双指针算法

什么是双指针

严格的来说，双指针只能说是是算法中的一种技巧。

双指针指的是在遍历对象的过程中，不是普通的使用单个指针进行访问，而是使用两个相同方向（快慢指针）或者相反方向（对撞指针）的指针进行扫描，从而达到相应的目的。

双指针问题套路

通俗的说，就是在数组遍历中，我们使用两个指针进行操作。所以双指针问题基本有以下几个细节：

1、双指针的初始位置。根据双指针的分类，有两种可能。具体看下面的介绍。

2、双指针的移动方法。根据双指针的分类，有两种可能。具体看下面的介绍。

1. 遍历的结束条件。根据双指针的分类，有两种可能。具体看下面的介绍。
2. 对撞指针

****对撞指针****是指在数组中，将指向最左侧的索引定义为左指针（left），最右侧的定义为右指针（right），然后从两头向中间进行数组遍历。快速排序就是典型的双指针问题。

代码细节

指针初始位置

左指针（left)一般指向数组的第一个元素。即 left = 0。

右指针（right)一般指向数组的第一个元素。即 right = n-1。

指针移动方法

左指针(left）向右边👉移动，一般每次移动一个位置，即 left++。

右指针(right）向左边👈移动，一般每次移动一个位置，即 right--。

结束条件

左指针（left）位置和右指针（right）位置逆序。

从上面的描述可知，开始的时候，right >= left。因此结束的条件就是 right < left。

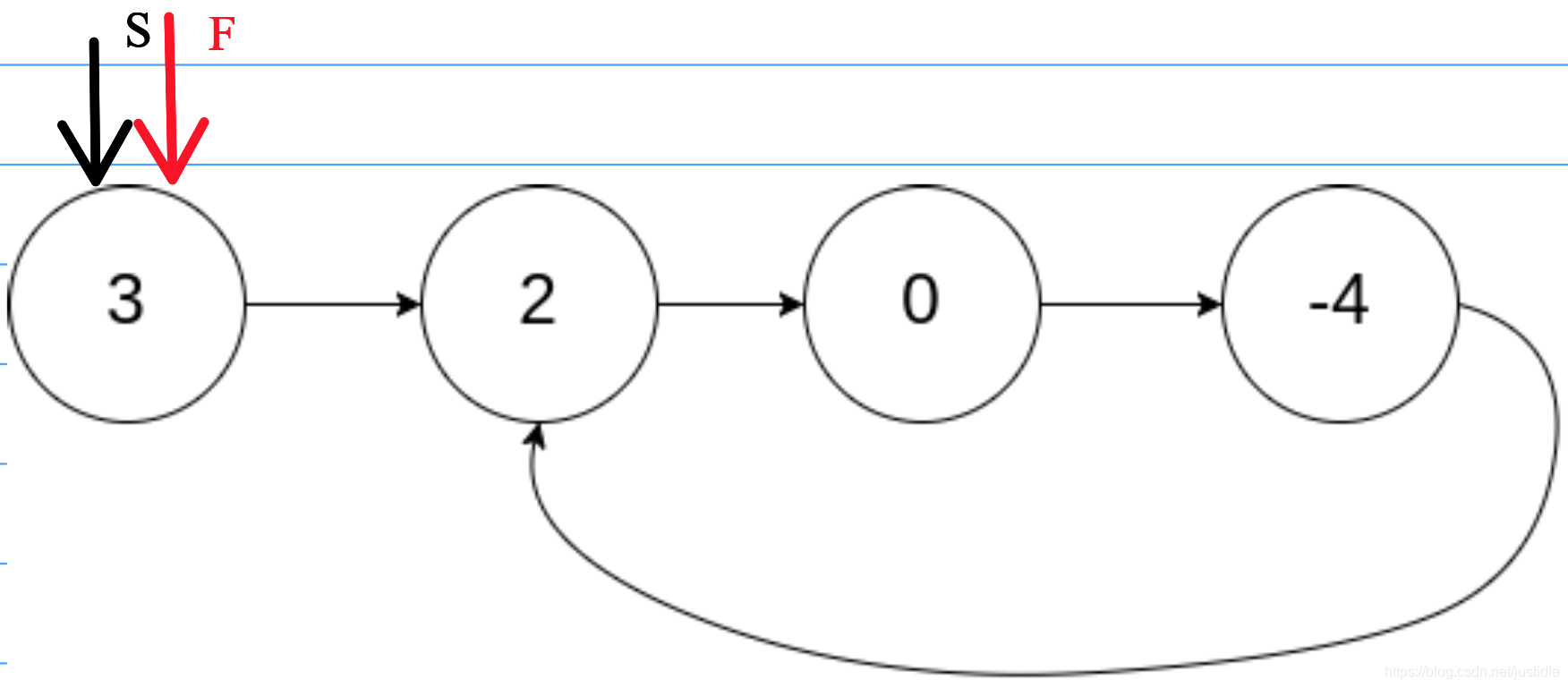
快慢指针

快慢指针是两个指针从同一侧开始遍历数组，将这两个指针分别定义为快指针（fast）和慢指针（slow），两个指针以不同的策略移动，直到两个指针的值相等（或其他特殊条件）为止，如快指针（fast）每次增长两个，慢指针（slow）每次增长一个。

