Coursera Algorithms week2 基础排序 练习测验: Dutch national flag 荷兰国旗问题算法

第二周课程的Elementray Sorts部分练习测验Interview Questions的第3题荷兰国旗问题很有意思。题目的原文描述如下：

Dutch national flag. Given an array of n buckets, each containing a red, white, or blue pebble, sort them by color. The allowed operations are:

- *swap*(i,j): swap the pebble in bucket *i* with the pebble in bucket *j*
- *color*(i)determine the color of the pebble in bucket *i*

The performance requirements are as follows:

- At most *n* calls to *color*()
- At most *n* calls to *swap*()
- Constant extra space.

该问题在维基百科上的解释为：

The Dutch national flag problem (DNF) is a computer science programming problem proposed by Edsger Dijkstra.The flag of the Netherlands consists of three colors: red, white and blue. Given balls of these three colors arranged randomly in a line (the actual number of balls does not matter), the task is to arrange them such that all balls of the same color are together and their collective color groups are in the correct order.

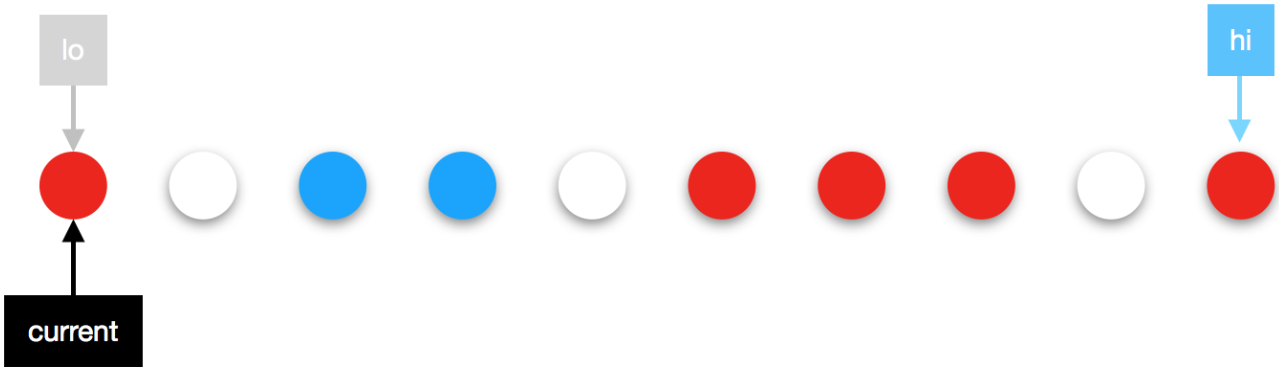
本题的限制条件n calls to color()意味着每个元素最多只能访问一次，也就是要求在遍历一次所有元素的情况下完成排序。遍历结束时红色球要全部在数组的左侧，白色球全部在中间，蓝色球全部在右侧，整个数组将被分成三部分。用current记录当前需要访问的球的位置，还需要两个位置记录红-白分界处和白-蓝分界处，用lo来标记白色球的开始位置，hi来标记白色球的结束位置。lo和current起始相同，两个从数组左侧以++方式前进，hi从数组右侧以--方式前进。循环主体设计方式：【后来看到3way-quicksort，发现这个做法与3way-quicksort很像，而且后者代码更清晰简单，特在文末附了3way-quicksort的实现】

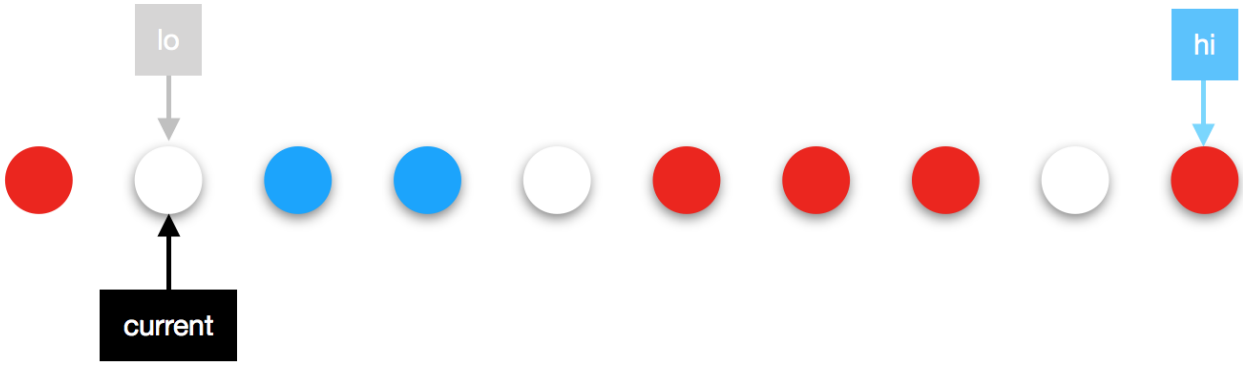
1、当current为红色时，如果lo==current，不需要交换位置，二者均前进++一步；如果不相等，意味着current 肯定大于lo，lo和current之间都为白色球，将lo和current位置处的球进行交换，lo处就变成了红色球，current处就变成了白色球，lo需要++前进一步来标志白色球的起始位置，而此时current处的球在之前已经被遍历过（左侧的球均被遍历过），current也需要++前进一步来访问下一个球

