信奥算法

sizeof()
内存大小

```cpp
#include<iostream>
using namespace std;
int main(){
    char c;
    int x;
    long long y;
    int f[100];
    cout<<sizeof(c)<<endl;
    cout<<sizeof(x)<<endl;
    cout<<sizeof(y)<<endl;
    cout<<sizeof(f)<<endl;
    return 0;
}
```

memset()
内存设置

```cpp
#include<iostream>
#include<cstring>
using namespace std;
int main(){
    const int N=5;
    int f[N]={1,2,3,4,5};
    memset(f,0,sizeof(f));
    for(int i=0;i<N;++i)cout<<f[i]<<" ";
    cout<<endl;
    memset(f,1,sizeof(f));
    for(int i=0;i<N;++i)cout<<f[i]<<" ";
    cout<<endl;
    memset(f,-1,sizeof(f));
    for(int i=0;i<N;++i)cout<<f[i]<<" ";
    cout<<endl;
    return 0;
}
```

数位分离

将非负整数x用数组储存
数组每个元素储存x的一位数字

```
4   const int N=100;
5   int d[N];
6   int x;
7   cin>>x;
8   int nD=0;
9   do{
10      d[++nD]=x%10;
11      x/=10;
12  }while(x);
```

nD计算x的位数

do-while结构

复杂度?
O(logx)
x的位数 ≈10的几次方为x

快快编程2550

数据规模

20%数据，x<=20

50%数据，x<=1000

100%数据，x<=1000000000

枚举1到x每个数
数位分离判断是否包含4

复杂度O(xlogx)

暴力解法

复杂度
O(xlogx)

```
3  bool find4(int x){
4      do{
5          if(          )
6              return 1;
7          
8      }while(x);
9      return 0;
10 }
```

```
14     int n,ans=0;
15     cin>>n;
16     for(int i=1;i<=n;i++){
17         while(find4(i))
18             i++;
19         ans++;
20     }
21     cout<<ans<<endl;
```

数据规模

20%数据，x<=20

枚举1到x每个数
数位分离判断是否包含4

50%数据，x<=1000

复杂度O(xlogx)

100%数据，x<=1000000000

复杂度O(x)也不够快

需要O(logx)算法

重叠子问题

输入样例
666

输出样例
455

动态规划

记忆化搜索

类似信息可以重复利用

百位数不放共80个
百位数是1时共81个
百位数是2时共81个
百位数是3时共81个
百位数是4时共0个
百位数是5时共81个
百位数是6时共51个

先考虑百位数放几
再考虑十位数放几
再考虑个位数放几

状态定义

已知x，求1到x里共几个数不包含4

请同学尝试设计状态	每位可以填0到9

f[p][0]	表示尾部共p位数待填写时不包含4的方案数	无其他限制
f[p][1]	表示尾部共p位数待填写时不包含4的方案数	不能超过x尾部p位数

举例：x=666

f[1][0]	9
f[1][1]	6

f[2][0]	81
f[2][1]	51

f[3][0]	729
f[3][1]	456

ans=f[3][1]-1

手算样例

f[p][0]	表示尾部共p位数待填写时不包含4的方案数	无其他限制
f[p][1]	表示尾部共p位数待填写时不包含4的方案数	不能超过x尾部p位数

举例：x=666

如何算	f[2][1]	51

2号位置(十位)放0	f[1][0]	9
2号位置(十位)放1	f[1][0]	9
2号位置(十位)放2	f[1][0]	9
2号位置(十位)放3	f[1][0]	9
2号位置(十位)放4	0	0
2号位置(十位)放5	f[1][0]	9
2号位置(十位)放6	f[1][1]	6

