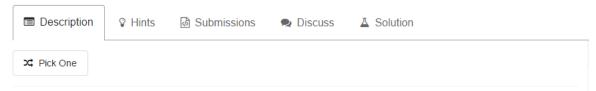
## 89. Gray Code



The gray code is a binary numeral system where two successive values differ in only one bit.

Given a non-negative integer *n* representing the total number of bits in the code, print the sequence of gray code. A gray code sequence must begin with 0.

## Example 1:

```
Input: 2
Output: [0,1,3,2]
Explanation:
00 - 0
01 - 1
11 - 3
10 - 2

For a given n, a gray code sequence may not be uniquely defined.
For example, [0,2,3,1] is also a valid gray code sequence.

00 - 0
10 - 2
11 - 3
01 - 1
```

## Example 2:

```
Input: 0

Output: [0]

Explanation: We define the gray code sequence to begin with 0.

A gray code sequence of n has size = 2^n, which for n = 0 the size is 2^0 = 1.

Therefore, for n = 0 the gray code sequence is [0].
```

```
* 这道题主要是靠格雷码的规律。
* 可以看到n位的格雷码由两部分组成,一部分是n-1位格雷码,另一部分是1<<(n-1) 和n-1位格雷码的逆序的和。
 * 1位的格雷码有两个码字 0和1
 * (n+1)位格雷码的前2^n个码字等于n位格雷码的码字,按顺序书写,加前{}^{ } {}^{ }
 * (n+1)位格雷码的后2^n个码字等于n位格雷码的码字,按逆序书写,加前缀1.
* 由于是二进制,在最高位加00跟原来的数本质没有变化,所以取得上一位算出的格雷码结果,再加上逆序添1的方法就是当前这位格雷码的结果了。
 * n=例寸,[0];
 * n=1时, [0, 1];
 * n=281, [00, 01, 11, 10]
 * n=3时, [000,001,011,010,110,111,101,100]
*当n=1时,0,1
 *当n=2时,原来的list 0,1不变,只是前面形式上加了个0变成了00,01。然后加数是1<<1为10,依次:10 + 1=11,10 + 0 = 10,结果为00,01,11,10
 *当n=3时,原来的list00,01,11,10(倒序为:10,11,01,00)。加数1<<2为100。倒序相加为:100+10=110,100+11=111,100+01=101,100+(
 * 最终结果为000,001,011,010,110,111,101,100
public class L89 {
     public List<Integer> grayCode(int n) {
         if (n == 0) {
           List<Integer> result = new ArrayList<Integer>();
           result.add(0);
           return result;
       }
         //这里是得到前半部分的数。
         List<Integer> result = grayCode(n-1);
         int addNumber = 1 << (n-1);</pre>
         int originalsize = result.size();
         //这里是得到后半部分的数
         for(int i = originalsize - 1; i >= 0; i--) {
             result.add(addNumber + result.get(i));
         return result;
     }
}
```