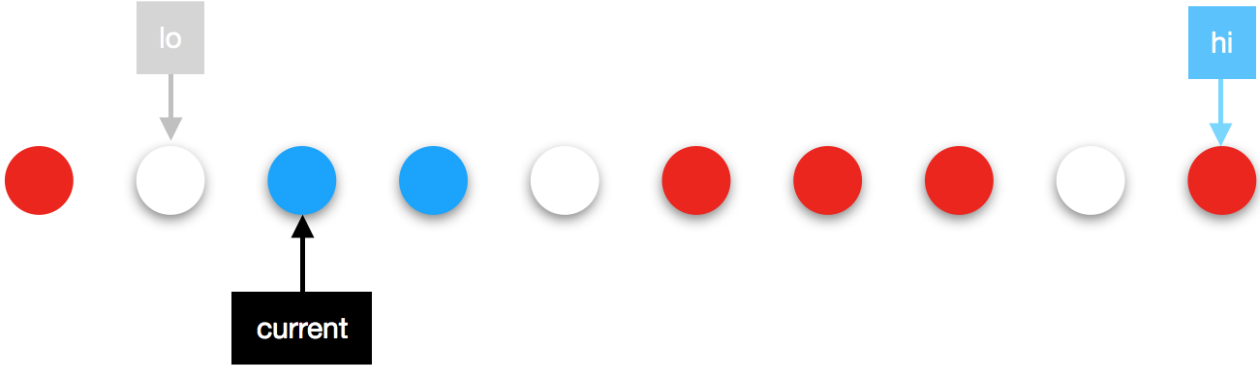
2. 当current为白色时，该球位置正确，不需要移动，current直接++前进一步

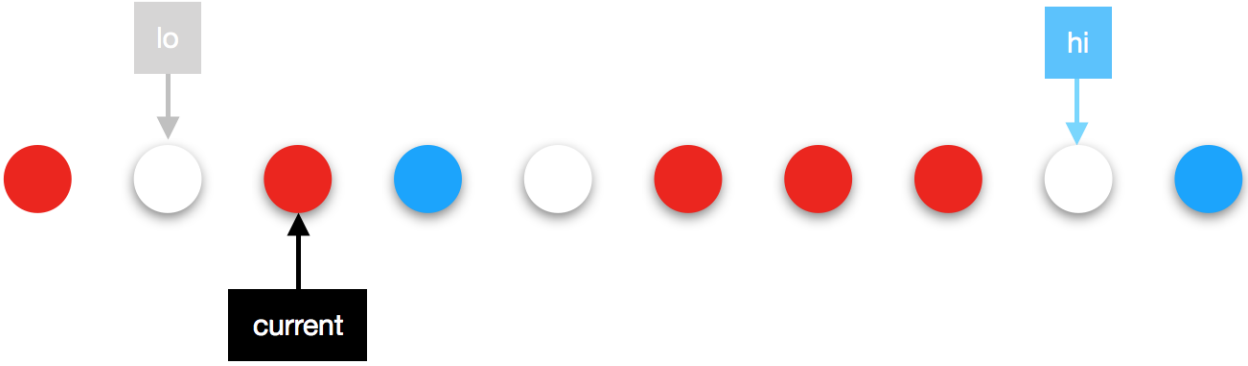
3. 当current为蓝色时，表明该球应该放置于数组右侧，此时需要和数组右侧hi处的球交换，然后hi处的球就变成了蓝色，并且该蓝色球是已经被访问过的，hi需要往右侧--前进。而此时current处交换过来的球还没有被访问过，故current位置不需要变动，留到下一次循环进行访问。