手算样例

f[p][0]	表示尾部共p位数待填写时不包含4的方案数	无其他限制
f[p][1]	表示尾部共p位数待填写时不包含4的方案数	不能超过x尾部p位数

举例：x=666

如何算	f[3][1]	456

3号位置(百位)放0	f[2][0]	81
3号位置(百位)放1	f[2][0]	81
3号位置(百位)放2	f[2][0]	81
3号位置(百位)放3	f[2][0]	81
3号位置(百位)放4	0	0
3号位置(百位)放5	f[2][0]	81
3号位置(百位)放6	f[2][1]	51

ans=f[3][1]-1

数位DP
对逐个数字位置依次决策

记忆化搜索
递归实现动态规划

记忆化搜索

f[p][0]	表示尾部共p位数待填写时不包含4的方案数	无其他限制
f[p][1]	表示尾部共p位数待填写时不包含4的方案数	不能超过x尾部p位数

统一状态描述	引入01状态lmt: limit(限制)的缩写

f[p][lmt] 答案存于数组	表示尾部共p位数待填写时不包含4的方案数 lmt是否为1表示填写后的数字能否超过x尾部p位数

F(p,lmt) 函数计算过程	表示尾部共p位数待填写时不包含4的方案数 lmt是否为1表示填写后的数字能否超过x尾部p位数

ok[p][lmt] 判断记忆	表示f[p][lmt]是否计算过

已知x，求1到x里共几个数不包含4

对输入的x进行数位分离后
用数组储存：d[nD],…,d[2],d[1]

```
6  int F(int p,bool lmt){
7      if(p==0) return 1;
8      if(ok[p][lmt]) return 
9      ok[p][lmt]=1;
10     for(int i=0;i<=(lmt?d[p]:9);++i){
11         if(i==4)
12         f[p][lmt]+=F(p-1,lmt&&i==d[p]);
13     }
14     return 
15 }
```

枚举p号位置放几

枚举上限由lmt定

是否取到上限

已知x，求1到x里共几个数不包含4
对输入的x进行数位分离后 用数组储存：d[nD],…,d[2],d[1]

```
19      int x;
20      cin>>x;
21      int nD=0;
22      do{
23          d[++nD]=x%10;
24          x/=10;
25      }while(x);
26      cout<<          <<endl;
```

易错点	F()和f[]通常将0的情况也统计在内

算法讨论

两种枚举思路的对比

1

枚举范围内所有数字

判断是否合法

2

枚举每一位

再对每一位枚举合法的数字

此方法包含重叠子问题

快快编程2551

已知n和m，求n到m里共几个数不包含4和13

区间计数问题 → 前缀计数问题

答案 = solve(m) – solve(n-1)

solve(x)计算从0到x内共有几个数满足要求

取x为m，再取x为n-1

不是从1开始

对参数x进行数位分离后
用数组储存：d[nD],…,d[2],d[1]

同一个函数被调用多次需要注意什么易错点?

相关变量和数组需要重新初始化：ok,f

状态定义

solve(x)计算从0到x内共有几个数不包含13和4

请同学尝试设计状态	每位可以填0到9

f[p][lmt][pre1]	表示尾部共p位数待填写时不含13和4的方案数
	额外状态要求用01变量lmt和pre1表示

lmt为0	表示可以超过x尾部p位数
lmt为1	表示不能超过x尾部p位数

pre1为0	表示前1位(从右数第p+1位)不是1
pre1为1	表示前1位(从右数第p+1位)放了1

x

6	6	6

填写情况

6	?	?

描述状态
3个参数

p=2
lmt=1
pre1=0

```
/*
f[p][lmt][pre1]:尾部p位数不含13和4的方案数
lmt为1: 不能超过x尾部p位数
pre1为1: 前1位(从右数第p+1位)放了1
*/
int F(int p,bool lmt,bool pre1){
    if(p==0) return 1;
    if(ok[p][lmt][pre1]) return f[p][lmt][pre1];
    ok[p][lmt][pre1]=1;
    f[p][lmt][pre1]=0;
    for(int i=0;                    ;++i){
        if(i==4)continue;
        if(                    )continue;
        f[p][lmt][pre1]+=                                    
    }
    return f[p][lmt][pre1];
}
```

solve(x)计算从0到x内共有几个数不包含13和4

```
23 int solve(int x){
24     
25     
26     int nD=0;
27     do{
28         d[++nD]=x%10;
29         x/=10;
30     }while(x);
31     return 
32 }
```

易错点	数位DP多次调用前缀计数函数时需要数组清零

快快编程2552

已知a和b，求a到b里共几个数是波动数

区间计数问题 → 前缀计数问题

答案 = solve(b) – solve(a-1)

solve(x)计算从0到x内共有几个数是波动数

取x为b，再取x为a-1

不是从1开始

对参数x进行数位分离后
用数组储存：d[nD-1],…,d[1],d[0]

同一个函数被调用多次需要注意什么易错点?

