

寻找自幂数

杨琦

西安交通大学计算机教学实验中心

【例】寻找自幂数



- ▶ 用户输入位数 n ，找出并显示出所有 n 位的自幂数。
- ▶ 数学家发现了很多有趣的数字。
- ▶ 比如，153，一个普通的三位数，然而 $1^3 + 5^3 + 3^3 = 153$ ，
- ▶ 即它的各位数字的三次方的和等于这个数本身。
- ▶ 更一般地，一个 n 位正整数，哪些数的各位数字的 n 次方的和加起来还等于这个数呢？
- ▶ 数学家称这样的数为自幂数，也叫自恋数。



n 为1时，自幂数称为独身，0,1,2,3,4,5,6,7,8,9都是自幂数。

n 为2时，没有自幂数。

n 为3时，自幂数称为水仙花数，153就是一个水仙花数。

$n=4$ ，称为四叶玫瑰数。

$n=5$ ，称为五角星数。

$n=6$ ，称为六合数。

$n=7$ ，称为北斗七星数。

$n=8$ ，称为八仙数。

$n=9$ ，称为九九重阳数。

【问题分析】



- ▶ n位自幂数，各位数字的n次方的和加起来还等于这个数。
- ▶ 两个要点，一是怎样找出“各位”，二是n次方的计算。
- ▶ n次方的计算有一个数学函数可用， $\text{pow}(x,n)$ 。
- ▶ 找出各位，举例来说，153，找个位，可用 $153\%10=3$ ；找十位呢？ $(153/10)\%10=5$ ，依次类推。直接求余，就是最低位的数字，除10，原来的十位就成为新的最低位，重复这一过程，就可以求出各位，直到这个数成为0。
- ▶ 还有一个问题要解决，就是构造n位数。0是最小的一位数，10的1次方是最小的两位数，10的平方是最小的三位数，那么，10的n-1次方就是最小的n位数。

【算法描述】



- ① 输入位数 n 。
- ② 计算 n 位数的起始值和终止值
 $\text{start}=10^{n-1}$, $\text{end}=10^n-1$, $i=\text{start}$
- ③ 如果 $i>\text{end}$ 转⑨。
- ④ $m=i$, $\text{sum}=0$
- ⑤ 如果 $m=0$,转⑦。
- ⑥ $d=m\%10$, $\text{sum}=\text{sum}+d^n$, $m=m/10$, 转⑤
- ⑦ 如果 $\text{sum}=i$,显示 i 。
- ⑧ $i=i+1$, 转③
- ⑨ 结束。

【源程序】



```
▶ #include <iostream>           //包含需要的头文件
▶ #include<cmath>               //数学函数需要的头文件
▶ using namespace std;         //名字空间
▶ int main() {
▶     int n;                    //表示数的位数
▶     int start,end;           //表示n位数的起始值和终止值
▶     int m;                   //待分解各位的数，即待判断的数
▶     int digit;               //某个数位的值
▶     int sum;                 //各位数的n次方的和
▶     int i;                   //循环变量，待检验的数
```



- ▶ `cout<<"求n位自幂数，请输入位数:";` `//提示信息`
- ▶ `cin>>n;` `//输入位数`
- ▶ `while(n>0){` `//大于0时计算`
- ▶ `start=pow(10,n-1);` `//n位数的起始值`
- ▶ `end=pow(10,n)-1;` `//n位数的终止值`
- ▶ `cout<<n<<"位自幂数:";` `//输出说明信息`
- ▶ `for(i=start;i<=end;i++){` `//从起始值到终止值逐个检验`
- ▶ `m=i;` `//将i赋给m`
- ▶ `//检验过程中m的值会改变，而i的值不变`
- ▶ `sum=0;` `//各位数的n次方和，检验前赋0`



```
while(m!=0){    //m开始为待检验的数
digit=m%10;    //取最低位数字
sum=sum+pow(digit,n);    //n次方，再求和
m=m/10;        //去掉个位，刚才的十位成为新个位
}
//上面的循环结束时sum就是各位数字的n次方的和
if(sum==i){    //逻辑表达式的值为true时，表示是自幂数
    cout<<i<<" ";    //显示该数
}
}
```



```
▶      cout<<endl;           //换行
▶      cout<<“求n位自幂数，请输入位数:”;
▶      cin>>n;               //再输入一个n表示位数
▶      }                     //while循环
▶      cout<<endl;
▶      return 0;
▶  }
```



【运行结果】



求n位自幂数，请输入位数:1

1位自幂数:1 2 3 4 5 6 7 8 9

求n位自幂数，请输入位数:2

2位自幂数:

求n位自幂数，请输入位数:3

3位自幂数:153 370 371 407

求n位自幂数，请输入位数:4

4位自幂数:1634 8208 9474

求n位自幂数，请输入位数:5

5位自幂数:54748 92727 93084

求n位自幂数，请输入位数:0

【思路扩展】



①本例应掌握的技巧，
一是如何分离各位数字，

二是如果一种计算会破坏（或改变）某个变量的值，而这个原始值在后面的计算中还会使用，那就先将其赋值给另一个变量，使用新变量作“破坏性”计算，随时可以通过原来的变量获得原始值。

这实际是计算机科学中常用的一种“冗余”的思想，要获得某种保障，有意使用更多的时间、空间。

【思路扩展】



②分离各位数字的相反运算是合成一个数，
例如有三个变量，a,b,c，分别存放一位整数，
比如1,2,3，如果将它们合成为a作百位，b作十位，c作各位的三位数
呢？

③C++中，int型变量能表示的最大正整数为2147483647，它不过10
位，那么有11,12,13位的自幂数吗？如果有，怎样计算呢？自幂数是
有限的吗？如果有，有多少呢？