西北工業大學
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C程序设计
Programming in C

1011014

主讲：姜学锋，计算机学院

编程实现枚举算法策略

1、枚举算法策略

- 1．枚举算法策略
- 枚举法，又称穷举法，其基本思想是在一个有穷的问题所有可能解的集合中，按某种顺序逐一枚举各个元素，用给定的约束条件判定元素是否符合条件，若满足条件，使命题成立的候选解就是问题的解；否则，该元素就不是该问题的解。
- 枚举法本质上属于搜索策略（search strategy）

3.5.7 循环结构程序举例

- 适用枚举法求解的问题必须满足两个条件：
- （1）可预先确定每个解的元素个数n；
- （2）解元素S1，S2，…，Sn的可能值为一个连续的值域。

- 应用枚举法的场合有：求不定方程、排列组合、暴力算法（brute force enumeration）

3.5.7 循环结构程序举例

- 枚举算法因为要列举问题所有可能解，所以它具备以下几个特点：
- （1）得到的结果肯定是正确的；
- （2）可能做了很多的无用功，浪费了宝贵的时间，效率低下。
- （3）通常会涉及到求极值（如最大，最小，最重等）。
- （4）数据量大的话，可能会造成时间崩溃。

- 并不是所有的问题都可以使用枚举算法来求解，只有当问题的所有可能解的个数不太多时，并在可以接受的时间内得到问题的所有解，才可使用枚举算法。

3.5.7 循环结构程序举例

- 枚举法的优点：
- （1）由于枚举算法一般是现实生活中问题的“直译”，因此比较直观，易于理解；
- （2）由于枚举算法建立在考察大量状态、甚至是穷举所有状态的基础上，所以算法的正确性比较容易证明。

3.5.7 循环结构程序举例

- 枚举法的特点是算法简单，但是运算量大是它的缺点，当问题的规模变大，循环的阶数愈大，执行的速度愈慢
- 从全局观点使用枚举法，计算量容易过大，在局部地方使用枚举法，其效果会十分显著。

- **求解策略**
- 采用枚举算法求解问题的基本思路为：
- （1）确定枚举对象、枚举范围和判定条件；
- （2）逐一枚举可能的解，验证是否是问题的解。

- **求解方法和步骤**
- （1）首先确定可能解的集合；
- （2）抽象出解包含的参数，确定每个参数的数据范围；
- （3）对解的每个参数的数据范围采用循环语句一一枚举；
- （4）对每次枚举，根据题意给定的条件判定是否解，是否是最优解；
- （5）优化程序，以便缩小搜索范围，减少程序运行时间。

- **编程模式**：
- 编程时可以采用“循环语句+判断语句”的枚举结构，其中循环语句用于“枚举”，判断语句用于“验证条件”。

3.5.7 循环结构程序举例

▸编程模式：

- 设Si1为解元素Si的最小值，Sik为解元素Si的最大值（1≤i≤n），即S11≤S1≤S1k，S21≤S2≤S2k，……，Si1≤Si≤Sik，……，Sn1≤Sn≤Snk。

```
for S1←S11 to S1k do
    for S2←S21 to S2k do
        ……
        for Si←Si1 to Sik do
            ……
                for Sn←Sn1 to Snk do
                    if 状态(S1，…，Si，…，Sn) 满足检验条件
                        输出问题的解
```

3.5.7 循环结构程序举例

- 用枚举法求解问题时，需要按照某种方式列举候选解。为了使枚举的结果不重复又不遗漏，要抓住对象的特征，选择适当的标准分类，有次序、有规律地列举。

3.5.7 循环结构程序举例

- 针对问题的数据类型，常用的列举方式有如下三种：
- （1）顺序枚举，候选解范围内的各种情况很容易与自然数对应甚至就是自然数，可以按自然数的变化顺序去枚举。
- （2）排列枚举，候选解的数据形式是一组数的排列，可以枚举出所有候选解所在范围内的排列。
- （3）组合枚举，当候选解的数据形式为一些元素的组合时，往往需要用组合枚举。组合是无序的。

【循环程序举例】

自幂数是指一个n位自然数，它的每位数字的n次幂之和等于它本身。当n为3时，例如 $1^3+5^3+3^3=153$ ，153即是一个自幂数，也称为水仙花数。求所有水仙花数。

153

370

371

407

3.5.7 循环结构程序举例

例3.61

```c
#include <stdio.h>
int main()
{
  int i,a,b,c;
  for (i=100; i<1000; i++) {
    a=i/100; //百位
    b=i/10%10; //十位
    c=i%10; //各位
    if (i==a*a*a+b*b*b+c*c*c)
      printf("%d\n",i); //水仙花数
  }
  return 0;
}
```

【例3.14】

百钱买百鸡问题：有人有一百块钱，打算买一百只鸡。公鸡一只5元，母鸡一只3元，小鸡3只1元，求应各买多少？

例题分析

显然可以用枚举法。

以三种鸡的个数为枚举对象（分别设为x、y、z），以三种鸡的总数（x+y+z）和买鸡用的钱的总数（5x+3y+z/3）为判定条件，枚举各种鸡的个数，找到问题的解。

3.5.7 循环结构程序举例

例3.14

```c
#include <stdio.h>
int main()
{
  int x,y,z;
  for (x=0; x<=20; x++) //枚举公鸡的可能数量，最多为20
    for (y=0; y<=33; y++) //枚举母鸡的可能数量，最多为33
      for (z=0; z<=100; z++) //枚举小鸡的可能数量，最多为100
        if(z%3==0&&x+y+z==100&&5*x+3*y+z/3==100)//约束条件
          printf("公鸡=%d,母鸡=%d,小鸡=%d\n",x,y,z);
  return 0;
}
```

