栈的压入与弹出序列是否一致\_剑指Offer\_31

# 栈的压入与弹出序列是否一致\_剑指Offer\_31

## 题目介绍

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*剑指Offer31:栈的压入、弹出序列\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 题目描述

\* 输入两个整数序列，第一个序列表示栈的压入顺序，请判断第二个序列是否可能为该栈的弹出顺序。

\* 假设压入栈的所有数字均不相等。例如序列1,2,3,4,5是某栈的压入顺序，

\* 序列4,5,3,2,1是该压栈序列对应的一个弹出序列，但4,3,5,1,2就不可能是该压栈序列的弹出序列。

\* （注意：这两个序列的长度是相等的）

## 思路介绍

\* 思路介绍：借助辅助栈，将第一个序列的元素顺序压入辅助栈中。

\* A. 若栈空或栈顶元素不是下一个待弹出元素，则继续执行入栈操作，直到入栈的数据就是下一个带弹出数字，入栈后直接跳出while循环；

\* B. 将栈顶元素出栈；

\* C. 若第一序列元素都入栈了，仍然没有找到下一个待弹出元素，直接返回false。

## Java代码

/\*\*

\* 剑指Offer31:栈的压入、弹出序列

\*/

public boolean **IsPopOrder**(int [] pushA,int [] popA) {

if(pushA == null||popA==null||pushA.length!=popA.length) return false;

if(pushA.length==0&&popA.length==0) return true;//长度都为0，默认为true

int pushIndex = 0,popIndex = 0,len = pushA.length;

ArrayDeque<Integer> stack = new ArrayDeque<Integer>();//辅助栈

while(pushIndex<len && popIndex < len){//pushIndex肯定在前，其实只要pushIndex>=len就可以了

//栈为空或栈顶元素不是下一个待弹出元素，则需要继续入栈操作；

// 直到刚压入的栈顶元素等于下一个待弹出元素为止；

// 压栈的过程中，要避免数组索引越界；若pushIndex已经到达len，

// 且栈顶还不是下一个待弹出元素，则直接返回false。

**while(stack.isEmpty()||stack.peek() != popA[popIndex]){**

**if(pushIndex >= len) return false;**

**stack.push(pushA[pushIndex++]);**

**}**

//出栈，栈非空且栈顶元素与下一个待弹出元素相等

**while(!stack.isEmpty()&&stack.peek()==popA[popIndex]){**

**stack.pop();**

**popIndex++;**

}

}

//popInex遍历完全返回true(由于push和pop元素个数相等，

// 因此popIndex等于len，肯定有pushIndex==len，且stack为空)

if(popIndex == len) return true;

else return false;

}