1). Find the Duplicate Number

class Solution {

    public int findDuplicate(int[] nums) {

        int len=nums.length,s=0;

        HashSet<Integer> set = new HashSet<>();

        for(int i=0;i<len;i++){

            if(!set.add(nums[i])){

                s=nums[i];

            }

        }

        return s;

    }

}

2).  Number of Good Pairs

class Solution {

    public int numIdenticalPairs(int[] nums) {

        int count=0;

        for(int i=0;i<nums.length;i++){

            for(int j=i+1;j<nums.length;j++){

                if(nums[i] == nums[j]){

                    count++;

                }

            }

        }

        return count;

    }

}

3).  Find Greatest Common Divisor of Array

class Solution {

    public int findGCD(int[] nums) {

        Arrays.sort(nums);

        int gcd=1,len=nums.length;

        int small=nums[0],larg=nums[len-1];

        for(int i=1;i<=small && i<=larg;i++){

            if(small%i==0 && larg%i==0){

                gcd=i;

            }

        }

        return gcd;

    }

}

4). Unique Number of Occurrences

class Solution {

    public boolean uniqueOccurrences(int[] arr) {

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

       for(int i:arr){

           if(!map.containsKey(i)){

               map.put(i,1);

           }

           else{

               map.put(i,map.get(i)+1);

           }

       }

       HashSet<Integer> set = new HashSet<>(map.values());

       int min=map.size();

       int max=set.size();

       if(min==max){

           return true;

       }

       return false;

    }

}

5).Find All Duplicates in an Array

class Solution {

    public List<Integer> findDuplicates(int[] nums) {

        List<Integer> l1 = new ArrayList<>();

        Arrays.sort(nums);

        for(int i = 1; i < nums.length ; i++){

            if(nums[i]==nums[i-1]){

                l1.add(nums[i]);

            }

        }

        return l1;

    }

}