# Educational Codeforces Round 111

比赛链接

## D. Excellent Arrays

### 题意

我们称满足以下条件的数组 $a$ 为好数组

$a_{i} != i,i∈[1,n],n$ 是数组长度

我们规定 $F(a)$ 为一个数组中满足 $1≤i□j≤n$ 且 $a_{i}+a_{j}=i+j$ 的对数。

再规定完美的数组为好数组且 $l≤a_{i}≤r$ 且它的 $F$ 函数值为所有好数组中最大的那个，给定 $n,l,r$ □求完美数组的个数对 $10^{9}+7$ 取模□ $t$ 组数据□
$1≤t≤1000,2≤n≤2·10^{5},-10^{9}≤l≤1,n≤r≤10^{9}$

不难发现对于给定的 $n$ □最大的 $F$ 值应为 ${\frac n^{2} 4}$ □

首先证明其为上界，因为 $a_{i} != i$ □所以对于任意一个数，要么它变大了要么它变小了，假设有 $x$ 个变大的，则有 $n-x$ 个变小的，就算所有的大+小都正好和原来的下标和一样，也只有 $x×(n-x)$ 对，可以证明无论 $n$ 是奇数还是偶数，这个值都等于 ${\frac n^2 4}$ □故为上界。

而构造是显而易见的，将这个数组分成两组（如果是奇数就分成差为1的两组），一组同时加 $1$，另一组同时减 $1$ 即可，注意 $n$ 要分在减的那组， $1$ 要分在加的那组，这样一定保证所有的数变化后依然在 $[1,n]$ 范围内，又由题目的 $l,r$ 的范围知这样一定满足题意。

下面的问题是如何计数。我们可以分类讨论出最大的要加的值和最小的要减的值，这两个决定我们能加/减多少。我们不妨先讨论简答的情况：即 $n$ 是偶数的情况。刨除一种特殊的情况是 $1$ 到 ${\frac n 2}$ 的数要增加□ ${\frac n 2}+1$ 到 $n$ 的数减小。这时这两种数的区域没有交点，这种情况的答案是 $min({\frac n 2}+1-l,r-{\frac n 2})$ □其余的情况我们设最大的要加的值为 $j$ 取值范围为 $[{\frac n 2},n]$□最小的要减的值为 $i$ 取值范围为 $[1,{\frac n 2}]$ □这种情况下移动的长度是 $min(i-l,r-j)$ □因为既不能加过头，也不能减过头。于是这个式子是 $\sum_{j=\frac n 2}^{n} \sum_{i=1}^{\frac n 2} min(i-l,r-j)c(j-i-1,j-{\frac n 2}-1)$ □之后分 $i-l≥r-j$ 和 $i-l□r-j$ 讨论，注意边界范围就可以了，这个看起来是 $O(n^2)$ 的，但是它是很多组合数相加，分别利用 $c(n,k)+c(n-1,k)+\ldots+c(k,k)=c(n+1,k+1)$ 和 $c(n,0)+c(n+1,1)+\ldots+c(n+k,k)=c(n+k+1,k)$ 就可以做到整个式子 $O(n)$ 算出来了，这样复杂度 $O(nt)$ □可以通过。

奇数的情况只需要分类讨论是增加的多还是减小的多就可以了，相当于两个上述问题。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll mod=1e9+7;
int n,l,r;
int inv[200100],jc[200100];
int C(int n,int m) {
```

```
    if(n<m||n<0||m<0) return 0;
    if(n==m) return 1;
    return 1ll*jc[n]*inv[m]%mod*inv[n-m]%mod;
}
int min(int a,int b) {
    if(a>b) return b;
    return a;
}
int max(int a,int b) {
    if(a<b) return b;
    return a;
}
int main() {
    inv[0] = inv[1] = jc[0] = 1;
    for(ll i=1; i<=200010; ++i) {
        jc[i]=1ll*jc[i-1]*i%mod;
    }
    for(ll i=2; i<=200010; ++i) {
        inv[i]=(ll)(mod-mod/i)*inv[mod%i]%mod;
    }
    for(ll i=2; i<=200010; ++i) {
        inv[i]=1ll*inv[i-1]*inv[i]%mod;
    }
    int t;
    scanf("%d",&t);
    while(t--) {
        scanf("%d %d %d",&n,&l,&r);
        ll ans=min(r-n/2,n/2+1-l);
        if(n%2==0) {
            for(int j=n/2+1; j<=n; j++) {
                if(r-j+l>n/2) continue;
                int down=max(1,r-j+l);
                ans=(ans+1ll*(r-j)*C(j-down,j-n/2)%mod)%mod;
            }
            for(int i=1; i<=n/2; i++) {
                if(l+r-i-1<n/2+1) continue;
                int up=min(n,l+r-i-1);
                ans=(ans+1ll*(i-l)*C(up-i,up-n/2-1)%mod)%mod;
            }
            printf("%lld\n",ans);
        } else {
            ans=(ans+min(r-n/2-1,n/2+2-l))%mod;
            for(int j=n/2+1; j<=n; j++) {
                if(r-j+l>n/2) continue;
                int down=max(1,r-j+l);
                ans=(ans+1ll*(r-j)*C(j-down,j-n/2)%mod)%mod;
            }
            for(int i=1; i<=n/2; i++) {
                if(l+r-i-1<n/2+1) continue;
                int up=min(n,l+r-i-1);
```

```
                ans=(ans+1ll*(i-l)*C(up-i,up-n/2-1)%mod)%mod;
            }
            for(int j=n/2+2; j<=n; j++) {
                if(r-j+l>n/2+1) continue;
                int down=max(1,r-j+l);
                ans=(ans+1ll*(r-j)*C(j-down,j-n/2-1)%mod)%mod;
            }
            for(int i=1; i<=n/2+1; i++) {
                if(l+r-i-1<n/2+2) continue;
                int up=min(n,l+r-i-1);
                ans=(ans+1ll*(i-l)*C(up-i,up-n/2-2)%mod)%mod;
            }
            printf("%lld\n",ans);
        }
    }
    return 0;
}
```