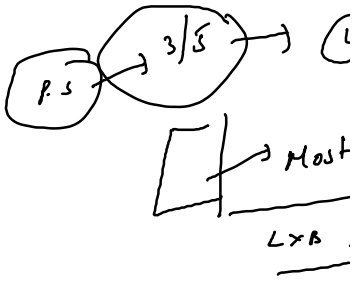
Day 02

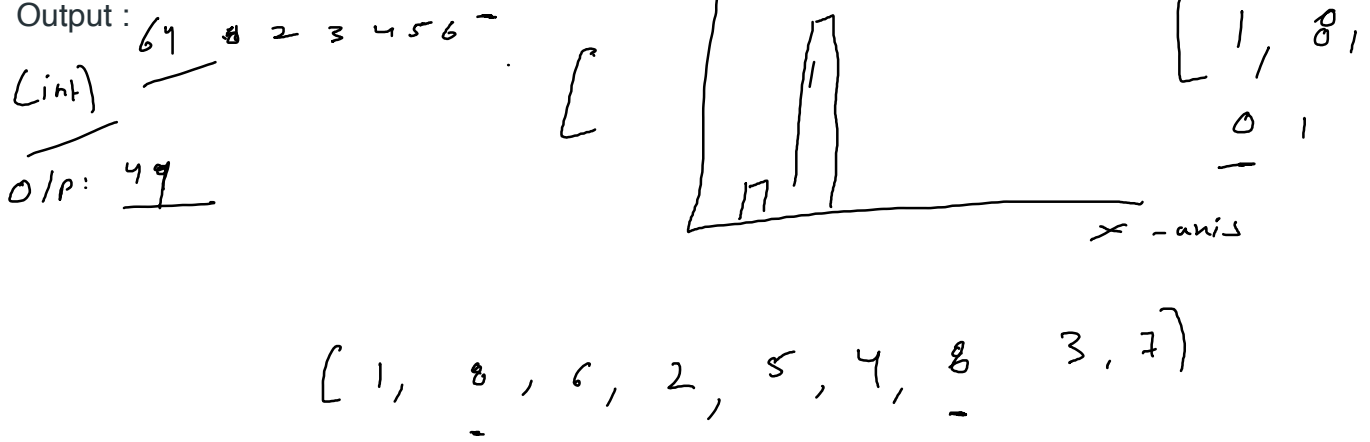
Wednesday, 30 December 2020 7:20 PM

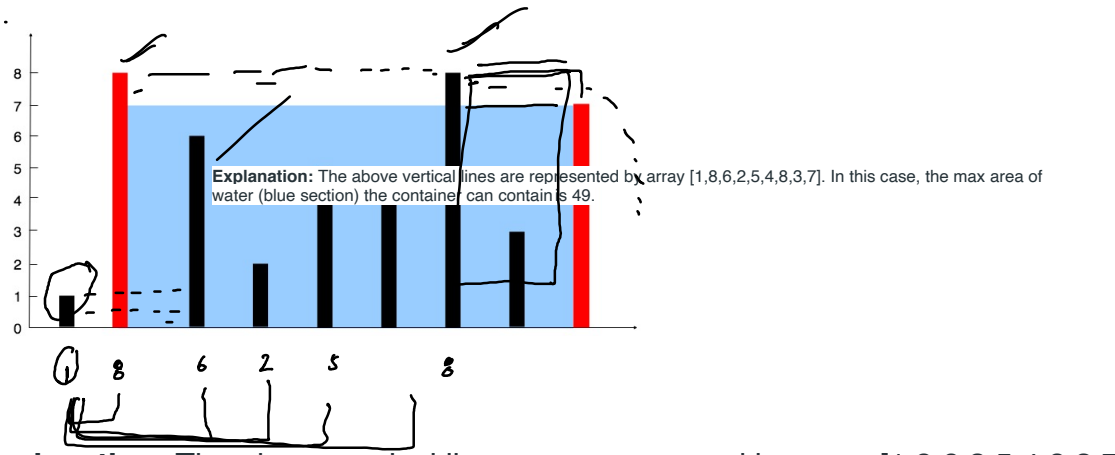


Problem Statement :

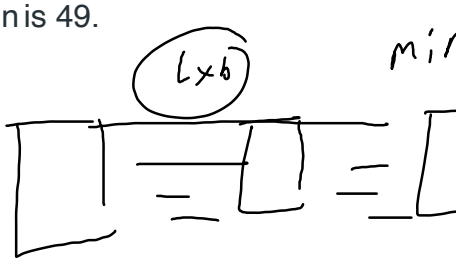
Given n non-negative integers a1, a2, ..., an , where each represents a point at coordinate (i, ai). n lines are drawn such that the two endpoints of the line i is at (i, ai) and (i, 0). Find two lines, which with the x-axis forms a container, such that the container contains the most water.

Input: height = [1,8,6,2,5,4,8,3,7]
Output :

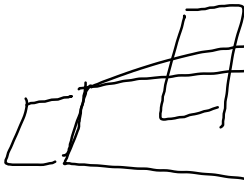




Explanation: The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this cas the max area of water (blue section) the container can contain is 49.



```
int maxArea(int[] A, int len)
{
    int area = 0;
    for(int i = 0; i < len; i++)
    {
```



```
for( int j = i+1;  j < len;  j++)
{
        // Calculate the  max Area
        area =  max(area, min(A[i], A[j])*(
}
return area;
}
```

$O(n^2)$

↳ $O(n)$:-

Home Work Assignments :

https://leetcode.com/problems/container-with-most-water/
https://leetcode.com/problems/trapping-rain-water/
https://leetcode.com/problems/maximum-subarray/
Find second maximum value in the array
https://leetcode.com/problems/rotate-array/
Difference between HashMap vs HashSet
https://leetcode.com/problems/move-zeroes/

Map.Put(key , map.getordefault(key , 0) + 1)

Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.

Input: height = [0,1,0,2,1,0,1,3,2,1,2,1]

The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.

Find the second maximum value in an Array :

Arr : [9 , 2 , 3 ,6]

Arr: [6 , 2 , 3 , 9]

```
Public static int findSecondMaxium(int[] arr)
{

    Int max = Integer.Min_Value;
    Int secondmax = Integer.Min_Value;

    For(int I = 0 ; I < arr.length; i++)
    {
        If(arr[i] > max)
        {
            max = arr[i];
        }
    }

    For(int I = 0; I < arr.length; i++)
    {

        If(arr[i] > secondmax && arr[i] < max)
        {
          secondmax = arr[i];
        }
    }

    Return secondmax;
```

```
}


For(int I = 0 ; I < arr.length; i++)
{
    If(arr[i] > max)
    {
        secondMax = max;
        max = arr[i];
    }

    Else if ( arr[i] > secondMax && arr[i] ! = max)
    {
     secondMax = arr[i]'
    }
}
```

Rotate Array
Given an array, rotate the array to the right by *k* steps, where *k* is non-negative.
Input: nums = [1,2,3,4,5,6,7], k = 3

Output : [5 , 6 7 , 1 , 2 , 3 , 4]

Brute Force Approach :

Moving Zeros :

Given an array nums, write a function to move all 0's to the end of it while maintaining the relative order of the non-zero elements.

Example:
Input: [0,1,0,3,12]
Output: [1,3,12,0,0]

```
public void moveZeroes(int[] nums) {


 }
```