

Our Solution(s)Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 package main
4
5 // O(n) time | O(n) space
6 func MaxSubsetSumNoAdjacent(array []int) int {
7     if len(array) == 0 {
8         return 0
9     } else if len(array) == 1 {
10        return array[0]
11    }
12    maxSums := make([]int, len(array))
13    maxSums[0], maxSums[1] = array[0], max(array[0], array[1])
14    for i := 2; i < len(array); i++ {
15        maxSums[i] = max(maxSums[i-1], maxSums[i-2]+array[i])
16    }
17    return maxSums[len(array)-1]
18 }
19
20 func max(a, b int) int {
21     if a > b {
22         return a
23     }
24     return b
25 }
26
```

Your SolutionsRun Code

Solution 1Solution 2Solution 3

```
1 package main
2
3 func MaxSubsetSumNoAdjacent(array []int) int {
4     // Write your code here.
5     return -1
6 }
7
```

Custom OutputSubmit Code

```
18 report()
19
20 "gettable: use the table to find the frequency of reports"
21
22
23 # Run the first function: find the frequency of reports
24 freq <- findFrequencyOfReports(2000, 10)
25 reportFrequency(freq, 10, 20)
26
27
28 # Run the second function: find the frequency of reports
29 freq <- findFrequencyOfReports(2000, 10, 20)
30 reportFrequency(freq, 10, 20)
31
32
33 # Run the third function: find the frequency of reports
34 freq <- findFrequencyOfReports(2000, 10, 20, 20)
35 reportFrequency(freq, 10, 20)
36
```

Run or submit code when you're ready.