

1. Generarea permutărilor

Se citește un număr natural n . Se cere să se tipărească toate permutările mulțimii $\{1,2,\dots,n\}$ în ordine lexicografică. **De exemplu**, pentru $n=3$ avem 1 2 3, 1 3 2, 2 1 3, 2 3 1, 3 1 2, 3 2 1.

Definiție: Permutările unei mulțimi A cu n elemente reprezintă toate modurile de aranjare a elementelor mulțimii A astfel încât fiecare element să fie scris o singură dată.

Reprezentarea soluției: - fiecare componentă are valori în mulțimea $\{1,2,\dots,n\}$
-soluția are n componente

Condiția de validare: - fiecare componentă apare o singură dată

Numărul permutărilor: $n! = 1*2*\dots*n$

Varianta iterativă

```
1. #include<iostream>
2. using namespace std;
3. int x[30];
4. int n,nr;
5.
6. void tipar()
7. {
8.     int i;
9.     for(i=1;i<=n;i++)
10.        cout<<x[i]<<' ';
11.    cout<<'\n';
12.    nr++;
13. }
14.
15. int valid(int k)
16. {
17.     int i;
18.     for(i=1;i<k;i++)
19.        if(x[i]==x[k])return 0;
20.     return 1;
21. }
22.
23. void back()
24. {
25.     int i,k=1;
26.     x[1]=0;
27.     while(k>0)//stiva nu este vida
28.     {
29.         while(x[k]<n)
30.         {
31.             x[k]=x[k]+1;
32.             if(valid(k))
33.                 if(k==n) tipar();
34.             else
35.                 {k=k+1;x[k]=0;}
36.         }
37.         k=k-1;
38.     }
39. }
40.
41. int main()
42. {
43.     cin>>n;
44.     back();
45.     cout<<nr;
46. }
```

Varianta recursivă

```
1. #include<iostream>
2. using namespace std;
3. int x[30];
4. int n,nr;
5.
6. void tipar()
7. {
8.     int i;
9.     for(i=1;i<=n;i++)
10.        cout<<x[i]<<' ';
11.    cout<<'\n';
12.    nr++;
13. }
14.
15. int valid(int k)
16. {
17.     int i;
18.     for(i=1;i<k;i++)
19.        if(x[i]==x[k])return 0;
20.     return 1;
21. }
22.
23. void back(int k)
24. {
25.     int i;
26.     if(k==n+1)tipar();
27.     else
28.         for(i=1;i<=n;i++)
29.         {
30.             x[k]=i;
31.             if(valid(k)) back(k+1);
32.         }
33. }
34.
35. int main()
36. {
37.     cin>>n;
38.     back(1);
39.     cout<<nr;
40. }
```

Varianta recursivă optimizată:

```
1. #include<iostream>
2. using namespace std;
3. int x[30],s[30];
4. int n;
5.
6. void tipar()
7. {
8.     int i;
9.     for(i=1;i<=n;i++)
10.        cout<<x[i]<<' ';
11.    cout<<'\n';
12. }
13.
14. int valid(int k)
15. {
16.     int i;
17.     for(i=1;i<k;i++)
18.        if(x[i]==x[k])return 0;
19.     return 1;
20. }
21.
22. void back(int k)
23. {
24.     int i;
25.     if(k==n+1) tipar();
26.     else
27.        for(i=1;i<=n;i++)
28.           if(s[i]==0)
29.              {
30.                 x[k]=i;s[i]=1;
31.                 back(k+1);
32.                 s[i]=0;
33.              }
34. }
35.
36. int main()
37. {
38.     cin>>n;
39.     back(1);
40. }
```