2's complement

Understanding *2's complement* representation is fundamental to learning about Computer Science. It allows us to write negative numbers in binary. The leftmost digit is used as a sign bit. If it is 1, we have a negative number and it is represented as the two's complement of its absolute value. Let's say you wrote down the 2's complement representation for each 32-bit integer in the inclusive range from a to b. How many 1's would you write down in all?

For example, using an 8-bit byte rather than 32 bit integer, the two's complement of a number can be found by reversing all its bits and adding 1. The two's complement representations for a few numbers are shown below:

```
        |Number|                Representation in
Number   Binary      Inverse     Two's Complement
-3      00000011    11111100    11111101
-2      00000010    11111101    11111110
-1      00000001    11111110    11111111
 0      00000000                00000000
 1      00000001                00000001
 2      00000010                00000010
 3      00000011                00000011
```

To write down that range of numbers' two's complements in 8 bits, we wrote 26 1's. Remember to use 32 bits rather than 8 in your solution. The logic is the same, so the 8 bit representation was chosen to reduce apparent complexity in the example.

Function Description

Complete the *twosCompliment* function in the editor below. It should return an integer.

twosCompliment has the following parameter(s):
- *a*: an integer, the range minimum
- *b*: an integer, the range maximum

Input Format

The first line contains an integer T, the number of test cases.

Each of the next T lines contains two space-separated integers, a and b.

Constraints

- $T \leq 1000$
- $-2^{31} \leq a \leq b \leq 2^{31} - 1$

Output Format

For each test case, print the number of 1's in the 32-bit 2's complement representation for integers in the inclusive range from a to b on a new line.

Sample Input 0

```
3
-2 0
-3 4
-1 4
```

Sample Output 0

```
63
99
37
```

Explanation 0

Test case 0
-2 has 31 ones
-1 has 32 ones
0 has 0 ones
31+32+0 = 63
Test case 1
-3 has 31 ones
-2 has 31 ones
-1 has 32 ones
0 has 0 ones
1 has 1 ones
2 has 1 ones
3 has 2 ones
4 has 1 ones
31+31+32+0+1+1+2+1 = 99
Test case 2
-1 has 32 ones
0 has 0 ones
1 has 1 ones
2 has 1 ones
3 has 2 ones
4 has 1 ones
32+0+1+1+2+1 = 37

Sample Input 1

```
4
-5 0
1 7
-6 -3
3 6
```

Sample Output 1

```
155
12
```

```
122
7
```

Explanation 1

Test case 0
-5 has 31 ones
-4 has 30 ones
-3 has 31 ones
-2 has 31 ones
-1 has 32 ones
0 has 0 ones
31+30+31+31+32+0 = 155
Test case 1
1 has 1 ones
2 has 1 ones
3 has 2 ones
4 has 1 ones
5 has 2 ones
6 has 2 ones
7 has 3 ones
1+1+2+1+2+2+3 = 12
Test case 2
-6 has 30 ones
-5 has 31 ones
-4 has 30 ones
-3 has 31 ones
30+31+30+31 = 122
Test case 3
3 has 2 ones
4 has 1 ones
5 has 2 ones
6 has 2 ones
2+1+2+2 = 7