```
package main

import "fmt"

// FindOutlier function identifies the outlier in a list of integers
// An outlier is defined as the only even number in a list of odd numbers or vice
versa
func FindOutlier(integers []int) int {
        // Slices to store even and odd numbers separately
        evenNumbers := []int{}
        oddNumbers := []int{}

        // Iterate over each number in the input slice
        for _, number := range integers {
                if number%2 == 0 {
                        // Append to evenNumbers slice if the number is even
                        evenNumbers = append(evenNumbers, number)
                } else {
                        // Append to oddNumbers slice if the number is odd
                        oddNumbers = append(oddNumbers, number)
                }
        }

        // Determine which slice contains the outlier
        if len(evenNumbers) > len(oddNumbers) && len(oddNumbers) > 0 {
                // If there are more even numbers, the outlier is in the oddNumbers
slice
                return oddNumbers[0]
        } else if len(evenNumbers) < len(oddNumbers) && len(evenNumbers) > 0 {
                // If there are more odd numbers, the outlier is in the evenNumbers
slice
                return evenNumbers[0]
        } else {
                // If the counts are equal or no outlier is found, return 0 (no
clear outlier)
                return 0
        }
}

func main() {
        // Test cases
        a := []int{2, 4, 0, 100, 4, 11, 2602, 36} // One odd number (11) among even
numbers
        b := []int{160, 3, 1719, 19, 11, 13, -21} // One even number (160) among
odd numbers
        c := []int{15, 3, 1719, 19, 11, 13, -21}  // All odd numbers, no outlier

        // Print the results of the FindOutlier function for each test case
        fmt.Println(FindOutlier(a)) // Output: 11
        fmt.Println(FindOutlier(b)) // Output: 160
        fmt.Println(FindOutlier(c)) // Output: 0
}
```