Prompt | Scratchpad | Our Solution(s) | Video Explanation | Run Code

Solution 1

```swift
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
    // O(n) time | O(1) space
    func findThreeLargestNumbers(array: [Int]) -> [Int] {
        var threeLargest: [Int?] = [nil, nil, nil]

        for number in array {
            updateLargest(&threeLargest, number)
        }

        let threeLargestWithoutOptionals = threeLargest.compactMap { $0 }
        return threeLargestWithoutOptionals
    }

    func updateLargest(_ threeLargest: inout [Int?], _ number: Int) {
        if threeLargest[2] == nil {
            shiftAndupdate(&threeLargest, number, 2)
        } else if threeLargest[1] == nil {
            shiftAndupdate(&threeLargest, number, 1)
        } else if threeLargest[0] == nil {
            shiftAndupdate(&threeLargest, number, 0)
        }

        if let thirdNumber = threeLargest[2], number > thirdNumber {
            shiftAndupdate(&threeLargest, number, 2)
        } else if let secondNumber = threeLargest[1], number > secondNumber {
            shiftAndupdate(&threeLargest, number, 1)
        } else if let firstNumber = threeLargest[0], number > firstNumber {
            shiftAndupdate(&threeLargest, number, 0)
        }
    }

    func shiftAndupdate(_ threeLargest: inout [Int?], _ number: Int, _ index: Int)
        for i in 0 ... index {
            if i == index {
                threeLargest[i] = number
            } else {
                threeLargest[i] = threeLargest[i + 1]
            }
        }
    }
}

```

Your Solutions | Run Code

Solution 1 | Solution 2 | Solution 3

```swift
class Program {
    func findThreeLargestNumbers(array: [Int]) -> [Int] {
        // Write ypour code here.
        return []
    }
}

```

Custom Output | Raw Output | Submit Code

Run or submit code when you're ready.

Our Tests Your Tests Quick Test BETA