QN Link : <https://www.desiqna.in/16820/amazon-oa-sde-ctc-45lpa-coding-questions-2024-set-141>

Observation :

* We va given that , we can deliver only 3 packages or 2 packages at a time.All numbers are comes under 3 & 2 except 1 .
* So when a number is occurred 1 time , return -1.

1 🡪 -1 6 🡪 2 times (3 + 3) 11 🡪 4 times ( 2 + 3 + 3+ 3)

2 🡪 1 time 7 🡪 3 times(2 + 2 + 3) 12 🡪 4 times (3 + 3+ 3 + 3)

3 🡪 1 time 8 🡪 3 times (3 + 3 + 2) 13 🡪 5 times (3 + 3 + 3 + 4)

4 🡪 2 times (2 + 2) 9 🡪 3 times (3 + 3 + 3) 14 🡪 5 times (12 + 2)

5 -> 2 times (3 + 2) 10 🡪4 times (2 + 2+ 4 + 4) 15 🡪 5 times (5 \* 3 )

So we can observe a pattern that , for each 3 number , the result is same

4 % 3 == 0 🡪 No 4 / 3 + 1 = 2

5 % 3 == 0 🡪 No 5 / 3 + 1 = 2

6 % 3 == 0 🡪 Yes 6/ 3 = 2

So this is the pattern , divide the frequency according to the above observation.

Step 1 : Init a hashMap and a variable min

Step 2 : Get the count of all elements in the array

Step 3 : For each element in the map , get the freq and check it is modulo by 3

Step 4 : if so , then add min += (freq / 3)

Step 5 : Else add min += (freq / 3 + 1)

class Solution {

    public int minimumTime(int [] nums) {

        Map<Character , Integer> map = new HashMap<>();

        int min = Integer.MAX\_VALUE;

        for(int c : nums){

            map.put(c , map.getOrDefault(c , 0) + 1);

        }

        for(int c : map.keySet()){

            int val = map.get(c);

            if(val == 1) return -1;

           if(val % 3 == 0)

            min += (val / 3);

            else

            min += (val / 3 + 1);

        }

        return min;

    }

}