

太戈编程
etiger.vip

信奥算法

1374

正解

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

```
58 void solve(){  
59     cout<<prefix(r)-prefix(l-1)<<endl;  
60 }
```

请写出prefix(r)含义

| | |
|-----------------------------|--|
| f[p][pre1][pre2][lead][lmt] | |
| 当前状态共有几个非回文数 | |

| | |
|---|----------|
| p | 尾部p位数待填写 |
|---|----------|

| | |
|------|------------------------|
| pre1 | 当前位的左侧第一位已填写pre1（可能缺省） |
| pre2 | 当前位的左侧第二位已填写pre2（可能缺省） |

| | |
|------|--|
| lead | 当前位的左侧已填写的有效位数 0 表示左侧未填写非零数字 1 表示左侧恰填写1个非零数字 2 表示左侧已填写不少于2个有效数字 |
|------|--|

| | |
|-----|---|
| lmt | 尾部p位数大小是否受限制 0 可以无限制随意填写 1 不能超过d[]尾部p位数 |
|-----|---|

x

| | | | |
|---|---|---|---|
| 6 | 6 | 6 | 6 |
|---|---|---|---|

nD=4

已填写情况

| | | | |
|---|---|---|---|
| 0 | 6 | ? | ? |
|---|---|---|---|



当前位
待填写

描述状态

| | | |
|------|---|----|
| p | 2 | |
| pre1 | 6 | |
| pre2 | 0 | 缺省 |
| lead | 1 | |
| lmt | 0 | |

```

27  /*
28  f[p][pre1][pre2][lead][lmt]
29  当前状态共有几个非回文数
30  p: 尾部p位数待填写
31  pre1: 当前位的左侧第一位已填写pre1 (可能缺省)
32  pre2: 当前位的左侧第二位已填写pre2 (可能缺省)
33  lead: 当前位的左侧已填写的有效位数
34  0 表示左侧未填写非零数字
35  1 表示左侧恰填写1个非零数字
36  2 表示左侧已填写不少于2个有效数字
37  lmt: 尾部p位数大小是否受限制
38  0 可以无限制随意填写
39  1 不能超过d[]尾部p位数
40  */
41  ll f[20][10][10][3][2];
42  bool ok[20][10][10][3][2];
43  ll F(ll p, ll pre1, ll pre2, ll lead, bool lmt){

```

请同学完成27-43行
给老师查看

```
34 11 F(11 p,11 pre1,11 pre2,11 lead,bool lmt){
35     if(p==0) return 
36     if(ok[p][pre1][pre2][lead][lmt])
37         return f[p][pre1][pre2][lead][lmt];
38     ok[p][pre1][pre2][lead][lmt]=1;
39     f[p][pre1][pre2][lead][lmt]=0;
40     for() {
41         if(lead>=1&&i==pre1) continue;
42         
43         11 L=(i||lead ? lead+1 : 0);
44         if(L>2) L=2;
45         f[p][pre1][pre2][lead][lmt]
46             += 
47     }
48     return f[p][pre1][pre2][lead][lmt];
49 }
```


讨论

```
memset(ok,0,sizeof(ok));  
memset(f,0,sizeof(f));
```

这两行能否省略?

1373

正解

数位DP

对逐个数字位置依次决策

难点

随着填数过程
数位总和在变化
自整除倍数关系较难维护

破解难点

枚举：数位总和MOD
作为除数，维护余数r

```

48 ll prefix(ll x){
49     ll nD=0;
50     do{
51         d[++nD]=x%10;
52         x/=10;
53     }while(x);
54     ll res=0;
55     for(ll MOD=1;MOD<=nD*9;++MOD){
56         res+=F(MOD,nD,MOD,0,1);
57     }
58     return res;
59 }
60 void solve(){
61     cout<<prefix(r)-prefix(l-1)<<endl;
62 }