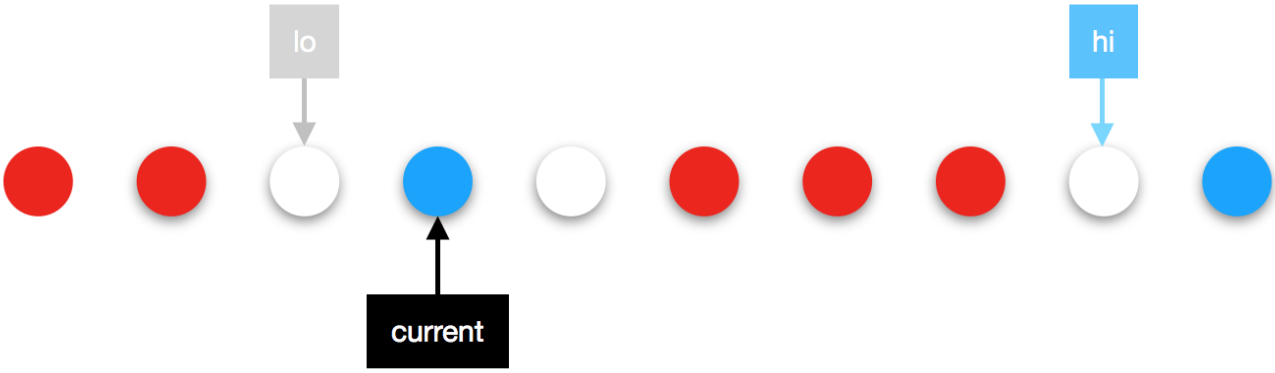
循环体在current访问完所有的球后结束，也就是current>hi时结束，因为大于hi的节点都是由current遍历交换过来的。