循环体执行了21x34x101=72114次。

3.5.7 循环结构程序举例

例3.14

```
#include <stdio.h>
```

程序运行屏幕

```
公鸡=0,母鸡=25,小鸡=75
公鸡=4,母鸡=18,小鸡=78
公鸡=8,母鸡=11,小鸡=81
公鸡=12,母鸡=4,小鸡=84
```

3.5.7 循环结构程序举例

- 在枚举算法中，枚举对象的选择是非常重要的，它直接影响着算法的时间复杂度，选择适当的枚举对象可以获得更高的效率。在枚举算法中，判定条件的确定也是重要的，如果约束条件不对或者不全面，就枚举不出正确的结果。

3.5.7 循环结构程序举例

- 枚举算法是用计算机解决问题的一种特色，特点是算法的思路简单，但运算量大。

- 当问题的规模变大，循环嵌套的层数越多，执行速度变慢。如果枚举范围太大，在时间上就难以承受，所以应尽可能考虑对枚举算法优化。

- **优化策略**
- （1）减少枚举次数
- （2）合理选择用于枚举的变量
- （3）注意枚举的顺序
- （4）减少判断每种情况的时间

例题分析

前述问题中，由于三种鸡的和是固定的，因此只要枚举二种鸡（x、y），第三种鸡就可以根据约束条件求得（z=100-x-y），这样就缩小了枚举范围变成双重循环。之所以选择z，是因为z的数量大，优化效果更好，此时循环体执行21x34=714次。

3.5.7 循环结构程序举例

例3.14

```c
#include <stdio.h>
int main()
{
  int x,y,z;
  for (x=0; x<=20; x++) //枚举公鸡的可能数量，最多为20
    for (y=0; y<=33; y++) { //枚举母鸡的可能数量，最多为33
      z=100-x-y; //小鸡的数量根据约束条件求得
      if (z%3==0 && 5*x+3*y+z/3==100) //约束条件
        printf("公鸡=%d,母鸡=%d,小鸡=%d\n",x,y,z);
    }
  return 0;
}
```

例题分析

如果能从数学角度来考虑枚举算法的进一步优化，程序的效率会大大提高。

根据题意，约束式5x+3y+z/3=100，x+y+z=100可以消去一个未知数z，得到7x+4y=100，x+y+z=100。于是只要枚举公鸡x（最多14），根据约束条件就可以求得y和z。

3.5.7 循环结构程序举例

例3.14

```
#include <stdio.h>
int main()
{
  int x,y,z;
  for (x=0; x<=14; x++) { //枚举公鸡的可能数量，最多为14
    y = 100 - 7*x;
    if (y%4 != 0) continue; //由方程知y应是4的倍数
    y = y/4 , z = 100 - x - y;
    if (z%3 != 0) continue; //由方程知z应是3的倍数
    printf("公鸡=%d,母鸡=%d,小鸡=%d\n",x,y,z);
  }
  return 0;
}
```

循环体执行14次，优化效果明显。

【循环程序举例】

三色球问题

有红、黄、绿三种颜色的球，其中红球3个，黄球3个，绿球6个。现将这12个球混放在一个盒子中，从中任意摸出8个球，编程计算摸出的球各种颜色搭配。

例题分析

从12个球中任意摸出8个球，求颜色搭配的种类。解决这类问题的一种比较简单直观的方法是应用穷举法，在可能的解空间中找出所有的搭配，然后再根据约束条件加以排除，最终筛选出正确的答案。

例题分析

针对本题，由于是任意地从12个球中摸取，一切都是随机事件，因此每种颜色的球被摸到的可能的个数如下表所示。

红球	黄球	绿球
0,1,2,3	0,1,2,3	2,3,4,5,6

显然绿球不可能被摸到0个或者1个。因为假设只摸到1个绿球，那么摸到的红球和黄球的总数一定为7，而红球与黄球全部被摸到的总数才为6，因此假设是不可能成立的。同理，绿球也不可能为0个。

例题分析

下面就要在表所划定的可能解空间的范围内寻找答案。

如果将红黄绿三色球可能被摸到的个数排列组合到一起构成解空间，那么解空间的大小为4×4×5=80种颜色搭配组合。但是在这80种颜色搭配组合中，只有满足"红球数+黄球数+绿球数=8"的才是真正的答案，其余的搭配组合都不能满足题目的要求。

3.5.7 循环结构程序举例

例3.63

```c
#include <stdio.h>
int main()
{ //三色球问题求解
  int red,yellow,green;
  printf("red yellow green\n");
  for(red=0;red<=3;red++) //红色：0，1，2，3
    for(yellow=0;yellow<=3;yellow++) //黄色：0，1，2，3
      for(green=2;green<=6;green++) //绿色：2，3，4，5，6
        if(red+yellow+green == 8)
          printf("%d   %d      %d\n",red,yellow,green);
  return 0;
}
```

CP 程序设计