一般来说，快慢指针常用于判断链表等数据结构中是否有环。比如下图所示的一个链表。

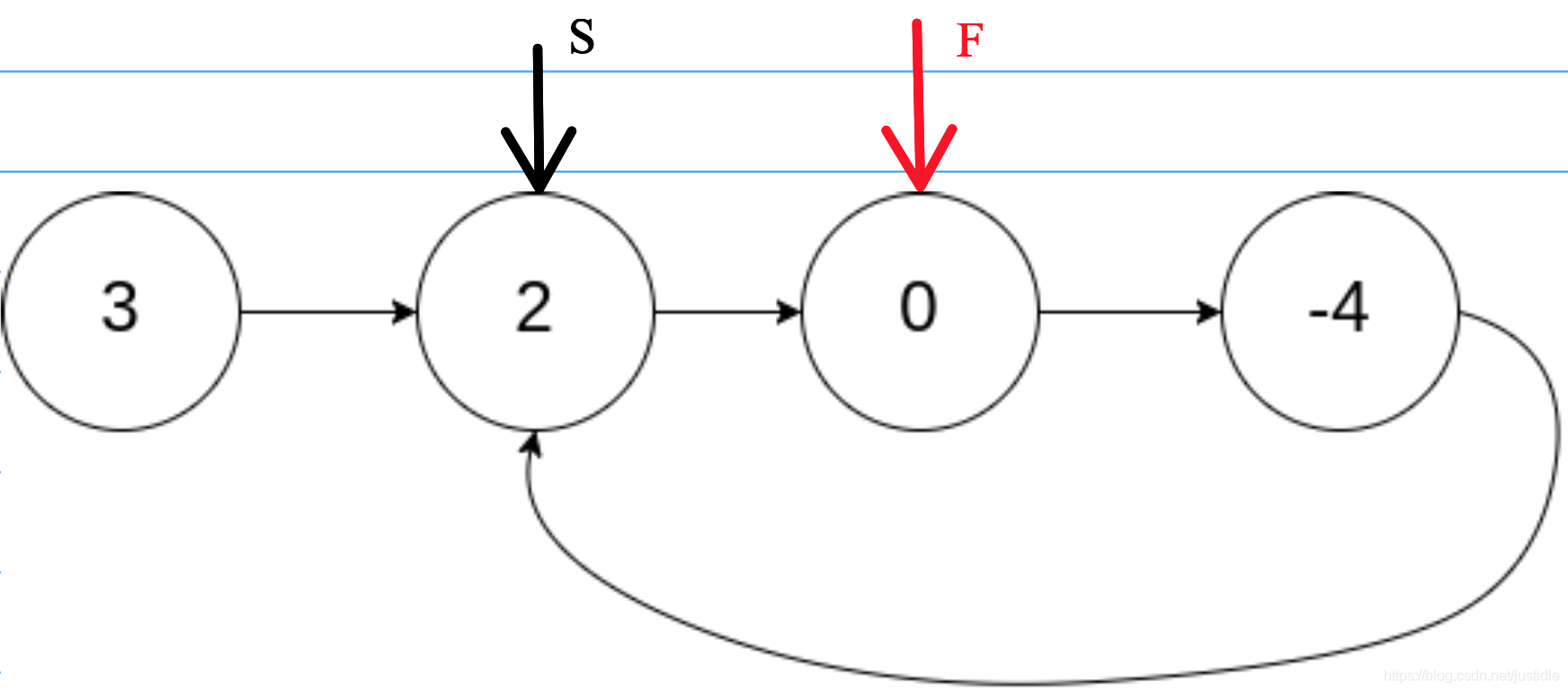
初始位置

如下图所示，慢指针（slow）和快指针（fast）同时指向第一个元素。



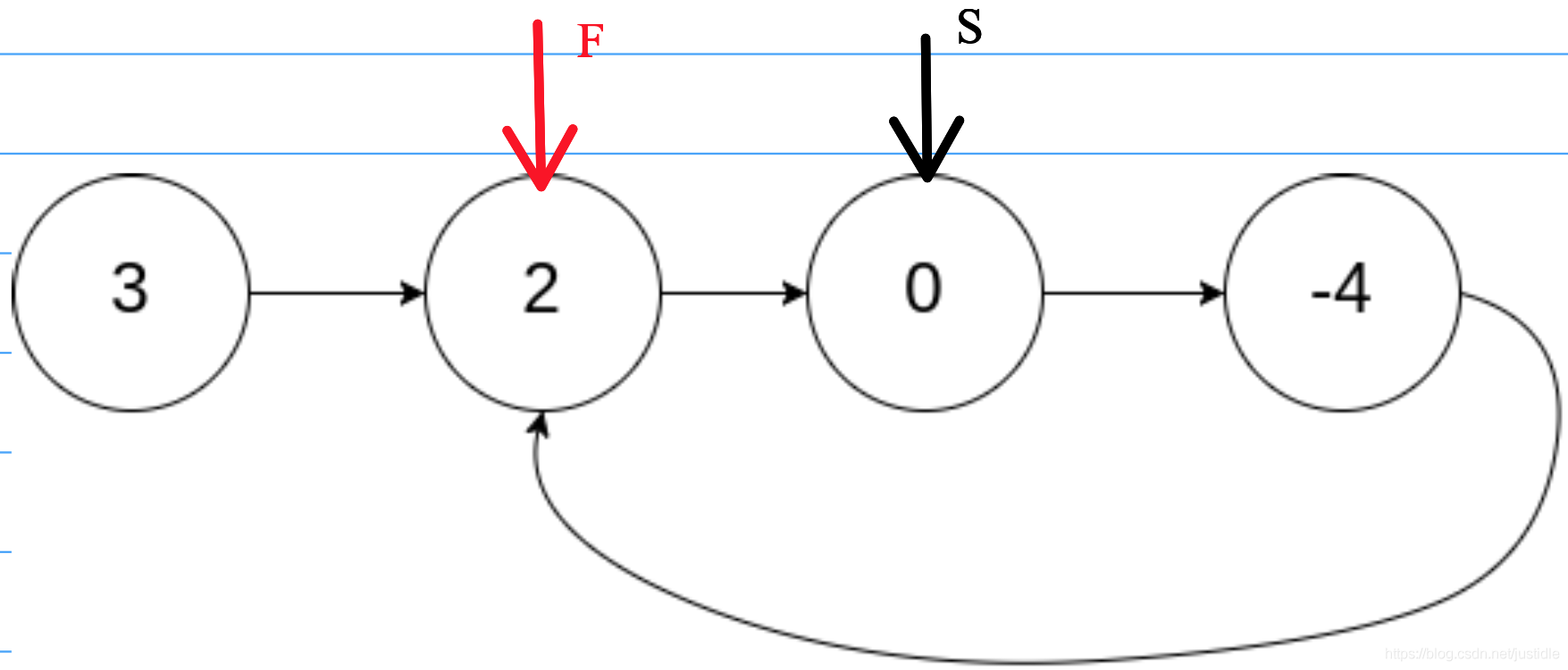
移动一次

慢指针（slow）向后移动一个位置，快指针（fast）向后移动两个位置，如下图所示。



