```
#include<bits/stdc++.h>
using namespace std;
long long n,x[100001],m;
bool Fix[100001];
void xulynghiem()
{
    for(int t=1;t<=n;t++)
        cout<<x[t];
    cout<<'\n';
}
void QL(int i)
{
    if(i>n) //Cach viet dung NEO
    {
      xulynghiem();
      return;
    }

  for(int j=0;j<=1;j++)
     {
        x[i]=j;
        QL(i+1);
    }
}
int main()
{   ios_base::sync_with_stdio(false);
    cin.tie(NULL);     cout.tie(NULL);
    freopen("BINSTR.inp","r",stdin);
    freopen("BINSTR.out","w",stdout);
    cin>>n;
    QL(1);
    return 0;
}
```

```
#include<bits/stdc++.h>
using namespace std;
long long n,x[100001],m;
bool Fix[100001];

void xulynghiem()
{...}
void QL(int i)
{
        if(i>n)  //Cach viet dung NEO
        {
          xulynghiem();
          return;
        }
    for(int j=1;j<=n;j++)
        if(Fix[j])
         {
            x[i]=j;
            Fix[j]=false;
            QL(i+1);
            Fix[j]=true;
        }
}
int main()
{   ios_base::sync_with_stdio(false);
        cin.tie(NULL);
        cout.tie(NULL);
        freopen("npermute.inp","r",stdin);
        freopen("npermute.out","w",stdout);
        cin>>n;
        memset(Fix,true,sizeof(Fix));
        QL(1);
        return 0;
}
```

SUBSET:

Ta sẽ lập chương trình liệt kê các tập con k phần tử của tập $S = \{1,2,\dots,n\}$ theo thứ tự từ điển.

Ví dụ: $n = 5, k = 3$, có 10 tập con:

{1, 2, 3}	{1, 2, 4}	{1, 2, 5}	{1, 3, 4}	{1, 3, 5}
{1, 4, 5}	{2, 3, 4}	{2, 3, 5}	{2, 4, 5}	{3, 4, 5}

Bài toán liệt kê các tập con k phần tử của tập $S = \{1,2,\dots,n\}$ có thể quy về bài toán liệt kê các dãy k phần tử $x_{1\dots k}$, trong đó $1 \le x_1 < x_2 < \dots < x_k \le n$. Nếu sắp xếp các dãy này theo thứ tự từ điển, ta nhận thấy:

Tập con đầu tiên (cấu hình khởi tạo) là $\{1,2,\dots,k\}$.
Tập con cuối cùng (cấu hình kết thúc) là $\{n-k+1, n-k+2,\dots,n\}$.
Xét một tập con $\{x_{1\dots k}\}$ trong đó $1 \le x_1 < x_2 < \dots < x_k \le n$, ta có nhận xét rằng giới hạn trên (giá trị lớn nhất có thể nhận) của x_k là n, của x_{k-1} là $n-1$, của x_{k-2} là $n-2$... Tổng quát: giới hạn trên của x_i là $n-k+i$.

Còn tất nhiên, giới hạn dưới (giá trị nhỏ nhất có thể nhận) của x_i là $x_{i-1}+1$.

Từ một dãy $x_{1\dots k}$ đại diện cho một tập con của S, nếu tất cả các phần tử trong x đều đã đạt tới giới hạn trên thì x là cấu hình cuối cùng, nếu không thì ta phải sinh ra một dãy mới tăng dần thoả mãn: dãy mới vừa đủ lớn hơn dãy cũ theo nghĩa không có một dãy k phần tử nào chen giữa chúng khi sắp thứ tự từ điển.

Ví dụ: $n = 9, k = 6$. Cấu hình đang có $x = \left(1,2, \underline{6,7,8,9}\right)$. Các phần tử $x_{3\ldots6}$ đã đạt tới giới hạn trên, nên để sinh cấu hình mới ta không thể sinh bằng cách tăng một phần tử trong số các phần tử $x_{3\ldots6}$ lên được, ta phải tăng $x_2 = 2$ lên 1 đơn vị thành $x_2 = 3$. Được cấu hình mới $x = (1,3,6,7,8,9)$. Cấu hình này lớn hơn cấu hình trước nhưng chưa thoả mãn tính chất *vừa đủ lớn*. Muốn tìm cấu hình vừa đủ lớn hơn cấu hình cũ, cần có thêm thao tác: Thay các giá trị $x_{3\ldots6}$ bằng các giới hạn dưới của chúng. Tức là:

$$x_3 = x_2 + 1 = 4$$
$$x_4 = x_3 + 1 = 5$$
$$x_5 = x_4 + 1 = 6$$
$$x_6 = x_5 + 1 = 7$$

Ta được cấu hình mới $x = (1,3,4,5,6,7)$ là cấu hình kế tiếp. Tiếp tục với cấu hình này, ta lại nhận thấy rằng $x_6 = 7$ chưa đạt giới hạn trên, như vậy chỉ cần tăng x_6 lên 1 là được cấu hình mới $x = (1,3,4,5,6,8)$.

Giá trị cận dưới và cận trên của x_i là:

$$x_{i-1} + 1 \leq x_i \leq n - k + i$$

(**Error! No text of specified style in document.**.1)

(Giả thiết rằng có thêm một số $x_0 = 0$ khi xét công thức (**Error! No text of specified style in document.**.1) với $i = 1$)

Thuật toán quay lui sẽ xét tất cả các cách chọn x_1 từ 1 $(= x_0 + 1)$ đến $n - k + 1$, với mỗi giá trị đó, xét tiếp tất cả các cách chọn x_2 từ $x_1 + 1$ đến $n - k + 2$, … cứ như vậy khi chọn được đến x_k thì ta có một cấu hình cần liệt kê.

Code tham khảo:

```
#include<bits/stdc++.h>
using namespace std;
long long n,x[100001],m;
void xulynghiem()
{ ...}

void QL(int i)
{
    if(i>m) //Cach viet dung NEO
    {
      xulynghiem();
      return;
    }

   for(int j=x[i-1]+1;j<=n;j++)
    {
        x[i]=j;
      // Cach viet cu:  if(i==m) xulynghiem();  else
        QL(i+1);
    }
}
int main()
{  ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout.tie(NULL);
       freopen("subset.inp","r",stdin);
    freopen("subset.out","w",stdout);
    cin>>n>>m;
    QL(1);
    return 0;
}
```