Sort an array of 0s 1s and 2s

- Given an array **A[]** consisting 0s, 1s and 2s. The task is to write a function that sorts the given array. The functions should put all 0s first, then all 1s and all 2s in last.

```
Input: {0, 1, 2, 0, 1, 2}
Output: {0, 0, 1, 1, 2, 2}

Input: {0, 1, 1, 0, 1, 2, 1, 2, 0, 0, 0, 1}
Output: {0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2}
```

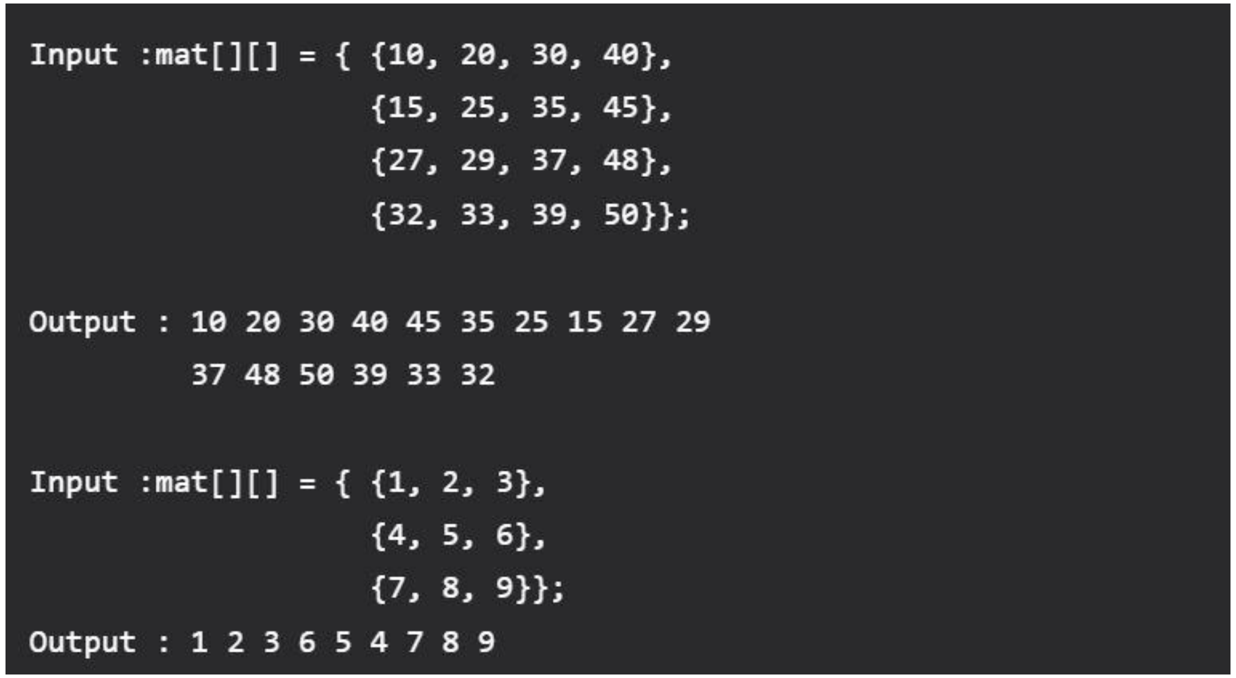
Merge Overlapping Intervals

- Given a set of time intervals in any order, merge all overlapping intervals into one and output the result which should have only mutually exclusive intervals. Let the intervals be represented as pairs of integers for simplicity.

For example, let the given set of intervals be {{1,3}, {2,4}, {5,7}, {6,8}}. The intervals {1,3} and {2,4} overlap with each other, so they should be merged and become {1, 4}. Similarly, {5, 7} and {6, 8} should be merged and become {5, 8}

Snake Pattern

- Given an n x n matrix .In the given matrix, you have to print the elements of the matrix in the snake pattern.