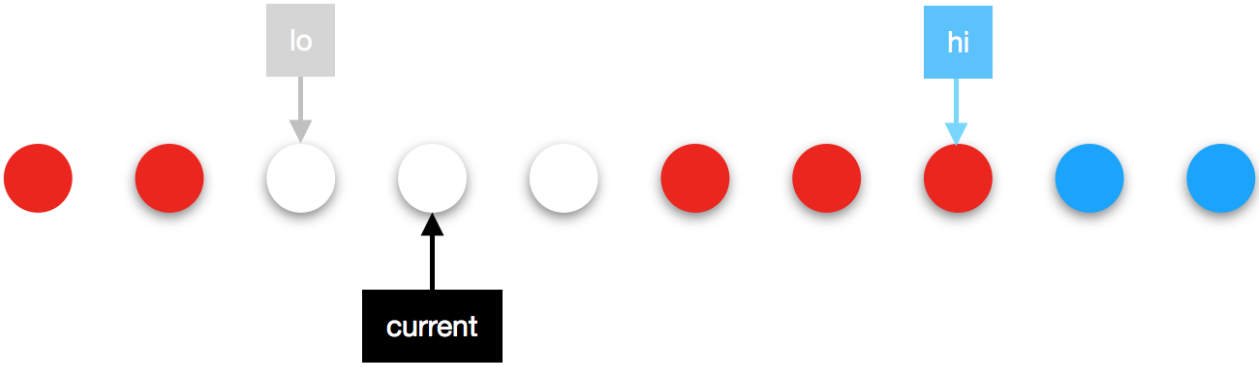


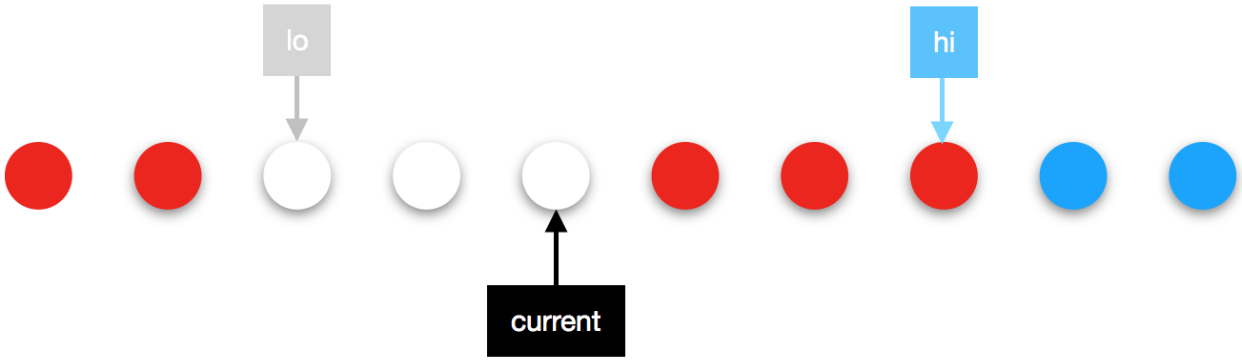

lo
hi
current

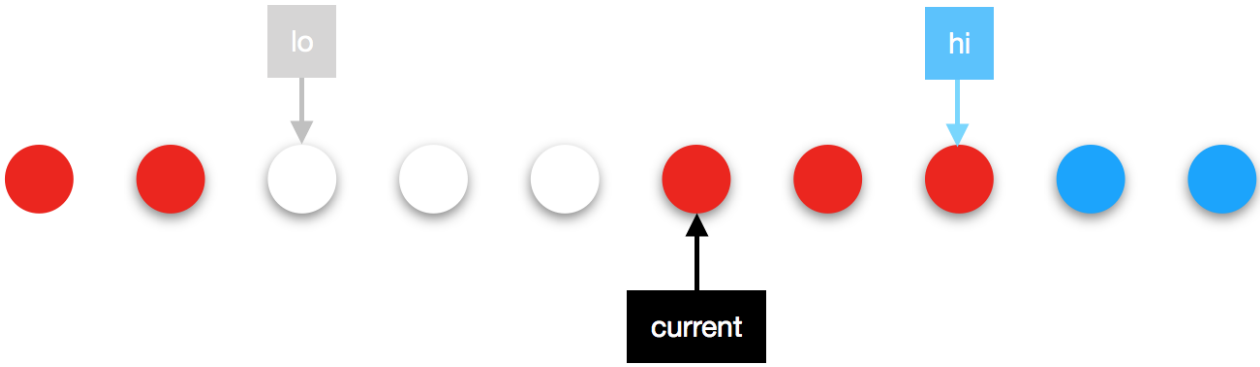

lo
hi
current

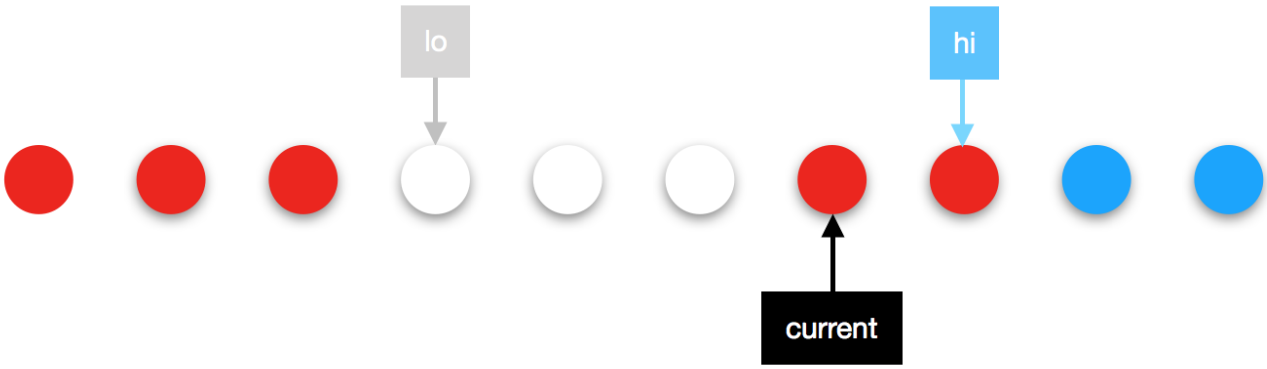

lo
hi
current

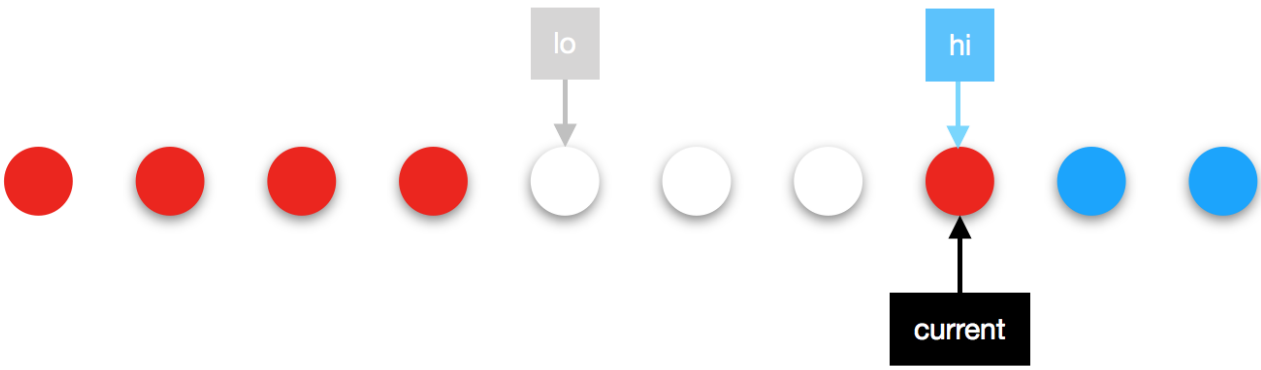

lo
hi
current

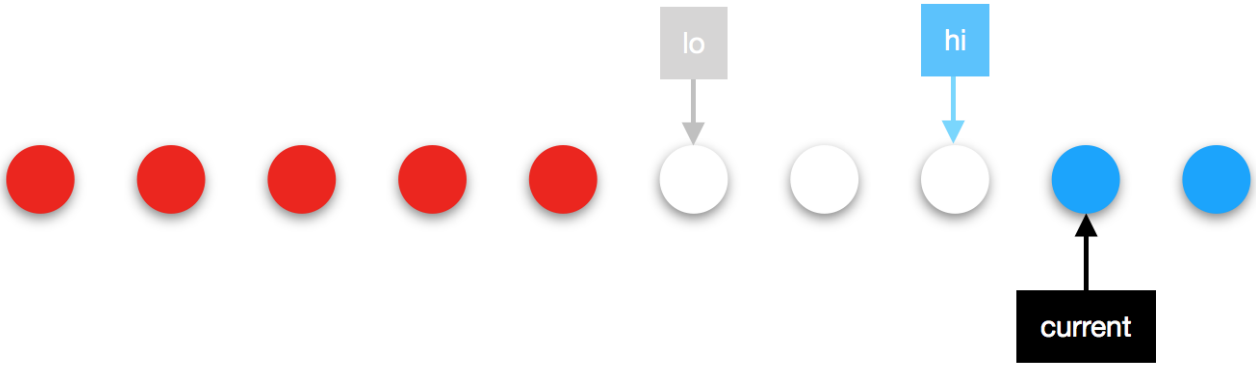

lo
hi
current


lo
hi
current


lo
hi
current


lo
hi
current


lo
hi
current



代码实现：

```
import java.util.Arrays;
import edu.princeton.cs.algs4.StdRandom;

public class DutchNationalFlag {
    public static final int RED = 0;
    public static final int WHITE = 1;
    public static final int BLUE = 2;
    private int n;
    private int[] buckets;
    public int callColorNum = 0;
    public int callSwapNum = 0;

    public DutchNationalFlag(int[] buckets) {
        n = buckets.length;
        this.buckets = buckets;
    }

    public void sort() {
        int lo = 0;
        int hi = n - 1;
        int current = lo;
        while (current <= hi) {
            switch(color(current)){
            case RED:
                if (current != lo)
                    swap(current, lo);
                current++;
                lo++;
                break;
            case WHITE:
                current++;
                break;
