Parity Sorting

QNo 1: Given an integer array. Sort the array in such a way that all the odd number are in the beginning of the array followed by the even numbers.

- The odd number and even number should follow an ascending order.

Input

Input Format

The first line contains the number of integers N

The second line contains the elements of the array

Constraints

1<=N<=10^4

Output

Print the final array.

Sample Input 1

25142

Sample Output 1

15224

Qno 2: **NEXT Greater Number**

Given an array of N elements, find the next greater element for each element in the array, print -1 if it does not exist. Refer to the sample I/O for better understanding

Input

The first line contains T. the number of test cases.

The first line of each test case contains N, the number of elements in the array. The next line contains N space separated integers denoting the elements of the array

Constraints

1<T<<10

1 <=N<= 10^5

0 <=A[i] <= 10000

Sample Input 1 and 2

1. 1,3,4,4

2. 5,1,3,4,4

Sample Output 1 and 2

344-1

-1344-1

Hint

In the sample test case, the array contains 4 elements, and for the first element, the next greater element will be 3, similarly for the next element, the next greater element will be 4, a nd for the third element as well, the next greater element will be 4. Since, the last element has no elements ahead of it, the next greater element will be -1.

Qn4:

you need to create a C++ program that takes the document as input and extracts the longest word present.

Input:

A string 'sentence' that contains the text document. The document can contain a variety of characters, including letters, digits, punctuation, spaces, and line breaks. The document's length does not exceed 10,000 characters.

Output:

A string 'longestWord' that represents the longest word found within the document.

input

documentText = "In the year 2022, technology is advancing rapidly."

longestWord = "technology"

SOLUTION …..CALL FUNCTION IN MAIN

```
string findLongestWord(const string& documentText) {
   string word, longestWord;
   int startPos = 0;

   for (int i = 0; i <= documentText.length(); i++) {
      if (i == documentText.length() || documentText[i] == ' ' || documentText[i] == '\n') {
         string currentWord = documentText.substr(startPos, i - startPos);
         if (currentWord.length() > longestWord.length()) {
            longestWord = currentWord;
         }
         startPos = i + 1;
      }
```

```
}
```

Q 5 You are given an array of integers and a target sum 'S'. The program aims to find and count pairs of array elements whose sum is greater than the target sum.

```
#include<iostream>
 using namespace std;
 int main()
 {
     int n;
     cout<<"Enter the value of n:";
     cin>>n;
     int arr[n];
     for(int i=0;i<n;i++)
     {

         cin>>arr[i];
     }
     cout<<"Array elements is:";

    for(int i=0;i<n;i++)
    {
        cout<<arr[i]<<" ";

    }
    int S;
    int count=0;
    cout<<endl;
    cout<<"Enter the sum:";
    cin>>S;
    for(int i=0;i<n;i++)
    {
    for(int j=i+1;j<n;j++)
    {
        if(arr[i]+arr[j]>S)
```

```
        count++;
    }
    }
    cout<<"Count is: "<<count;
}
```

Q N0 6

Sure, let's consider a scenario from a different context. Imagine you have a large circular racetrack, and there are runners on it, each at different positions. You want to synchronize the timing of these runners because there's a timing error. The clock shows that they start running at 2:00 PM, but it's supposed to be 3:00 PM.

To fix this timing issue, you decide to rotate the clock hands one hour to the right (k = 1). So, when the clock originally points at 12, indicating 12:00 PM, after the rotation, it correctly shows 1:00 PM.

This rotation should be circular, just like a clock, where after 12, it wraps around to 1. By applying this rotation, you Initial GPS Data: [10:00 AM, 11:00 AM, 12:00 PM, 1:00 PM, 2:00 PM]

Rotation Steps (k): 1

Expected Result: [2:00 PM, 10:00 AM, 11:00 AM, 12:00 PM, 1:00 PM] ensure that the timing is accurate for the runners on the racetrack.

Q NO 7:

Imagine you are running an online store, and you want to analyze customer purchase data. You have a list of product prices in your store, and you want to find out how many pairs of products are bought together by customers that add up to a specific total purchase amount, like a promotional deal or a customer's budget. Each product has its price, and you aim to calculate the number of product pairs that, when purchased together, meet the given total purchase amount

Product Prices: [5, 10, 15, 20]

Total Cost: 30

Expected Output: 2 (Pairs: [5, 15] and [10, 20])

Hint

```
int countPairsWithSum(int arr[], int n, int targetSum) {
   sort(arr, arr + n); // Sort the array
```

```
int left = 0;
int right = n - 1;
int count = 0;

while (left < right) {
    int currentSum = arr[left] + arr[right];

    if (currentSum == targetSum) {
        count++;
        left++;
        right--;
    } else if (currentSum < targetSum) {
        left++;
    } else {
        right--;
```

Qno 7

Your task is to write a C++ program that takes a string containing employee titles and capitalizes the first letter of each word, ensuring uniformity in title formatting.

Input: how are You

Output How are You

Hint

```
for (char c : input) {
    if (newWord && isalpha(c)) {
        formattedString += toupper(c);
        newWord = false;
    } else {
        formattedString += c;
    }

    if (c == ' ') {
        newWord = true;
```

```
        }
    }


    return formattedString;
}
```

Qno 7:

Imagine you work in a large retail store where stock audits are performed regularly. During these audits, two lists of inventory items are compared: one representing the items currently in stock and the other representing the items recorded in the inventory database. Your task is to efficiently identify any missing inventory items that are in the database but not found in the physical stock

Stock List: [1, 2, 3, 4, 5]

Database List: [3, 4, 5, 6, 7]

Missing Inventory Items: [1, 2]