个人资料

niuox

访问： 241490次
积分： 4670
等级： BLOG 5
排名： 第2487名

原创： 239篇 转载： 48篇
译文： 1篇 评论： 43条

数位DP问题整理（一）

分类： Algorithm_动规 2013-08-13 23:11 2595人阅读 评论(0) 收藏 举报

第一题：Amount of degrees (ural 1057)

题目链接：http://acm.timus.ru/problem.aspx?space=1&num=1057

题意：[x,y]范围内的数，可以拆分成k个b进制的不同幂的和 的数字有多少。

我们可以将x转换成二进制来讨论。二进制转化时，找到第一个非0非1的数，将其及其后面的数都变为1.

那么问题就变成了求[0,x]范围内，二进制表示中含有k个1的数字有多少个。

求[x,y]区间相减。我们可以给数建立0，1的表示树。

在求高度为i的完全二叉树中含有j个1的路径有多少个时，递推式为：f[i,j] = f[i-1,j-1] + f[i-1,j]

```cpp
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <map>
#include <queue>
#include <algorithm>
using namespace std;

int f[35][35];
int d[35];
//高度为i(i>=0)时，含有j个1的个数
void init()
{
   memset(f,0,sizeof(f));
   f[0][0] = 1;
   for(int i=1;i<=31;i++)
   {
      f[i][0] = 1;
      for(int j=1;j<=i;j++)
      {
         f[i][j] = f[i-1][j-1] + f[i-1][j];
      }
   }
}
//[0,x]范围内二进制含有k个1的个数
int calc(int x,int k)
{
   //路径上含有的1的个数
   int tot = 0;
   int ans = 0;
   for(int i=31;i>0;i--)
   {
      if(x&(1<<i))
      {
         tot++;
         if(tot>k) break;
         x ^= (1<<i);
      }
```

Algorithm_排序 (9)
Algorithm_贪心 (6)
Algorithm_计算几何 (15)
MultiCore Programming (2)
Algorithm_三分 (1)
Algorithm_math (10)
Algorithm_枚举 (3)
Algorithm_图论 (25)
Algorithm_数据结构 (14)
Algorithm_并查集 (4)
Algorithm_字符串 (9)
Vijos (4)
Algorithm_Open Problem (1)
OpenGl (1)
Algorithm_网络流 (2)
Algorithm_Hash (3)
Algorithm_Treap (1)
Daily-Life (1)
SQA (1)
Algorithm_强连通 (5)
Algorithm_二分图 (1)
Algorithm_2-Sat (2)
Algorithm_差分约束系统 (2)
Git/Github版本控制 (0)
Shell编程 (1)
自然语言处理 (2)
Python (3)
操作系统 (1)
设计模式 (2)

友情链接

byvoid
cplusplus
Matrix67

文章存档

2014年05月 (1)
2013年10月 (1)
2013年09月 (11)
2013年08月 (26)
2013年07月 (3)

展开

```
41.       if((1<<(i-1))<=x) ans += f[i-1][k-tot];
42.    }
43.    if(tot + x == k) ans++;
44.    return ans;
45. }
46. //b进制转化为二进制
47. int transfer(int b,int x)
48. {
49.    int m = 0;
50.    int ans = 0;
51.    while(x)
52.    {
53.       d[m++] = x % b;
54.       x/=b;
55.    }
56.    for(int i=m-1;i>=0;i--)
57.    {
58.       if(d[i]>1)
59.       {
60.          for(int j=i;j>=0;j--) ans |= (1<<j);
61.       }
62.       else ans |= d[i]<<i;
63.    }
64.    return ans;
65. }
66. int main()
67. {
68.    #ifndef ONLINE_JUDGE
69.       freopen("in.txt","r",stdin);
70.    #endif
71.    int x,y;
72.    int k,b;
73.    init();
74.    while(scanf(" %d %d",&x,&y)!=EOF)
75.    {
76.       scanf(" %d %d",&k,&b);
77.       x = transfer(b,x-1);
78.       y = transfer(b,y);
79.       printf("%d\n",calc(y,k) - calc(x,k));
80.    }
81.    return 0;
82. }
```

