1. Given a binary array `nums`, return *the maximum number of consecutive* `1`*'s in the array*.

```
Input: nums = [1,1,0,1,1,1]
Output: 3


Input: nums = [1,0,1,1,0,1]

Output: 2
```

Constraints:

- `1 <= nums.length <= 105`
- `nums[i]` is either `0` or `1`.

2. Given an array `nums` of integers, return how many of them contain an even number of digits.

```
Input: nums = [12,345,2,6,7896]

Output: 2

Input: nums = [555,901,482,1771]

Output: 1
```

Constraints:

- `1 <= nums.length <= 500`
- `1 <= nums[i] <= 10^5`

3. Given an integer array `nums` sorted in non-decreasing order, return *an array of the squares of each number sorted in non-decreasing order*.

   ```
   Input: nums = [-4,-1,0,3,10]

   Output: [0,1,9,16,100]

   Input: nums = [-7,-3,2,3,11]

   Output: [4,9,9,49,121]
   ```

   Constraints:

   - `1 <= nums.length <= 104`
   - `-104 <= nums[i] <= 104`
   - `nums` is sorted in non-decreasing order.

4. Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to `target`*.

   You may assume that each input would have *exactly* one solution, and you may not use the *same* element twice.

   You can return the answer in any order.

   ```
   Input: nums = [2,7,11,15], target = 9

   Output: [0,1]

   Input: nums = [3,2,4], target = 6

   Output: [1,2]
   ```

Constraints:

- $2 <= nums.length <= 10^3$
- $-10^9 <= nums[i] <= 10^9$
- $-10^9 <= target <= 10^9$
- Only one valid answer exists.

Solution using hashmap

```
public int[] twoSum(int[] nums, int target) {

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

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

        map.put(nums[i], i);

    }

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

        int complement = target - nums[i];

        if (map.containsKey(complement) && map.get(complement)
!= i) {

            return new int[] { i, map.get(complement) };

        }

    }
    throw new IllegalArgumentException("No two sum solution");
}
```

5. Factor Game

For a given list of numbers, find its factors and add the factors. Then if the sum of factors is present in the original list, sort it and print it else print -1.

Input :

First Input : List of numbers

Output : List of factors

Sample Testcases :

I/P 1:

1,2,4,7

O/P 1:

[1,4]

I/P 2:

2,4

O/P 2:

-1

6. **Pronic Number**

Given a string of random numbers, your job is to find the product of the numbers(one is lesser and one is greater) who is already present in the string. For instance, a pronic number is a number which is the product of two consecutive integers, that is, a number of the form n(n + 1). Like 6 is the pronic number as 2*3 = 6.

Input :

First Input : String of random numbers

Output : List of pronic numbers

Sample Testcases :

I/P 1:

123456

O/P 1:

[2,6,12]

I/P 2:

4567

O/P 2:

-1

7. **Even Odd Series**

Given a string and it contains the digits as well as non-digits. We have to find the largest even number from available digits after removing the duplicates. If not possible, print -1.

Input :

First Input : String

Output :

Largest number

Sample Testcases :

I/P 1:

%#32%#%2

```
O/P 1:

32


I/P 2:

%#2373#@

O/P 2:

732
```

8. **String rotation in special manner**

You are provided two or more strings, where each string is associated with the number

(seperated by :). If sum of square of digits is even then rotate the string right by one

position, and if sum of square of digits is odd then rotate the string left by two position.

Input :

Accept multiple values in the form of String:Integer seperated by ‘,’ Output :

Rotated Strings.

Sample Testcases :

I/P 1 :

```
abcde:234,pqrs:246
```

O/P 1 :

```
cdeab
spqr
```

Explanation :

For first teststring, ‘abcde’ associated integer is 234, squaring 4+9+16=29, which is odd, so we rotate string left by two positions. For second teststring, ‘spqr’ associated integer is 246, squaring 4+16+36=56, which is even, so we rotate string right by one position.

9. Word Manipulation

Consider a string, group the similar characters in combinations. Then, concatenate first element and last element alternatively. For instance, Consider a string “HelLoWOrld”, combinations of similar characters will

be : [‘d’, ‘e’, ‘H’, ‘lLl’, ‘oO’, ‘r’, ‘W’]

So, final output : dWerHoOlLl

Input :

First Input : s, string

Output :

String [Manipulated]

Sample Testcases :

I/P 1:

HelLoWOrld

O/P 1 : dWerHoOlLl

10. Great Sum

Given a special set of numbers and a special sum. Find all those combinations from the given set, equaling the given sum.

Input :

First Input : Set of numbers

Second Input : Special Sum

Output :

Combinations satisfying criteria

Sample Testcases :

I/P 1:

-1, 1, 0, 0, 2, -2

0

O/P 1

3

Explanation : Following combinations are satisfying (-1,1,2,-2), (0, 0, 1, -1), (0, 0, -2, 2)

Note : Make combinations with 4 integers only.

11. Palindromic String

You are provided with a string of numbers, check whether it is a palindrome or not. If it is not a palindrome, then, reverse the string, add it to the original string and check again. You are required to repeat the process until it becomes palindrome. Find the length of palindromic string.

Input :

First Input : String

Output : Length of Palindrome

Sample Testcases :

I/P 1:

1

O/P 1:

1

I/P 2:

145

O/P 2:

3

Explanation :

Given string is 145, it is not a palindrome. Reversing and adding, 145+541 = 686, Hence it is a palindrome.

12. **4 Digit OTP**

You will be given a number in the form of string, extract out digits at odd places, square & merge them. First 4 digits will be the required OTP.

First Input : String

Output : 4 digit OTP

Sample Testcases :

I/P 1:

34567

O/P 1:

1636

Input :

First Input : String

```
Output : 4 digit OTP
```