CS301 Programming Competition Problems
March 18, 2021

1. Write a function `divisibleSumPairs` that takes an array of integers and a positive integer k, determines the number of `(i,j)` pairs where `i<j` and `arr[i]+arr[j]` is divisible by k.
Example:

Input	Output	Explanation

ar = [1,2,3,4,5,6]
k= 5
	3	Three indices' pairs meet the criteria: (0,3), (1,2), and (3,5), hence the result is 3.

ar = [1, 3, 2, 6, 1, 2]
k=3
	5	Five indices' pairs meet the criteria: (0,2), (0,5), (1,3), (2,4) and (4,5), hence the result is 5

2. Write a function `mostFrequentSight`, that takes an array of bird sightings where every element represents a bird type id. It determines the id of the most frequently sighted type. If more than 1 type of birds are most frequent, return the smallest of their ids.

Input	Output	Explanation
[1,1,2,2,3]	1	There are two each of types 1 and 2, and one sighting of type 3. Pick the lower of the two types seen twice: type 1
[1,4,4,4,5,3]	4	Type 4 is sighted most

3. There is a large pile of socks that must be paired by color. Write a function `countSocksPairs`, that given an array of integers representing the color of each sock, determines how many pairs of socks with matching colors there are.

Input	Output	Explanation
[1,2,1,2,1,3,2]	2	There is one pair of color 1 and one of color 2
[10,20,20,10,10,30,50,10,20]	3	Two pairs of color 10 and one pair of color 20

4. The Utopian Tree goes through 2 cycles of growth every year. Each spring, it doubles in height. Each summer, its height increases by 1 meter.

Now, write a function `computeHeight(height, cycles)`, that returns how tall the tree will be after n growth cycles given the original height when planted.

For example, if the height of the tree when planted is 1 and number of growth cycles is n=4, result should be 7 based on following computation.

Growth cycles	Height	Growth cycle
0	1	Initial planted height
1	2	Spring growth (double)
2	3	Summer growth (plus 1)
3	6	Spring growth (double)
4	7	Summer growth (plus 1)

i.e. `computeHeight(1,4)` → `7`

5. Write a function, computeCharges, that will accept an array that contains objects with properties for the customer Id and charges. Your function should return an array that contains objects with the customer Id, average charge, and maximam charge.
Hint: write helper functions to getMax and getAverage of an array of charges to keep your code organized.

```
const charges = [{custId: 1,charges:[5,7,3]}, {custId: 2, charges: [20,60,50,30]}]

computeCharges(charges)→
   [{custId: 1, average: 5, maximum: 7}, {custId: 2, average: 40, maximum: 60}]
```

6. The distance between two array values is the number of indices between them. Write a function `findMinDistance`, that given an array finds the minimum distance between matching pairs of elements in the array. If no such value exists, return -1.

Input	Output	Explanation
[3,2,1,2,3]	2	There are two matching pairs of value: 3 and 2. Distance between indices of value 3 is 4 and distance between indices of value 2 is 2 and 2 is the minimum.
[7,1,3,4,1,7]	3	There are two matching pairs for values: 7 and 1. Distance between indices of value 7 is 5 and distance between indices of value 1 is 3 and 3 is the minimum.
[1,2,3,4]	-1	No such value exists