            case BLUE:
                swap(hi,current);
                hi--;
                break;

            }
        }
```

```
    }

    private void swap(int i, int j) {
        callSwapNum++;
        int t = buckets[i];
        buckets[i] = buckets[j];
        buckets[j] = t;
    }

    private int color(int i) {
        callColorNum ++;
        return buckets[i];
    }

    public static void main(String[] args) {
        int n = 10;
        int[] buckets = new int[n];
        for (int i = 0; i < n; i++) {
            buckets[i] = StdRandom.uniform(3);
        }
        System.out.println(Arrays.toString(buckets));
        DutchNationalFlag dnf = new DutchNationalFlag(buckets);
        dnf.sort();
        System.out.println("after sort call color+"+dnf.callColorNum+"times, call swap="+ dnf.callSwapN
        System.out.println(Arrays.toString(buckets));
    }
}
```

思路参考http://www.cnblogs.com/gnuhpc/archive/2012/12/21/2828166.html

3way-quicksort实现：

```
    public void quickSort3way(){
        int lt = 0;
        int gt = n - 1;
        int i = 0;
        while (i <= gt) {
            int cc = color(i);
            if(cc < WHITE) swap(lt++,i++);
            else if(cc > WHITE) swap(i,gt--);
            else i++;
        }
    }
```