Example for invariant in a while loop

This problem is called 2-color Dutch National Flag Construction. Given an array of two colors, RED and BLUE, the program is supposed to rearrange the colors in such a way that all the RED colors come before all the BLUE colors. For simplicity of programming, we represent the RED color by '0' and the BLUE color by '1' so that our array consists of zeroes and ones. We randomly initialize the array with zeroes and ones. Following is the program that performs the re-arranging.

```
int[] A = new int[25];
// initialize A with random number of 0 and 1.
for (int i = 0; i < A.length; i++)
   if ((int)(Math.random()*10) >= 5)
      A[i] = 1;
   else A[i] = 0;
// rearrange the colors
int m = 0, k = 0, temp = 0;
while (m != A.length){
   if (A[m] == 0){
   // swap A[k] with A[m]
      temp = A[k];
      A[k] = A[m];
      A[m] = temp;
      k++;
   }
   m++;
}
```

Prove that "*the entries from A[0] to A[k-1] will all be zeroes (or RED)*" is a loop invariant for this code.

Solution

We need to prove that the invariant is true (1) before the while loop, (2) at the end of an arbitrary number of iterations of the while loop and (3) at the exit of the while loop.

Before entering the while loop

In this case, $k = 0$ and so $(k - 1)$ becomes -1. The invariant says "*the entries from A[0] to A[k-1] will all be zeroes (or RED)*". Since there is no array A[0..-1], the invariant is vacuously true.

At the end of p iterations, $1 \leq p \leq A.length$

Let the initial array contain q number of zeroes between 0 and p (i.e., $0 \leq q \leq p$). That is, there were q number of zeroes oved to the fonts of the array within this p iterations. In other words, the 'if' condition inside the loop will be true for q iterations of the loop.

In each of these q iterations, a RED ball is moved to the k^{th} position in the array. Since k started with zero and is incremented only when a RED ball is moved to the k^{th} position, it is clear that all RED balls are accumulated to the left of the array. Further, $k = q$ pointing to the position of the most recent RED ball that is moved to the left. Thus, all balls from zero to $k - 1$ will be RED.

At the exit of the while loop

The loop terminates when all elements of the array are scanned. At this time, k will still be pointing to the last RED ball that is moved to the left because the only time k is incremented is when a RED ball is moved. So, the invariant is true at the exit of the loop as well.

The proof can also consider the first iteration of the loop. In this case, there are two possibilities:
Case 1: The first ball is RED.
In this case, at the end of the first iteration, k is incremented by 1. So, all balls from 0 to $k - 1$ (i.e., all balls from 0 to 0) are RED.
Case 2: The first ball is BLUE.
In this case, the *if* block is skipped and hence k is not incremented (it still stays at 0). So, there is no $A[k - 1]$ and hence the invariant is true.