前两天看到一篇介绍[二分原理](http://wz.csdn.net/item/2698569/" \t "_blank) 的帖子，想起了以前写二分法的事情。二分法看似简单，但实际写的时候却发现 +1 -1 的地方很容易弄错。幸好之前看过循环不变量的介绍。

所谓循环不变量，是指在循环过程中保持不变的量。具体取什么样的量呢？显然，pi之类的常量在任何循环中都保持不变，但对分析循环并没有用处。

因此，为便于分析，循环不变量一般会取一个关于循环中的变量 V 的布尔函数 F，在整个循环过程中，F(V)为真，而当循环结束后，(F(V)->R)为真，R是循环的目的。这样，只要证明 F(V) ，即证明了这个循环达到了所要求的目的。

以二分法为例：已知 a[1..n] 是单调递增的数列，求 a 中所有大于或等于 v 的值中，最小的那一个的序号。

如果只是求 a 中等于 v 的值的序号，程序是很容易写出的：

**[c-sharp]** [view plaincopy](http://blog.csdn.net/yizhili/article/details/4705377)

1. find( a[1..n], v )
2. {
3. **int** min = 1, max = n;
4. **while**( min < max )
5. {
6. **int** mid = (min + max)/2;
7. **if**( a[mid] == v ) **return** mid;
8. **else** **if**( a[mid] < v ) min = mid + 1;
9. **else** max = mid - 1;
10. }
11. **if**( min == max && a[min] == v ) **return** min;  // a[min] 并没有比较过
12. **else** **return** -1;  // 没找到
13. }

在看下面的分析前，不妨试着先写一个找 a 中大于或等于 v 的最小值的函数，看看循环不变量究竟有没有用。

(说明一下 ":=" 表示“定义为”，"="表示“赋值”，"=="表示“相等”)

在这循环中用到的变量是min,max，首先想到的循环不变量是 F(min,max) := (a[min] <= v <= a[max])，但其中的问题也是显而易见的：循环开始时，v 可能小于 a[1]，也可能大于 a[n]，而在循环过程中，如果 a[mid-1] < v < a[mid]，或者 a[mid] < v < a[mid+1]，也会导致 F(min,max) 为假。

解决的方法是在 a 的首尾添加哨兵，即假设 a[0] == -∞， a[n+1] == +∞，循环开始时，min = 0，max = n+1，F(min,max) := (a[min] < v < a[max]) 为真。

循环过程中，当 a[mid] < v 时 min = mid，于是有F(min,max) := (a[min] < v < a[max]) == (a[mid] < v < a[max]) 为真；同样的，当 v < a[mid] 时 max = mid，于是有 F(min,max) := (a[min] < v < a[max]) == (a[min] < v < a[mid]) 为真。

由于 F(min,max) 为真，必然有 min < max (否则，min>=max 且 a[1..n] 升序 -> a[min] >= a[max]，与 F(min,max)为真矛盾)，因此循环应当在 min == max-1 时结束。此时有：a[max-1] == a[min] < v < a[max]，即 a[max] 是 a 中大于或等于 v 的最小值。

程序如下：

**[c-sharp]** [view plaincopy](http://blog.csdn.net/yizhili/article/details/4705377)

1. upBound( a[1..n], v )
2. {
3. **int** min = 0, max = n+1;
4. **while**( min < max-1 )
5. {
6. **int** mid = (min + max)/2;
7. **if**( a[mid] == v ) **return** mid;
8. **else** **if**( a[mid] < v ) min = mid;
9. **else** max = mid;
10. }
11. **return** max;
12. }

循环不变量，并不能用来设计出一个算法，但却可以帮助程序员在写循环时不致因为一些细节问题出错，也可以用于证明循环的正确性，便于阅读者分析循环，增强对代码的信心。