这是一道很好的数位dp的题目

题意：数字满足的条件是该数字可以被它的每一位非零位整除。

思路：其实我数位dp刚刚入门（其实还没入门，刚开始懂），发现其实数位dp只要理解了大部分的套路部分，核心部分就是要找到判断这个数满足条件的方法，如果找到了判断数字满足题目条件的方法那么这个题目就解决了。

这个题的条件是数字被每一位非零数整除，那么可以到的，这个数应该被每一位的最小公倍数整除，1-9的最小公倍数是2520，所以其他位数最小公倍数都在2520内，即比2520小，所以这样思路就很明显了

dfs（pos, num,lcm,limit）,分别代表第几位，当前数字，当前数字所有非零位的最小公倍数，是否有限制。当前数字因为太大，而我们最终枚举完后只需要判断

num % lcm是否为0，所以每次传入的num每次模2520即可，这样就可以小很多，所以我们的dp数组可以开三维dp[pos][num][lcm]，分别是第几位，数字取模后的大小，最小公倍数，这样我们至少要开19x2520x2520的大小，肯定开不了，所以对于lcm这一维我们需要进行离散化，经过打表可以发现，2520内可以整除2520的只有48个，所以我们可以离散化一下让lcm映射到1-48既可以了这样就可以开19x2520x48大小的了。

#include <iostream>

#include <cstdio>

#include <cstring>

#include <cmath>

#include <algorithm>

using namespace std;

#define mod 2520

#define ll long long

int Hash[3000];

ll dp[30][3000][100];

int a[30];

inline void init()

{

int cnt=0;

for(int i=1;i<=mod;i++)

if(mod%i==0)

Hash[i]=cnt++;

}

ll dfs(int pos,int num,int lcm,int limit)

//dp[pos][num][Hash[lcm]]

{

if(pos==-1)

return num%lcm==0;

if(limit==0 && dp[pos][num][Hash[lcm]]!=-1)

return dp[pos][num][Hash[lcm]];

int up=(limit==1?a[pos]:9);

ll ans=0;

for(int i=0;i<=up;i++)

{

ans+=dfs(pos-1,(num\*10+i)%mod,(i==0?lcm:i\*lcm/\_\_gcd(i,lcm)),limit&&i==up);

}

if(limit==0)

dp[pos][num][Hash[lcm]]=ans;

return ans;

}

inline ll solve(ll x)

{

int pos=0;

while(x)

{

a[pos++]=x%10;

x/=10;

}

return dfs(pos-1,0,1,1);

}

int main()

{

//freopen("input.txt","r",stdin);

int T;

cin>>T;

memset(dp,-1,sizeof(dp));

init();

ll l,r;

while(T--)

{

cin>>l>>r;

cout<<solve(r)-solve(l-1)<<endl;

}

return 0;

}