

## 递归实现指类型枚举

这一栏十分抽象 隐晦难懂 没有计算机思维难以想到，也难以理解 **题目如下** 

对于这道题，我们依旧采取递归的思想来解决

- 设计函数  $f(i, j, n)$ 
  - 其中  $i$  表示一个位置
  - $j$  表示  $i$  位置能存放的最小数字
  - $n$  表示  $i$  位置能存放的最大数字
- 边界条件：  $j > n$
- $f(i, j, n)$  可以分成多种情况
  - $i$  取  $j$  的情况下加上  $f(i+1, j+1, n)$
  - $i$  取  $j+1$  的情况下加上  $f(i+1, j+2, n)$
  - $i$  取  $j+2$  的情况下加上  $f(i+1, j+2, n)$
  - ..... 因此当  $i$  位置存放的数字  $j > n$  时，不满足题目要求，return 即可

代码实现：

```
#include<iostream>
using namespace std;
int arr[10];
void print(int n)
{
    for(int i=0;i<=n;i++)
    {
        if(i) cout<<" ";
        cout<<arr[i];
    }
    cout<<endl;
}
int f(int i,int j,int n)
{
    if(j>n) return 0 ; //边界条件
    for(;j<=n;j++)
    {
        arr[i]=j;
        print(i);          //循环起始下标是0，因此循环上限应是i，即当前状态的位置，因为要遍
        f(i+1,j+1,n);    到起始位置到当前位置并进行输出
    }
}
```

```
int main()
{
    int n;
    cin >> n;
    f(0,1,n); //起始状态
    return 0;
}
```

## 递归实现组合型枚举



- 设计函数  $f(i, j, n, m)$ 
  - $i$  表示当前位置
  - $j$  表示当前位置可选的最小值
  - $n$  表示当前位置可选的最大值
  - $m$  表示可选的数字的个数
- 边界条件：  $i=m$   $i$  是从位置 0 开始，因此只需枚举到  $m-1$  即可
- $f(i, j, n)$  可以分成多种情况
  - $i$  取  $j$  的情况下加上  $f(i+1, j+1, n)$
  - $i$  取  $j+1$  的情况下加上  $f(i+1, j+2, n)$
  - $i$  取  $j+2$  的情况下加上  $f(i+1, j+2, n)$
  - .....

```
#include<iostream>
using namespace std;
int n,m;
int a[10];
void print(int m)
{
    for(int i=0;i<m;i++)
        cout << a[i]<<" ";
    cout<<endl;
}
void f(int i,int j,int n,int m)
{
    if(i==m){print(m);return;}
    else
    {
        for(;j<=n && m-1-i<=n-j ;j++)
            /*m-1-i<=n-j的意思是从i位置开始枚举到m-1位置时数字一定要够，因此m-1-i代表的是从
+1位置到m-1位置剩下的位置个数，n-j代表的是i+1位置开始最多可填的数字个数，我们要保证*/
```

```

    数字个数大于等于位置个数*/
{
    a[i]=j;
    f(i+1,j+1,n,m);
}

}

int main()
{
    cin >> n >> m;
    f(0,1,n,m);
    return 0;
}

```

## 递归实现排列型枚举



- 设计函数f(i, n)
  - i表示当前位置
  - n表示最大的整数，也表示一组里面数字的个数
- **边界条件：**当i==n，即当前位置i达到了数字个数的上限时，进行return, print
- f(i, n) 可以表示成如下情况
  - b[0] + f(i+1,n)
  - b[1] + f(i+1,n)
  - b[2] + f(i+1,n)
  - .....
    - | b[0],b[1],b[2]分别代表前面位置未出现的数字，枚举到使用过的数字舍弃，枚举到n暂停

```

#include<iostream>
using namespace std;
int a[10];
int b[10]={0}; //判断数字是否出现过，出现过则标记为1，未出现则标记为0
int n;
void print(int n)
{
    for(int i=0;i<n;i++)
    {
        if(i) cout << " ";
        cout << a[i] ;
    }
}

```

```
cout << endl;
return ;
}
void f(int i,int n)
{
    if(i==n)
    {
        print(i);
        return ;
    }
    for(int j = 1;j <= n ; j++)      //b[0]~n, 枚举
    {
        if(b[j])
            continue;      //判断其是否在前面出现过
        a[i]=j;           //当前位置填上数字
        b[j]=1;           //标记, 表明其已经使用过
        f(i+1,n);         //进行下一个位置的遍历
        b[j]=0;           //难点！！ 执行到这一步表明函数已经返回至当前位置, 后面的枚举已经
        全部完成并且结果已经输出, 结果输出后则必须要将此数字取消标记
    }
}

int main()
{
    cin >> n;
    f(0,n);
    return 0;
}
```

**抽象，难以理解是必然的，反复看，慢慢理解吧**