E. Salazar Slytherin's Locket

time limit per test 2 seconds
memory limit per test 256 megabytes
input standard input
output standard output

Harry came to know from Dumbledore that Salazar Slytherin's locket is a horcrux. This locket was present earlier at 12 Grimmauld Place, the home of Sirius Black's mother. It was stolen from there and is now present in the Ministry of Magic in the office of Dolorous Umbridge, Harry's former Defense Against the Dark Arts teacher.

Harry, Ron and Hermione are infiltrating the Ministry. Upon reaching Umbridge's office, they observed a code lock with a puzzle asking them to calculate count of magic numbers between two integers l and r (both inclusive).

Harry remembered from his detention time with Umbridge that she defined a magic number as a number which when converted to a given base b, all the digits from 0 to b - 1 appear even number of times in its representation without any leading zeros.

You have to answer q queries to unlock the office. Each query has three integers b_i, l_i and r_i, the base and the range for which you have to find the count of magic numbers.

Input

First line of input contains q $(1 \leq q \leq 10^5)$ — number of queries.

Each of the next q lines contain three space separated integers b_i, l_i, r_i $(2 \leq b_i \leq 10, 1 \leq l_i \leq r_i \leq 10^{18})$.

Output

You have to output q lines, each containing a single integer, the answer to the corresponding query.

Examples

input
```
2
2 4 9
3 1 10
```

output
```
1
2
```

input
```
2
2 1 100
5 1 100
```

output
```
21
4
```

Note

In sample test case 1, for first query, when we convert numbers 4 to 9 into base 2, we get:

- $4 = 100_2$,
- $5 = 101_2$,
- $6 = 110_2$,
- $7 = 111_2$,
- $8 = 1000_2$,
- $9 = 1001_2$.

Out of these, only base 2 representation of 9 has even number of 1 and 0. Thus, the answer is 1.

```cpp
1   /// http://codeforces.com/contest/855/problem/E
2   #include<bits/stdc++.h>
3   using namespace std;
4   #define ll long long
5   ll dp[11][65][1030][2];
6   int a[] = {0,0,1,2,3,4,5,6,7,8,9};
7   int SIZE[] = {0,0,62,41,32,28,26,24,22,21,20};
8   string s;
9   ll fun(int b,int p,int m,int f){
10      if(p>SIZE[b]){
11          if(f==0)return 0LL;
12          if(m==0)return 1LL;
13          else return 0LL;
14      }
15
16      if(dp[b][p][m][f] != -1)return dp[b][p][m][f];
17
18      ll ret = 0;
19      for(int i=0; i<=a[b]; i++){
20          if(i==0 && f==0) ret += fun(b,p+1,m,f);
21          else ret += fun(b,p+1,m^(1<<i),f|(i>0));
22      }
23      return dp[b][p][m][f] = ret;
24  }
25  ll go(int b,int p,int m,int f,int c){
26      if(p>SIZE[b]){
27          if(f && m==0)return 1LL;
28          else return 0LL;
29      }
30
31      if(c)return dp[b][p][m][f];
32
33      int lo = 0, hi = s[p]-'0';
34      ll ret = 0;
35      for(int i=lo; i<=hi; i++){
36          if(i==0 && f==0) ret += go(b,p+1,m,f,c|(i<hi));
37          else ret += go(b,p+1,m^(1<<i),f|(i>0),c|(i<hi));
38      }
39      return ret;
40  }
41  string getString(ll b,ll n){
42      string ss;
43      while(n!=0){
44          int r = n%b;
45          ss += r+'0';
46          n /= b;
47      }
48      int p = SIZE[b]-(int)ss.size();
49      for(int i=0; i<=p; i++) ss+='0';
50      reverse(ss.begin(),ss.end());
51      return ss;
52  }
53  ll solve(ll b,ll l,ll r){
54      s = getString(b,r);
55      ll ans = go(b,0,0,0,0);
56      if(l>1){
57          s = getString(b,--l);
58          ll x = go(b,0,0,0,0);
59          ans -= x;
60      }
61      return ans;
62  }
63  int main(){
64      memset(dp,-1,sizeof(dp));
65      for(int b=2; b<=10; b++){
66          fun(b,0,0,0);
67      }
68      int q; scanf("%d",&q);
69      while(q--){
70          ll b,l,r; scanf("%I64d %I64d %I64d",&b,&l,&r);
71          ll ans = solve(b,l,r);
72          printf("%I64d\n",ans);
73      }
74      return 0;
75  }
```