补充文档0x02：Bits Bytes 傻傻分不清楚

Bit和Bytes

分别翻译作 比特(位)和字节

首先我们理一下数值上的关系 $1Bytes = 8bits$

Bytes是大单位，bit是小单位

bit，是我们计算机的内存的基本单元，一个Bit就像是一个开关，只有开/关两个状态，没错，一个bit，只能容纳一个1或者0，一个位，只能容纳0/1，所以位也只能记录二进制数字

一个二进制数字，$(10101100)_2$，就需要8个bit，

然鹅字节是一个比bit大的单位，因为单独讨论bit，有时候会把数字弄得太大

比如某个网线的传输速度是8000bit每秒，就不如说成1000Bytes每秒

两个单位都是B开头，简写的时候为了有所区分，bit就用小写的b，bytes就用大写的B

这是两个单位，有时候我们看有些无良的网络经销商，做网线安装服务的时候会说

“我们的网络是1Gb每秒！”

害，1Gb每秒算个屌咯，除以8，才是你在百度云/迅雷上看到的那个数值（的上限）

在C语言中我们讨论的int，就是4个字节的整数

4个字节，就是32位

32位，也就是 有32个放0/1的地方

```c
int main()
{
    int i = 11;
    return 0;
}
```

这一段代买的第三行`int i = 11`

意思就是，内存里开辟32个位(4个字节)，存放数字11对应的二进制数字

所以这个变量`i`在内存中的储存“状态”，就是：

`0000 0000 0000 0000 0000 0000 0000 0111`

那么“-11”咋办哪！

简单，把所有的0都变成1，把1变成0，在+1

就是

`0000 0000 0000 0000 0000 0000 0000 0111`

-> `1111 1111 1111 1111 1111 1111 1111 1000`+1

->`1111 1111 1111 1111 1111 1111 1111 1001`

这就是-11啦

负数也就是：

1. 先把任何进制的数字转换为二进制，这一步叫做取原码
2. 然后把0和做反转，这一步叫做取反码，把源码的1和0进行反转，就是反码
3. 然后在尾巴上+1，不够就进1，这一步叫做取补码

好啦，不用纠结bit和bytes在物理上是个东西

你只要知道

1个Bit(位/比特)可以储存1一个1/0

1个bytes等于8个bit

就好啦