代码之谜（一）- 有限与无限 从整数的绝对值说起

一、引子

开始本章之前我先提个问题：“如果一个整数的绝对值等于它自己，那么这个数是几？”如果你回答是 0 和 所有正数，那么请你耐心读完这篇文章吧。

本章是我『代码之谜』系列的第二篇，前一篇『代码之谜 - 开篇/前言/序』简单介绍了计算机与数学的不同。

数学中有许多复杂深刻的矛盾，数学家的工作就是解释或者反驳这些矛盾， 例如有限与无限、连续与离散、存在与构造、逻辑与直观、具体与抽象、概念与计算等等。

在本章中，我们把目标缩小，主要讨论内容

- 概念: 有限与无限
- 对象: 8bit整数

二、绝对值之谜

终于到主题了，也许你很想知道“负数的绝对值可能等于自己吗？”，也就是 “如果x等于-x，那么x有几个解？”按照我一贯的作风，我是不会轻易告诉你答案的。 《编程珠玑》记载，作者告诉了他同事一个结果，而不是方法，最后追悔莫及。 所以，我在这里要**告诉你方法，而不是告诉你答案。**

告诉你答案之前，首先得回答个问题：“整数(8bit)的表示范围是多少？”，（也许你已经把教科书的知识背下来了，是 -2^7 到 2^7 - 1，也就是 -128 到 +127，现在的计算机科学都快成为文科了 ^_^ ）。

如果你不知道也没关系，至少你知道 8bit 可以表示的整数个数是 2^8 个，这个数等于多少无所谓，但是，它一定是个偶数(256)。

那么这里就有一个很有意思的问题了，0既不是正数也不是负数，把0去掉的话，整数的个数就是奇数了，整数还剩 255 个。 奇数个整数不可能平均分成两部分(正数和负数)，要么负数多一个，要么正数多一个。事实就是，负数比正数多一个，最小的负数是 -128， 最大的整数是 127。

现在的问题是， -128 的绝对值是多少呢？ -(-128)等于多少呢？是溢出呢，还是等于它自己呢？也许计算机课本没有告诉你， **整数是不会出现溢出异常的**，整数的溢出被认为是正常的舍弃（其实只要很合理）。**整数只有被0除才会异常，而浮点数，即使被0除也不会抛出异常。** 浮点数除0的操作将放在本系列浮点数篇讨论。

绝对值等于自己的数有两个，0 和最小的负数。

你可能要像香港电影里女主角那样歇斯底里的大喊“绝对值怎么可能是负数呢？ 不可能，我不信，我不信…”

忘掉你那可怜的数学知识吧，“发生这种事，大家都不想的。感情的事呢，是不能强求的。所谓吉人自有天相，做人呢，最要紧的就是开心…”跑题了，赶紧回来。

在经典数学（非皮亚诺算术系统，皮亚诺绝对是欧几里德的铁杆粉丝，要不怎么会有如此天才的构想，这个以后会给大家普及）中，绝对值定义为：“从原点到点A的距离，称为A的绝对值，它是一个非负数”。 既然讲到了距离，不妨剧透一下（本系列“逻辑篇”会涉及到），两个数的大小在数学中如何定义，“距离数轴原点的距离远近”，计算机中大小如何定义的呢？给大家留个作业吧（别告诉我是设计编程语言或者设计电脑的科学家规定的，计算机科学绝对不是文科）。

计算机中没有数轴，绝对值是如何定义的呢？看看java、C、Python的源码（感谢那些开源大牛），和咱们学的小学数学一样。

```
abs(x) :=  (x >= 0) ? x : -x
```

翻译过来就是，x的绝对值定义为：正数和0的绝对值等于它自己，负数的绝对值等于-x。（这里使用的是-x，而没有用0-x，因为在浮点数中，这两者是有区别的。 ）

三、深入 -x

那么 -x 是如何计算的呢？ **计算**是数学概念，在计算机中，我们应该说 -x是如何**求值**的呢？还得回到源码，我只看了linux中关于C的源码，如果你看过其它语言源码发现和我说的不同，请联系我。 学过计算机原理的都知道，负数在计算机中以补码形式存储，计算补码的方式和取反操作类似。

```
符号位不变，其它位取反，最后加一。
```

比如 -5

```
原码:        1000,0101
其它位取反:  1111,1010
加一:        1111,1011
```

当我们求它的绝对值时如何操作呢

```
补码:      1111,1011       这是-5在计算机中的表示
各位取反:  0000,0100
加一:      0000,0101       此时结果为+5
```

现在我们回到最小的负数问题，最小的负数在计算机中表示为 1000,000，下面我们对这个数操作

```
补码:      1000,0000
各位取反:  0111,1111
加一:      1000,0000
```

神奇吗，尼玛，居然又回到自己了。

继续阅读关于 代码之谜 的文章