第二题：windy数。

题意：求给定区间范围内的，求相邻数位之差绝对值不小于2的数的个数。

题目链接：http://www.lydsy.com/JudgeOnline/problem.php?id=1026

```
[cpp]
01. #include <iostream>
02. #include <stdio.h>
03. #include <stdlib.h>
04. #include <string.h>
05. #include <math.h>
06. #include <map>
07. #include <queue>
08. #include <algorithm>
09. using namespace std;
10.
11. int A[12];
12.
13. int f[12][10];
14.
15. //f[i][j]代表长度为i,最高位为j的windy数个数
16. void init()
17. {
18.    memset(f,0,sizeof(f));
19.    for(int i=0;i<10;i++) f[1][i] = 1;
20.    for(int i=2;i<=10;i++)
21.    {
22.       for(int j=0;j<10;j++)
23.       {
24.          for(int k=0;k<10;k++)
25.          {
26.             if(abs(j-k)>1) f[i][j] += f[i-1][k];
27.          }
```

```
        }
    }
}
//(0,a)范围内的windy数个数
int calc(int a)
{
    int m = 0;
    while(a)
    {
        A[m++] = a%10;
        a/=10;
    }
    int ans = 0;
    //先处理长度小于m的windy数的个数
    for(int i=1;i<m;i++)
    {
        //题目要求不含前导0
        for(int j=1;j<10;j++)
        {
            ans += f[i][j];
        }
    }
    //长度等于m且最高位和原数不同且小于原数的windy数
    for(int j=1;j<A[m-1];j++) ans += f[m][j];
    //依次循环将最高位 变为和原数相同
    for(int i=m-1;i>=1;i--)
    {
        for(int j=0;j<A[i-1];j++)
        {
            if(abs(j-A[i]) > 1) ans += f[i][j];
        }
        if(abs(A[i] - A[i-1])<=1) break;
    }
    return ans;
}


int main()
{
    #ifndef ONLINE_JUDGE
        freopen("in.txt","r",stdin);
    #endif
    int a,b;
    init();
    while(scanf(" %d %d",&a,&b)!=EOF)
    {
        int ans = calc(b+1) - calc(a);
        printf("%d\n",ans );
    }
    return 0;
}
```

第三题：Hdu 2089 不要62

题目连接：http://acm.hdu.edu.cn/showproblem.php?pid=2089

求给定区间中不含有62和4的数的个数。

```
[cpp]
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <map>
#include <queue>
#include <algorithm>
using namespace std;

int dp[10][3];

int A[10];

//(0,a]范围内有多少个吉利数
int calc(int a)
```

```
{
   int sum = a;
   int m = 0;
   int ans = 0;
   bool flag = false;
   while(a)
   {
      A[++m] = a%10;
      a/=10;
   }
   A[m+1] = 0;
   for(int i=m;i>=1;i--)
   {
      ans += dp[i-1][2] * A[i];
      if(flag)
      {
         ans += dp[i-1][0] * A[i];
      }
      else
      {
         if(A[i]>4) ans += dp[i-1][0];
         if(A[i+1] == 6 && A[i]>2) ans += dp[i][1];
         if(A[i]>6) ans += dp[i-1][1];
         if(A[i] == 4 || (A[i+1] == 6 && A[i] == 2)) flag = true;
      }
   }
   //数本身
   if(flag) ans++;
   return sum - ans;
}

//dp[i][0]:长度<=i的吉利数个数
//dp[i][1]:长度为i,且最高位含有2的吉利数个数
//dp[i][2]:长度<=i的非吉利数个数
void init()
{
   memset(dp,0,sizeof(dp));
   dp[0][0] = 1;
   for(int i=1;i<=8;i++)
   {
      dp[i][0] = dp[i-1][0]*9 - dp[i-1][1];
      dp[i][1] = dp[i-1][0];
      dp[i][2] = dp[i-1][0] + dp[i-1][1] + dp[i-1][2] * 10;
   }
}

int main()
{
   #ifndef ONLINE_JUDGE
      freopen("in.txt","r",stdin);
   #endif
   int a,b;
   init();
   while(scanf(" %d %d",&a,&b)!=EOF)
   {
      if(a == 0 && b == 0) break;
      int ans = calc(b) - calc(a-1);
      printf("%d\n",ans);
   }
   return 0;
}
```

