Jon fought bravely to rescue the wildlings who were attacked by the white-walkers at Hardhome. On his arrival, Sam tells him that he wants to go to Oldtown to train at the Citadel to become a maester, so he can return and take the deceased Aemon's place as maester of Castle Black. Jon agrees to Sam's proposal and Sam sets off his journey to the Citadel. However becoming a trainee at the Citadel is not a cakewalk and hence the maesters at the Citadel gave Sam a problem to test his eligibility.

Initially Sam has a list with a single element *n*. Then he has to perform certain operations on this list. In each operation Sam must remove any element *x*, such that *x* > 1, from the list and insert at the same position , , sequentially. He must continue with these operations until all the elements in the list are either 0 or 1.

Now the masters want the total number of 1s in the range *l* to *r* (1-indexed). Sam wants to become a maester but unfortunately he cannot solve this problem. Can you help Sam to pass the eligibility test?

Input

The first line contains three integers *n*, *l*, *r* (0 ≤ *n* < 250, 0 ≤ *r* - *l* ≤ 105, *r* ≥ 1, *l* ≥ 1) – initial element and the range *l* to *r*.

It is guaranteed that *r* is not greater than the length of the final list.

Output

Output the total number of 1s in the range *l* to *r* in the final sequence.

Example

Input

7 2 5

Output

4

Input

10 3 10

Output

5

Note

Consider first example:

Elements on positions from 2-nd to 5-th in list is [1, 1, 1, 1]. The number of ones is 4.

For the second example:

Elements on positions from 3-rd to 10-th in list is [1, 1, 1, 0, 1, 0, 1, 0]. The number of ones is 5.

题意：给你一个数n和区间l,r，每次都能把任意数拆成n/2,n%2,n/2三个数，直到变成0和1，问区间l,r里有多少个1。

因为[l,r]的范围小于1e5，所以想到枚举区间中的第pos个数，判断这个数是0还是1即可。递归去判断最多50，所以不会超时。递归如何判断呢？每次把n去递归模拟，一层层往下，就可以找到第pos个位置，然后看一下是0还是1即可。

上面递归判断的思想稍微转化一下，线段树的结构呼之欲出（这不就是线段树吗WOW），所以找到n展开后的最大区间长度，然后在这个区间里去找[l,r]，这就很容易了。

类似线段树的查询

#include<iostream>

#include<stdio.h>

typedef long long ll;

using namespace std;

ll n,L,R;

int query(ll n,ll l,ll r)//在[l,r]区间上查询n

{

if(r<L||l>R||n==0)

return 0;

if(n==1)

return 1;

ll mid=(l+r)>>1;

return query(n/2,l,mid-1)+query(n%2,mid,mid)+query(n/2,mid+1,r);

}

int main()

{

cin>>n>>L>>R;

ll len(1),x=n;

while(x>1)

{

len=len\*2+1;

x/=2;

}

cout<<query(n,1,len);

return 0;

}