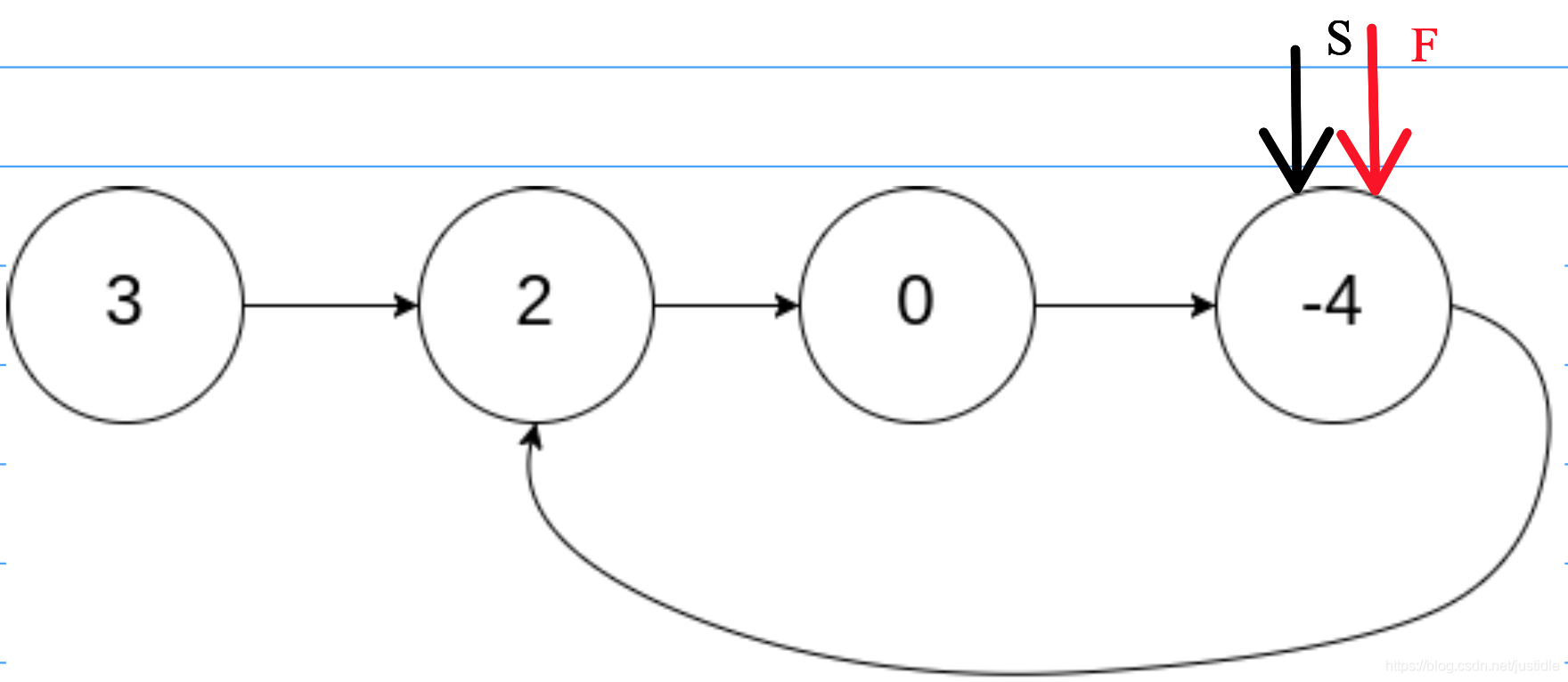
移动两次

慢指针（slow）向后移动一个位置，快指针（fast）向后移动两个位置，如下图所示。



移动三次

慢指针（slow）向后移动一个位置，快指针（fast）向后移动两个位置，如下图所示。我们发现慢指针（slow）和快指针（fast）重合，说明有环。



代码细节

指针初始位置

慢指针（slow）一般指向数组的第一个元素。即 slow = 0。

快指针（fast）一般指向数组的第一个元素。即 fast = 1。

指针移动方法

慢指针（slow）向右边👉移动，一般每次移动一个位置，即 slow++。

快指针（fast）向右边👉移动，一般每次移动两个个位置，即 fast += 2。

结束条件

慢指针（slow）位置和快指针（fast）位置重合；快指针（fast）达到数组的最后一个元素。

特别注意，要判断达到最后一个元素。否则会出现死循环。