相关变量和数组需要重新初始化：ok,f

状态定义

solve(x)计算从0到x内共有几个数是波动数

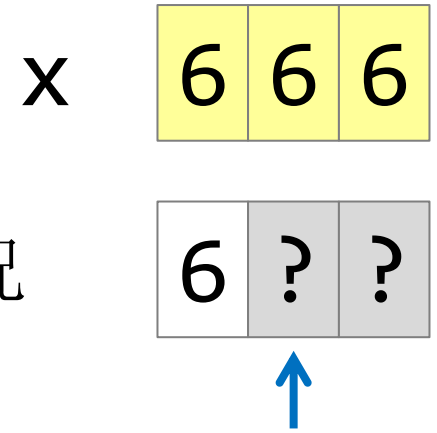
请同学尝试设计状态	每位可以填0到9

f[p][pre][lmt][lead0]	表示前一位已填pre时尾部p位数有几个波动数
	额外状态要求用01变量lmt和lead0表示

pre可以取0到9

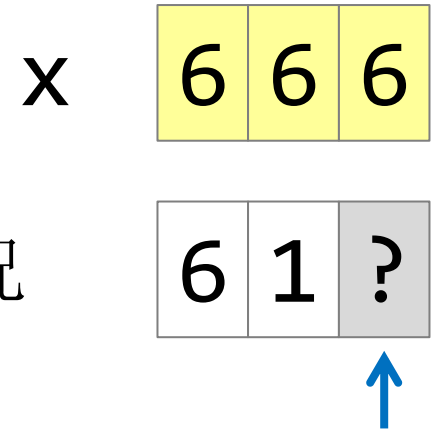
lmt为0	表示可以超过x尾部p位数
lmt为1	表示不能超过x尾部p位数

lead0为0	之前数位并不都是填写0
lead0为1	之前数位都是填写0



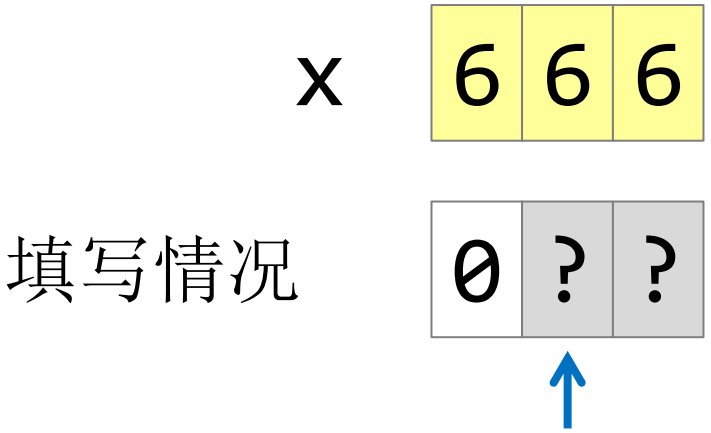
描述状态
4个参数

p=2
pre=6
lmt=1
lead0=0



描述状态
4个参数

p=1
pre=1
lmt=0
lead0=0



描述状态
4个参数

p=2
pre=0
lmt=0
lead0=1

```
/*
f[p][pre][lmt][lead0]:前一位已填pre时尾部p位数有几个波动数
pre: 取0到9
lmt为1: 不能超过x尾部p位数
lead0为1: 之前数位都是填写0
*/
int F(int p,int pre,bool lmt,bool lead0){
    if(p==0) return 1;
    if(ok[p][pre][lmt][lead0]) return f[p][pre][lmt][lead0];
    ok[p][pre][lmt][lead0]=1;
    f[p][pre][lmt][lead0]=0;
    for(int i=0;                    ;++i){
        if(                         )continue;
        f[p][pre][lmt][lead0]+=                                   ;
    }
    return f[p][pre][lmt][lead0];
}
```

solve(x)计算从0到x内共有几个数是波动数

```
23 int solve(int x){
24     [          ]
25     [          ]
26     int nD=0;
27     while(x){
28         d[++nD]=x%10;
29         x/=10;
30     }
31     return [          ]
32 }
```

易错点	数位DP多次调用前缀计数函数时需要数组清零

太戈编程

2550
2551
2552

拓展题	1373,1374,1375