[LeetCode] First Bad Version - 二分查找

题目概述：

You are a product manager and currently leading a team to develop a new product. Unfortunately, the latest version of your product fails the quality check. Since each version is developed based on the previous version, all the versions after a bad version are also bad.

Suppose you have n versions [1, 2, ..., n] and you want to find out the first bad one, which causes all the following ones to be bad.

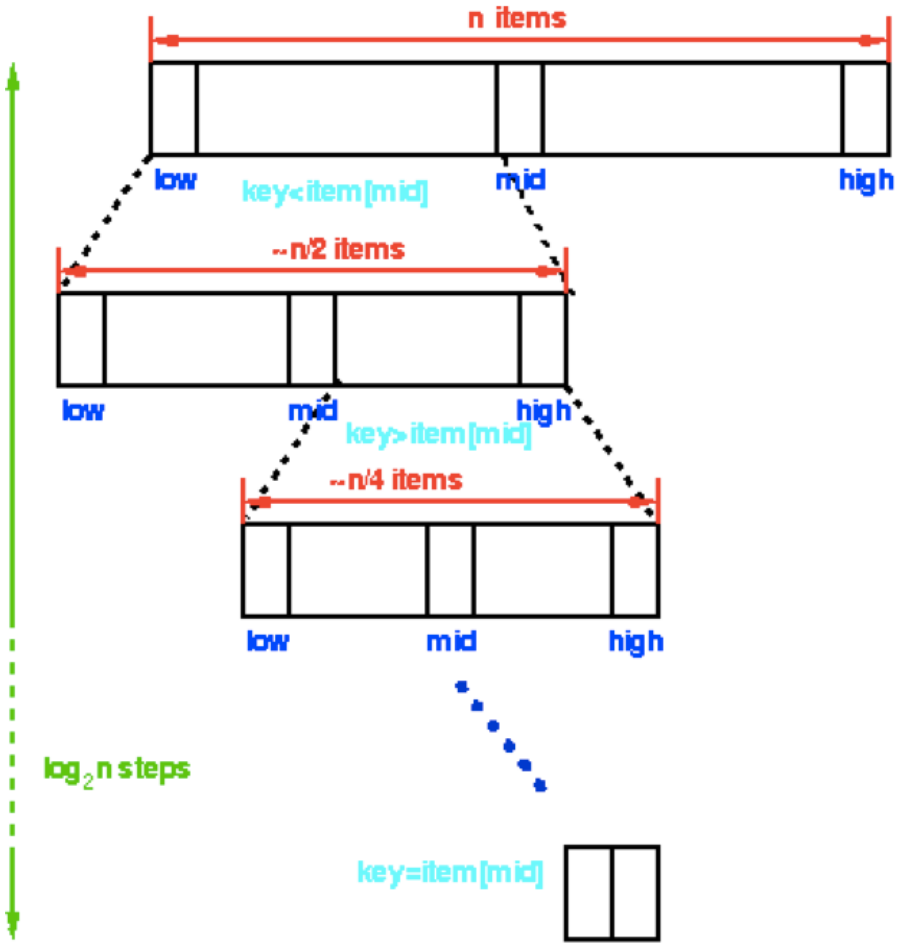
You are given an API boolisBadVersion(version) which will return whetherversion is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.

题目解析：

数组[1,2..n]中存在一个bad版本时，后面的版本都是bad，通过调用函数isBadVersion可以判断是否是bad版本。例如：[1,2,3]中2是bad版本，则调用isBadVersion(2)=true、isBadVersion(1)=false、isBadVersion(3)=true，结果返回2第一个导致bad的版本。

解决方法：二分查找

需注意middle=left+(right-left)/2、二分查找的下标移动和返回值left。



我的代码：

```
// Forward declaration of isBadVersion API.
bool isBadVersion(int version);

/*
 * 二分查找 关键步骤:
 * 1.middle定位
 * 2.大于middle查找右部分 left=middle+1
 * 3.小于middle查找左部分 right=middle-1
 */
int firstBadVersion(int n) {
    int middle;
    int left;
    int right;

    left=1;
    right=n;
    while(left<=right) {
        middle = left+(right-left)/2; //重点&能防止越界 例1+(5-1)/2=3
        if(isBadVersion(middle)==true) {
```