题目：

编写一个函数，输入是一个无符号整数，返回其二进制表达式中数字位数为 ‘1’ 的个数（也被称为汉明重量）。

示例 1：

输入： 00000000000000000000000000001011
输出： 3
解释： 输入的二进制串 **00000000000000000000000000001011** 中，共有三位为 '1'。

示例 2：

输入： 00000000000000000000000010000000
输出： 1
解释： 输入的二进制串 **00000000000000000000000010000000** 中，共有一位为 '1'。

示例 3：

输入： 11111111111111111111111111111101
输出： 31
解释： 输入的二进制串 **11111111111111111111111111111101** 中，共有 31 位为 '1'。

解法：

```
# 方法 1：公式法
```

```python
def hammingWeight(self, n: int) -> int:
    return bin(n).count('1')
```

```
# 方法2 十进制转二进制，每次取余数，判断是否为1，超时了
```

```python
def hammingWeight(self, n: int) -> int:
        count = 0
        for i in range(n):
            if n % 2 == 1:
                count += 1
            # 注意，在python3中，n = 9, n / 2 = 4.5, n /
            n = n // 2
        return count
```

方法3：先把n转化为二进制，手动循环计算1 的个数

```
def hammingWeight(self, n: int) -> int:
        n = bin(n)
        count = 0
        for c in n:
            if c == "1":
                count += 1
        return count
```

方法4:把n与1进行与运算，将得到n的最低位数字，可以取出最低位数，在将n右移一位，循环此步骤，直到n等于0

```
def hammingWeight(self, n: int) -> int:
        count = 0
        while n:
            count += n & 1
            n >>= 1
        return count
```

方法5 ： 每次都清零最低位的1，一直到最后为0. n & (n - 1)清零最低位的1，

```
def hammingWeight(self, n: int) -> int:
        count = 0
        while n:
            count += 1
            n = n & ( n - 1)
        return count
```