Given a string containing just the characters

'(' = 1

')' = 2

'{' = 3

'}' = 4

'[' = 5

']' = 6

determine if the input string is valid.

An input string is valid if:

Open brackets must be closed by the same type of brackets.

Open brackets must be closed in the correct order.

Note that an empty string is also considered valid.

Example 1:

Input: "()" = 12

Output: true

Example 2:

Input: "()[]{}" = 12 34 56

Output: true

Example 3:

Input: "(]" 15

Output: false

Example 4:

Input: "([)]" = 1526

Output: false

Example 5:

Input: "{[]}" = 3564

Output: true

Remap the symbols to numbers,

If the first number is even return false

If the last number is odd return false

If an odd number is found next to the odd++, remove it

Reevaluate the rest

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 will return whether the version is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.

Example:

Given n = 5, and version = 4 is the first bad version.

call isBadVersion(3) -> false

call isBadVersion(5) -> true

call isBadVersion(4) -> true

Then 4 is the first bad version.

We most likely will have more good versions at the beginning than bad versions at the end, hopefully, so I would start at the end.

i = n - 1

badVersion = 4

Version = [1,2,3,4,5]

while i >= 0

if versions[i] == badVersion

end