第四题：Hdu 3555 Bomb

题目连接：http://acm.hdu.edu.cn/showproblem.php?pid=3555

题意：求给定区间的含有49的数的个数。

方法与上题类似，比上题要简单许多。

```
[cpp]

#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```cpp
#include <math.h>
#include <map>
#include <queue>
#include <algorithm>
using namespace std;

#define LL __int64
LL dp[25][3];

int A[25];

//(0,a]范围内有多少个吉利数
LL calc(LL a)
{
   int m = 0;
   LL ans = 0;
   bool flag = false;
   while(a)
   {
      A[++m] = a%10;
      a/=10;
   }
   A[m+1] = 0;
   for(int i=m;i>=1;i--)
   {
      ans += dp[i-1][2] * A[i];
      if(flag)
      {
         ans += dp[i-1][0] * A[i];
      }
      else
      {
         if(A[i]>4) ans += dp[i-1][1];
         if(A[i+1] == 4 && A[i] == 9) flag = true;
      }
   }
   //数本身
   if(flag) ans++;
   return ans;
}

//dp[i][0]:长度<=i的不含49的数的个数
//dp[i][1]:长度为i,且最高位含有9的不含49的数的个数
//dp[i][2]:长度<=i的含有49的数个数
void init()
{
   memset(dp,0,sizeof(dp));
   dp[0][0] = 1;
   for(int i=1;i<=22;i++)
   {
      dp[i][0] = dp[i-1][0]*10 - dp[i-1][1];
      dp[i][1] = dp[i-1][0];
      dp[i][2] = dp[i-1][2] * 10 + dp[i-1][1];
   }
}

int main()
{
   #ifndef ONLINE_JUDGE
      freopen("in.txt","r",stdin);
   #endif
   int t;
   LL a;
   init();
   scanf(" %d",&t);
   while(t--)
   {
      scanf(" %I64d",&a);
      LL ans = calc(a);
      printf("%I64d\n", ans);
   }
   return 0;
}
```

第五题：Hdu 3709 Balanced Number

题目链接：http://acm.hdu.edu.cn/showproblem.php?pid=3709

平衡数。枚举平衡位置。采用记忆化搜索的方式记录已有的值。加适当剪枝。然后排除掉重复的0即可。

[cpp]

```cpp
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <map>
#include <queue>
#include <algorithm>
using namespace std;

#define LL long long
#define Maxn 20

LL dp[Maxn][Maxn][2005];
int digit[Maxn];

LL dfs(int pos,int pivot,int pre,bool limit)
{
   if(pos<=0) return pre == 0;
   if(pre<0) return 0;
   if(!limit && dp[pos][pivot][pre]!=-1) return dp[pos][pivot][pre];
   int end = limit ? digit[pos] : 9;
   LL ans = 0;
   for(int i=0;i<=end;i++)
   {
      ans += dfs(pos-1,pivot,pre + i*(pos-pivot),limit && (i == end));
   }
   if(!limit) dp[pos][pivot][pre] = ans;
   return ans;
}

LL calc(LL a)
{
   if(a<0) return 0;
   int len = 0;
   LL ans = 0;
   while(a>0)
   {
      digit[++len] = a%10;
      a/=10;
   }
   for(int i=1;i<=len;i++)
   {
      ans += dfs(len,i,0,1);
   }
   ans = ans - len + 1;
   return ans;
}
int main()
{
   #ifndef ONLINE_JUDGE
      freopen("in.txt","r",stdin);
   #endif
   int t;
   LL x,y;
   scanf(" %d",&t);
   memset(dp,-1,sizeof(dp));
   while(t--)
   {

      scanf(" %I64d %I64d",&x,&y);
      printf("%I64d\n",calc(y) - calc(x-1) );
   }
   return 0;
}
```

第六题：Hoj 1983 Beautiful numbers

题目链接：http://acm.hit.edu.cn/hoj/problem/view?id=1983

题意：如果一个数能够被其每个数位的数都整除，那么这个数就叫做美丽数。

基本思路是用：dp[len][mod][lcm]表示<=len的长度中，此数为mod,各数位的最小公倍数为lcm的数的个数来进行记忆化搜索。方法和上一题类似。

但我们发现，len在[1,20]范围内，mod在[1,1^18]范围内，lcm在[1,2520]范围内。所以dp数组肯定超内存。

下面我们来进行内存优化：

假设这个数为a,各个数位的值分别为ai,那么我们发现lcm(ai) | a.

而[1,9]的最小公倍数是2520.那么lcm(ai) | 2520, 所以lcm(ai) | (a%2520).

所以第二维大小我们可以从1^18降到2520,方法是%2520.

