XOR FROM LEFT TO RIGHT

✡ In this problem, we are given 2 numbers L and R, and our job is to find the XOR of all numbers from L → R.

Brute force solution is to obviously iterate through L → R and calculate entire XOR.

In case L = 1, Optimal Solution comes from a pattern. Every fourth XOR is 1.

N =	XOR =
1	1
2	3
3	0
4	4
5	1
6	7
7	0
8	8
9	1

∴ if (N % 4 == 0) return N
if (N % 4 == 1) return 1
if (N % 4 == 2) return N + 1
if (N % 4 == 3) return 0

But if $L \neq 1$, then we use the above algorithm for $1 \to L$ and then for $1 \to R$. Then we just XOR both of these.

Eg : $L = 4$ $\quad R = 7$

$$(\cancel{1} \wedge \cancel{2} \wedge \cancel{3}) \wedge (\cancel{1} \wedge \cancel{2} \wedge \cancel{3} \wedge 4 \wedge 5 \wedge 6 \wedge 7)$$

$$= (4 \wedge 5 \wedge 6 \wedge 7)$$

```
xorRange(N) {
    if (N % 4 == 0)  return  N
    if (N % 4 == 1)  return  1
    if (N % 4 == 2)  return  N + 1
    if (N % 4 == 3)  return  0
}

xorRangeLeftToRight(L, R) {
    return  xorRange(L - 1) ^ xorRange(R)
}
```