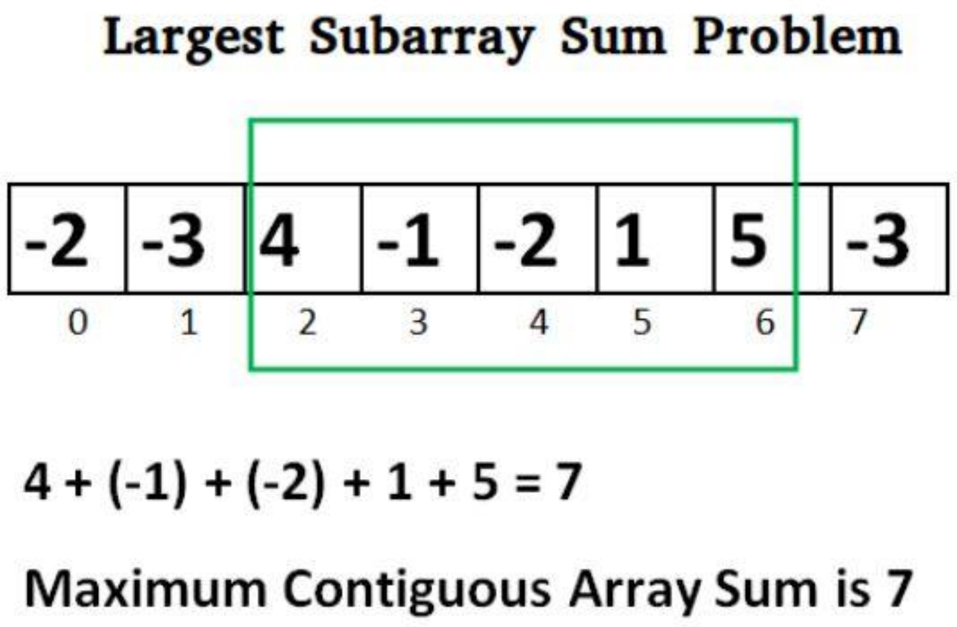
```
Input :mat[][] = { {10, 20, 30, 40},
                   {15, 25, 35, 45},
                   {27, 29, 37, 48},
                   {32, 33, 39, 50}};

Output : 10 20 30 40 45 35 25 15 27 29
         37 48 50 39 33 32

Input :mat[][] = { {1, 2, 3},
                   {4, 5, 6},
                   {7, 8, 9}};
Output : 1 2 3 6 5 4 7 8 9
```

Largest Sum Contiguous Subarray

- Write an efficient program to find the sum of contiguous subarray within a one-dimensional array of numbers that has the largest sum.



Largest Subarray Sum Problem
-2
-3
4
-1
-2
1
5
-3
0
1
2
3
4
5
6
7
4 + (-1) + (-2) + 1 + 5 = 7
Maximum Contiguous Array Sum is 7

Majority Element

- Write a function which takes an array and prints the majority element (if it exists), otherwise prints “No Majority Element”. A ***majority element*** in an array A[] of size n is an element that appears more than n/2 times (and hence there is at most one such element).

```
Input : {3, 3, 4, 2, 4, 4, 2, 4, 4}
Output : 4
Explanation: The frequency of 4 is 5 which is greater than the half of the size of the
array size.

Input : {3, 3, 4, 2, 4, 4, 2, 4}
Output : No Majority Element
```

Minimum number of chairs required to ensure that every worker is seated at any instant

Given a string **S** representing the record of workers entering and leaving the rest area, where **E** represents entering and **L** represents leaving the rest area. For each worker, one chair is required. The task is to find the minimum number of chairs required so that there is no shortage of chairs at any given time.

Examples:
Input: S = “EELEE”
Output: 3
Input: S = “EL”
Output: 1

Move all negative numbers to beginning and positive to end with constant extra space

An array contains both positive and negative numbers in random order. Rearrange the array elements so that all negative numbers appear before all positive numbers.

Examples :

```
Input: -12, 11, -13, -5, 6, -7, 5, -3, -6
Output: -12 -13 -5 -7 -3 -6 11 6 5
```

N/3 repeated number in an array with O(1) space

We are given a read only array of n integers. Find any element that appears more than n/3 times in the array in linear time and constant additional space. If no such element exists, return -1.

Examples:

```
Input : [10, 10, 20, 30, 10, 10]
Output : 10
10 occurs 4 times which is more than 6/3.

Input : [20, 30, 10, 10, 5, 4, 20, 1, 2]
Output : -1
```