=x; x=y; y=aux;

}

void perm(int k, int n)

{

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

{

inter(x[k], x[i]);

if(k==n) afis();

else perm(k+1, n);

inter(x[k], x[i]);

}

}

int main()

{

cin>>n;

for(int i=1;i<=n;i++) cin>>x[i];

perm(1,n);

return 0;

}

# 5. Se citeste un numar natural n. Afisati permutarile multimii 1,2,3...n in care elementele pare sunt puncte fixe (se afla pe pozitie egale cu valoarea lor).

Exemplu:

n=5

permutarile care respecta conditia sunt:

1 2 3 4 5

1 2 5 4 3

3 2 1 4 5

3 2 5 4 1

5 2 1 4 3

5 2 3 4 1

(2 si 4 sunt puncte fixe)

#include <fstream>

using namespace std;

ifstream fin("date.in");

ofstream fout("date.out");

int n, sol[40], p[40];

void afis()

{

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

if(i%2==0) fout<<i<<" ";

else fout<<sol[i]<<" ";

fout<<endl;

}

void back(int k)

{

for(int i=1;i<=n;i=i+2)

if(p[i]==0)

{

sol[k]=i;

p[i]=1;

if(n%2==0 && k==n-1 || n%2==1 && k==n) afis();

else back(k+2);

p[i]=0;

}

}

int main()

{

fin>>n;

back(1);

fin.close();

fout.close();

}

# Se citeste un numar natural n. Afisati permutarile multimii 1,2,3...n in care pana la jumatate elementele sunt in ordine descrescatoare, iar de la jumatate pana la final in ordine crescatoare.

Exemplu:

n=5

permutarile care respecta conditia sunt:

3 2 1 4 5

4 2 1 3 5

4 3 1 2 5

5 2 1 3 4

5 3 1 2 4

5 4 1 2 3

#include <fstream>

using namespace std;

ifstream fin("date.in");

ofstream fout("date.out");

int n, sol[20], p[20];

void afis()

{

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

fout<<sol[i]<<" ";

fout<<endl;

}

int valid(int k)

{

if(k>1)

if(k<=(n+1)/2) { if(sol[k]>sol[k-1]) return 0; }

else if(sol[k]<sol[k-1]) return 0;

return 1;

}

void back(int k)

{

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

if(p[i]==0)

{

sol[k]=i;

p[i]=1;

if(valid(k)) if(k==n) afis();

else back(k+1);

p[i]=0;

}

}

int main()

{

fin>>n;

back(1);

fin.close();

fout.close();

}

7.Sa se genereze si sa numere toate permutarile multimii {1,2,3,...,n} care au proprietatea ca oricare doua elemente alaturate au paritate diferita.

Ex: pentru n=4:

1 2 3 4

1 4 3 2

2 1 4 3

...

4 3 2 1

#include<iostream>

using namespace std;

int x[100],pus[100],n,nr=0;

void Write()

{ for(int i=1;i<=n;i++) cout<<x[i]<<" ";

cout<<endl;

nr++;

}

void Perm(int k)

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

if(!pus[i])

{ x[k]=i;

pus[i]=1;

if(x[k]%2!=x[k-1]%2 || k==1)

if(k==n) Write();

else Perm(k+1);

pus[i]=0;

}

}

int main()

{ cin>>n;

Perm(1);

cout<<nr;

system("pause");

return 0;

}

8.Se citeste un numar natural n si o permutare a multimii {1,2,...,n}. Sa se afiseze permutarile multimii {1,2,...,n} in care oricare doua elemente alaturate nu au fost alaturate in parmutarea citita.

Ex. Pt n= 4 si permutarea 1 2 3 4 , o permutare care respecta regula este 2 4 1 3

#include<iostream.h>

int x[100],a[20][20],p[100],n;

void citire()

{ cin>>n;

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

cin>>p[i];

for(i=2;i<=n;i++)

{

a[p[i-1]][p[i]]=1;

a[p[i]][p[i-1]]=1;

}

}

void afis()

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

cout<<x[i]<<" ";

cout<<endl;

}

int cond(int k)

{ for(int i=1;i<=k-1;i++)

if(x[k]==x[i]) return 0;

if(k>1 && a[x[k-1]][x[k]]==1) return 0;

return 1;

}

void back(int k)

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

{ x[k]=i;

if(cond(k)) if(k==n) afis();

else back(k+1);

}

}

void main()

{ citire();

back(1);

}

# 9.Sa se afiseze toate permutarile multimii {1,2,...,n} care au proprietatea ca pentru orice element x din permutare (exceptandu-l pe primul) exista un element generat anterior care sa aiba valoare cu unu mai mica sau mai mare ca x.

#include<iostream.h>

int x[50],n;

void afis()

{ int i;

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

cout<<x[i];

cout<<endl;

}

int cond(int k)

{ int i;

for(i=1;i<=k-1; i++)

if(x[k]==x[i]) return 0;

if(k>1)

{ for(i=1;i<=k-1;i++)

if(x[k]-x[i]==1 || x[k]-x[i]==-1) return 1;

return 0;

}

return 1;

}

void back(int k)

{int i;

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

{x[k]=i;

if(cond(k))

if(k==n) afis();

else back(k+1);

}

}

void main()

{ cin>>n;

back(1);

}

# 10. Se citeste un numar natural n>=4. Sa se afiseze toate permutarile multimii {1, 2, ... n} care au proprietatea ca diferenta absoluta a oricaror 2 elemente alaturate este cel putin egala cu 2.

Ex: Pentru n=4 se obtin permutarile 2 4 1 3 si 3 1 4 2.

#include<iostream.h>

int x[100],p[100],n;

void afis()

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

cout<<x[i]<<" ";

cout<<endl;

}

int cond(int k)

{ if(k>1) if(x[k]-x[k-1]==1 || x[k]-x[k-1]==-1) return 0;

return 1;

}

void back(int k)

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

if(!p[i])

{ x[k]=i;

p[i]=1;

if(cond(k)) if(k==n) afis();

else back(k+1);

p[i]=0;

}

}

void main()

{ cin>>n;

back(1);

}

# 11. Se citeste un numar natural n (n<10) si apoi o multime A cu n elemente naturale ordonate crescator. Afisati in ordine lexicografica toate permutarile multimii A in care elementele pare sunt puncte fixe.

Exemplu: n=5 si A={1,4,6,7,9} => se vor genera permutarile in care 4 si 6 nu isi modifica pozitia (6 solutii).

#include <fstream>

using namespace std;

ifstream fin("p1.in");

ofstream fout("p1.out");

int A[11],P[11],X[11],n;

void afis()

{

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

fout<<A[X[i]]<<" ";

fout<<endl;

}

void back(int k)

{

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

if(!P[i])

{

P[i]=1;

X[k]=i;

if(A[i]%2!=0 || X[k]==k) //impar sau punct fix

if(k==n) afis();

else back(k+1);

P[i]=0;

}

}

int main()

{

fin>>n;

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

fin>>A[i];

back(1);

return 0;

}