枚举排列和枚举子集

19信安张华睿

枚举

有的时候是先想出一个暴力枚举的方法，让后发现它可以优化...

枚举方法的优化、空间换时间的优化...

例题1：

给出一个正整数n，请按照从小到大的顺序输出形如$abcde/fghij = n$的表达式，其中每个字母代表0～9中的一个数字且每个数字只出现一次，可以有前导0。2≤N ≤79

朴素想法：

直接上一个全排列，复杂度O(10!=3628800)

优化：减少没有必要的枚举：

$$abcde = n * fghij$$

分析一下：第一个数肯定是个五位数，第二位要么是五位数，要么是四位数

乘法来枚举 `fghij`，复杂度差不多几万

例题２ 分数拆分：

输入正整数k，找到所有的正整数x和y，满足 $x \geq y$时$1/k=1/x+1/y$。

($0 < k \leq 10000$)

$$1/k = 1/x + 1/y$$

暴力想法:

枚举x,y。但是x,y的枚举范围是什么？

继续分析：

由于 $x \geq y$，有$\frac{1}{x} \leq \frac{1}{y}$,因此$\frac{1}{k} - \frac{1}{y} \leq \frac{1}{y}$,推出：

$$y \leq 2k$$

所以只需要在2*k的范围内枚举y,然后尝试计算出x即可

- 实际上可以进一步将枚举范围减小至从[k+1,2k],

排列枚举

生成一个1-n的排列：

P1706 全排列问题

方法一：DFS枚举

方法二：下一个排列

next_permutation

下一个排列的意思：就是下一个比这个序列字典序大1的序列。字典序的大小关系等价于从头开始第一个不同位置处的大小关系。

用法举例：

```
template<class Iterator>
  bool next_permutation (Iterator first, Iterator last);
template<class Iterator, class Compare>
  bool next_permutation (Iterator first, Iterator last, Compare comp);
```

```
#include <iostream>
#include <algorithm>//头文件
using namespace std;
int main() {
  int a[3] = {1,2,3};
  do {
    cout << a[0] << a[1] << a[2] << endl;
  } while (next_permutation(a, a+3));
  return 0;
}
```

输出：

```
123
132
213
231
312
321
```

注：

最开始需要传入字典序最小的序列。

如何得到?

`sort`

枚举子集

给定一个集合，枚举所有的子集

例题3：

从一个大小为n的整数集中选取一些元素，使得它们的和等于给定的值T。每个元素限选一次，不能一个都不选。

这里主要讲一下二进制枚举法

思想就是用二进制来表示{0,1,2..n-1}的子集S,从右往左第i位（从0开始）表示元素i是否在集合S中。

枚举代码：

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    int n;
    cin >> n;
    for(int i = 0; i < (1<<n); i++){
        for(int j = 0; j < n; j++){
            if(i & (1 << j)){
                printf("%d ",j);//如果存在输出第j个元素
            }
        }
        printf("\n");
    }
    return 0;
}
```

简单组合学

~~—点高中知识~~

加法原理：

如果要完成一件事有 k 类方式，第一类方式有 n1 个方法，第二类方式有 n2 个方法，…，第 k 类方式有 nk个方法。

则完成这件事有n1+n2+ … +nk 个方法。

乘法原理：

如果要完成一件事，依次要进行 k 个步骤；做完第一个步骤有 n1 个方法，做完第二个步骤有 n2 个方法，…，做第 k 个步骤有 nk种方法。

则完成这件事有n1·n2· … ·nk 种方法。

组合数公式

组合数计算公式:

$$\binom{n}{m} = \frac{A_n^m}{m!} = \frac{n!}{m!(n-m)!}$$

组合数一些性质：

- $\binom{n}{m} = \binom{n}{n-m}$ (对称性)
- $\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}$(递推式)
- $\binom{n}{k} = \frac{n}{k}\binom{n-1}{k-1}$(定义)

组合数板子：

阶乘+快速幂求逆元：

```
//预处理阶乘表
ll fac[Maxn+10];
void setFac(int n){
        fac[0]=1;
        for(int i=1;i<=n;i++){
                fac[i]=1LL*fac[i-1]*i%mod;
        }
}
ll binaryPow(ll a,ll b,ll m){
        ll ans = 1;
    while(b){
        if(b & 1){
            ans = ans * a % m;
        }
        a = a * a % m;
        b >>= 1;
    }
    return ans;
}
//计算组合数$C_{n}^{m}$
ll C(int n,int m){
        if(n<m) return 0;
        if(n<0||m<0) return 0;
        ll t=fac[n-m]*fac[m]%mod;
    ll inv=binaryPow(t,mod-2,mod);
        return fac[n]*inv%mod;
}
```

参考资料：

- 《算法竞赛入门经典》
- 《算法竞赛进阶指南》