```swift
import Foundation
// 剑指 Offer 31. 栈的压入、弹出序列
// 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} 是某栈的压栈序列，
// 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。
//
// 输入：pushed = [1,2,3,4,5], popped = [4,5,3,2,1]
// 输出：true
// 解释：我们可以按以下顺序执行：
// push(1), push(2), push(3), push(4), pop() -> 4,
// push(5), pop() -> 5, pop() -> 3, pop() -> 2, pop() -> 1
class Solution {
    func validateStackSequences(_ pushed: [Int], _ popped: [Int]) -> Bool {
        var (stack, id): ([Int], Int) = ([], 0)
        for num in pushed {
            stack.append(num)
            while (!stack.isEmpty) && (stack.last! == popped[id]) {
                stack.popLast()
                id += 1
            }
        }
        return stack.isEmpty
    }
}

let solution = Solution()
print(solution.validateStackSequences([1,2,3,4,5], [4,5,3,2,1]))
print(solution.validateStackSequences([1,2,3,4,5], [4,3,5,1,2]))
print(solution.validateStackSequences([2,1,0], [2,1,0]))
print(solution.validateStackSequences([1,0,2], [2,1,0]))
print(solution.validateStackSequences([2,1,0], [1,2,0]))
print(solution.validateStackSequences([2,3,0,1], [0,3,2,1]))
```