现在的dp数组的内存是20*2520*2520，还是很大。

然后我们再考虑：

我们发现某一个数的各个数位的数的最小公倍数最大是2520，而且只能是2520的公约数。而2520的公约数有48个。所以第三维我们只用[50]的空间就行了。

方法是用Hash进行离散化。‘

这样内存就成了20*2520*50,可以拿下这道题目了。

```cpp
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <map>
#include <queue>
#include <algorithm>
using namespace std;

#define LL long long

LL dp[20][2525][55];
int digit[20];
int hash[2525];

int gcd(int a,int b)
{
   if(b == 0) return a;
   return gcd(b,a%b);
}
int calc_lcm(int a,int b)
{
   return a/gcd(a,b)*b;
}
LL dfs(int pos,int mod,int lcm,bool limit)
{
   LL ans = 0;
   if(pos<=0) return mod % lcm == 0;
   if(!limit && dp[pos][mod][hash[lcm]]!=-1) return dp[pos][mod][hash[lcm]];
   int end = limit ? digit[pos] : 9;
   for(int i=0;i<=end;i++)
   {
      ans += dfs(pos-1,(mod*10+i)%2520,i?calc_lcm(lcm,i):lcm,limit && (i==end));
   }
   if(!limit) dp[pos][mod][hash[lcm]] = ans;
   return ans;
}

LL calc(LL a)
{
   if(a<0) return 0;
   int len = 0;
   while(a>0)
   {
      digit[++len] = a%10;
      a/=10;
```

```
    }
    //0也当作其中的一个美丽数，因为两者相减会抵消掉
    LL ans = dfs(len,0,1,1);
    return ans;
}
void init()
{
    memset(dp,-1,sizeof(dp));
    int id = 0;
    for(int i=1;i*i<=2520;i++)
    {
        if(2520%i == 0)
        {
            hash[i] = id++;
            if(i*i!=2520) hash[2520/i] = id++;
        }
    }
    //printf("id = %d\n", id);
}
int main()
{
    #ifndef ONLINE_JUDGE
        freopen("in.txt","r",stdin);
    #endif
    init();
    int t;
    LL x,y;
    while(scanf(" %lld %lld",&x,&y)!=EOF)
    {
        printf("%lld\n",calc(y) - calc(x-1));
    }
    return 0;
}
```

第七题：吉哥系列故事——恨7不成妻

题目链接：http://acm.hdu.edu.cn/showproblem.php?pid=4507

与上一题做法也类似，只不过dp需要保存三种值，所以把它结构体了

```
[cpp]
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <map>
#include <queue>
#include <algorithm>
using namespace std;

#define MOD 1000000007
#define LL __int64

int digit[20];
LL power[20];

struct  Node
{
    LL n,s,sq;
}dp[20][10][10];

Node dfs(int pos,int mod,int modSum,bool limit)
{
    if(pos<=0)
    {
        Node t;
        t.n = (mod!=0 && modSum!=0);
        t.s  = t.sq = 0;
        return t;
    }
    if(!limit && dp[pos][mod][modSum].n!=-1) return dp[pos][mod][modSum];
    int end = limit ? digit[pos] : 9;
    Node ans,temp;
    ans.n = ans.s = ans.sq = 0;
    for(int i=0;i<=end;i++)
    {
```

```
        if(i == 7) continue;
        temp = dfs(pos-1,(mod*10+i)%7,(modSum+i)%7,limit && (i == end));
        ans.n = (ans.n + temp.n)%MOD;
        ans.s = (ans.s + temp.s + ((i * power[pos])%MOD * temp.n) % MOD) % MOD ;
        ans.sq = (ans.sq + temp.sq + ((2*i*power[pos])%MOD*temp.s)%MOD + (((i*i*power[pos])%MOD*powe
    }
    if(!limit) dp[pos][mod][modSum] = ans;
    return ans;
}
LL calc(LL a)
{
    int len = 0;
    while(a>0)
    {
        digit[++len] = a%10;
        a/=10;
    }
    Node ans = dfs(len,0,0,true);
    return ans.sq;
}
void init()
{
    memset(dp,-1,sizeof(dp));
    memset(power,0,sizeof(power));
    power[1] = 1;
    for(int i=2;i<=19;i++)
    {
        power[i] = (power[i-1] * 10)%MOD;
    }
}
int main()
{
    #ifndef ONLINE_JUDGE
        freopen("in.txt","r",stdin);
    #endif
    int t;
    LL l,r;
    init();
    scanf(" %d",&t);
    while(t--)
    {
        scanf(" %I64d %I64d",&l,&r);
        LL ans = (calc(r) - calc(l-1) + MOD)%MOD;
        printf("%lld\n", ans);
    }
    return 0;
}
```


上一篇 Hoj 3040 Team Mate

下一篇 概率DP问题整理（一）

hdu 2089 不要62和4

ZOJ - 3180 Number Game

zoj 3818 Pretty Poem暴力处理字符串2014年牡丹江赛

2012年长春赛区小结

拓扑排序

【精品课程】备战2015信息系统项目管理师-案例梳理与

【精品课程】华为IE讲师：直通华为HCNA课程实战第一

【精品课程】通过经典案例教你学Javascript

【精品课程】2015软考网络工程师--基础知识视频教程

【精品课程】Android入门实战教程

查看评论

暂无评论

发表评论

用 户 名： MacinChang

评论内容：

提交

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide Maemo Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP HBase Pure Solr Angular Cloud Foundry Redis Scala Django Bootstrap