First Bad Version

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 `bool isBadVersion(version)` which returns whether `version` is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.

Example 1:

```
Input: n = 5, bad = 4

Output: 4

Explanation:

call isBadVersion(3) -> false

call isBadVersion(5) -> true

call isBadVersion(4) -> true

Then 4 is the first bad version.
```

Example 2:

```
Input: n = 1, bad = 1

Output: 1


Program :


/* The isBadVersion API is defined in the parent class VersionControl.

      boolean isBadVersion(int version); */


public class Solution extends VersionControl {

    public int firstBadVersion(int n) {

    int i = 1, j = n;
```

```
    while (i < j) {
        int m = i + (j-i) / 2;
        if (isBadVersion(m)) {
            j = m;
        } else {
            i = m+1;
        }
    }

    if (isBadVersion(i)) {
        return i;
    }

    return j;
}
}
```

Output :

Accepted

Runtime: 2 ms

Your input

```
5
4
```

Output

```
4
```

Diff

Expected

```
4
```

Constraints:

- $1 <= bad <= n <= 2^{31} - 1$