```

memset放哪里?



```
24  /*
25  MOD为除数, 即所有nD位数位的总和
26  f[p][s][r][lmt]表示当前状态共几个自整除数
27  p:尾部p位数待填写
28  s:尾部p位数的待分配的总和
29  r:前nD-p位数组成的数模MOD的余数
30  lmt:尾部p位数大小是否受限制
31  0  可以无限制随意填写
32  1  不能超过d[]尾部p位数
33  */
34  ll f[20][165][165][2];
35  bool ok[20][165][165][2];
36  ll F(ll&MOD, ll p, ll s, ll r, bool lmt){
```

请同学完成24-36行
给老师查看

```

36 11 F(11&MOD, 11 p, 11 s, 11 r, bool lmt){
37     if(p==0) return 1
38     if(ok[p][s][r][lmt])
39         return f[p][s][r][lmt];
40     ok[p][s][r][lmt]=1;
41     f[p][s][r][lmt]=0;
42     for(11 i=0; i<=(lmt?d[p]:9); ++i){
43         3
44         4
45         f[p][s][r][lmt] +=
46         2
47     }
48     return f[p][s][r][lmt];
49 }

```

可行性剪枝

删除剪枝是否
保证正确性

1375

暴力



数位分离

正解

数位DP


```
69 int main(){
70     freopen("statistics.in","r",stdin);
71     freopen("statistics.out","w",stdout);
72     input();
73     if(r-l<=100000)
74         solveBF();
75     else
76         solve();
77     return 0;
78 }
```

```
16 void solveBF(){
17     for(ll T=0;T<=9;++T){
18         ll ans=0;
19         for(ll x=1;x<=r;++x)
20             ans+=count(T,x);
21         cout<<ans<<" ";
22     }
23     cout<<endl;
24 }
```

```
8 ll count(ll T, ll x){  
9     ll res=0;  
10    do{  
11          
12          
13    }while(x);  
14    return res;  
15 }
```

正解

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

0到9的处理是否有区别?

```
64 void solve(){
65     for(ll T=0;T<=9;++T)
66         cout<<prefix(T,r)-prefix(T,l-1)<<" ";
67     cout<<endl;
68 }
```

请同学设计状态

| | |
|---------------------------|--|
| f[T][p][lmt][lead0] | |
| 当前状态的尾部p位数有几个目标字T | |
| p | 尾部p位数待填写 |
| lmt | 尾部p位数大小是否受限制 0 可以无限制随意填写 1 不能超过d[]尾部p位数 |
| lead0 | 当前位左侧是否全部是先导0 0 表示左侧并不全是先导0 1 表示左侧并全是先导0 |
| cnt[p][lmt] | |
| 当前状态尾部p位数共有几个非负整数 | |
| cnt[p][0]为 10^p | |
| cnt[p][1]为d[]尾部p位数组成的数值+1 | |

```
47 11 prefix(11 T, 11 x){
48     memset(ok, 0, sizeof(ok));
49     memset(f, 0, sizeof(f));
50     11 y=x;
51     11 nD=0;
52     do{
53         d[++nD]=x%10;
54         x/=10;
55     }while(x);
56     
57     
58     for(11 p=1; p<=nD; ++p){
59         cnt[p][0]= 
60         cnt[p][1]= 
61     }
62     return 
63 }
```

```
35 11 F(11 T, 11 p, bool lmt, bool lead0){
36     if(p==0) return 
37     if(ok[T][p][lmt][lead0])
38         return f[T][p][lmt][lead0];
39     ok[T][p][lmt][lead0]=1;
40     f[T][p][lmt][lead0]=0;
41     for(11 i=0; i<=(lmt?d[p]:9); ++i){
42         if() 0的特殊处理
43             f[T][p][lmt][lead0]+=cnt[p-1][lmt&& i==d[p]];
44         f[T][p][lmt][lead0]
45             += 
46     }
47     return f[T][p][lmt][lead0];
48 }
```




太戈编程

1374,1373,1375

要求

搭配暴力+对拍