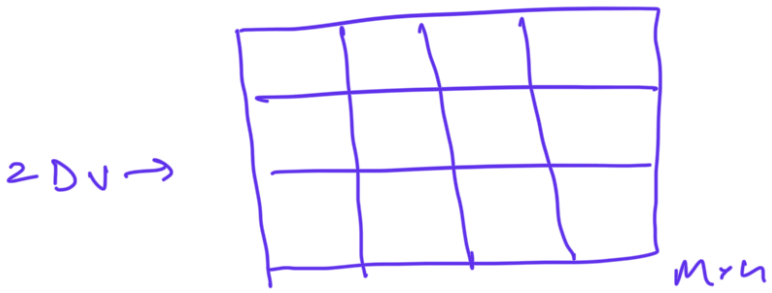
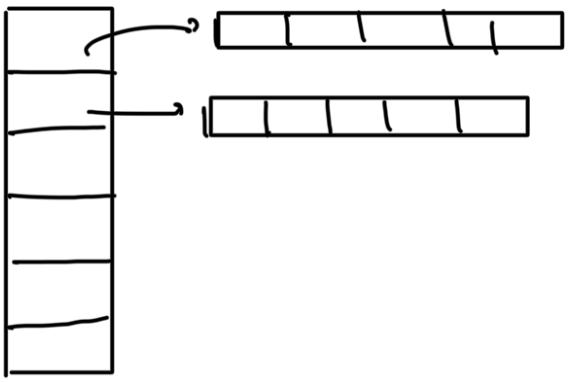
2-D Vector:



Initialization of 2-D vector:

```
vector<vector<int>> name;
```

Memory Representation:



Initialization of 2-D vector for an input of Size;

```
vector<vector<int>> arr(5, vector<int>(10,0))
```

- 2D vector
- Name of Array.
- Size of Row
- Size of Col
- Initial value

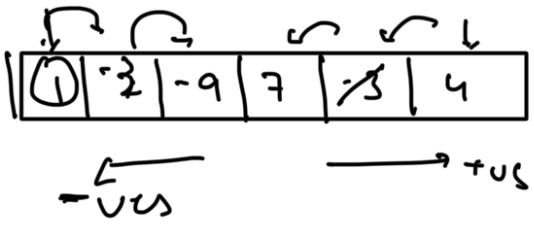
10 You have given an array of you are

Supposed to sort the given array in such a way that -ve elements arrive before +ves.

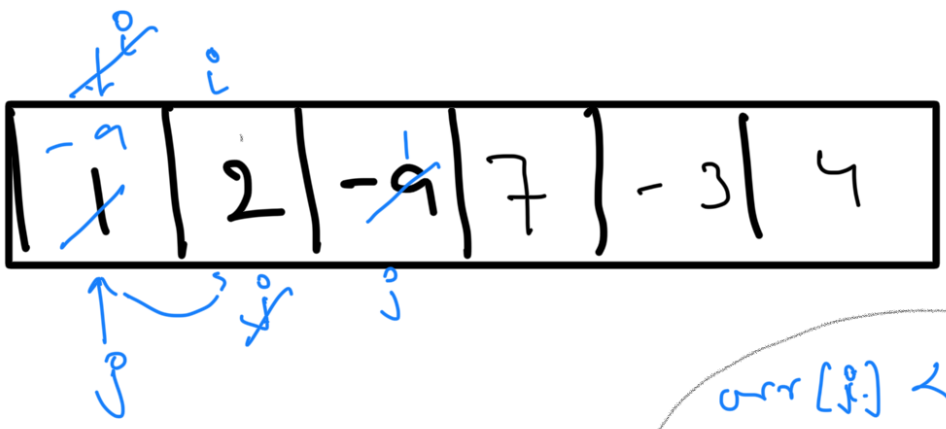
1	2	-9	7	-3	4

Sol^n.

Problem Statement:



Approach.



arr[j] < 0

j++

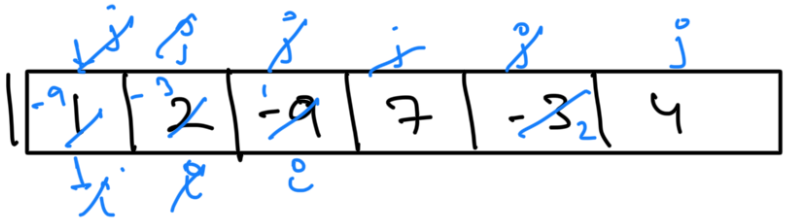
swap i → j

Condition →

```
if [arr[j] <0) {
        Swap (arr[i], arr[j]);
        i++;
        j++;
}
else {
        j++
```

3.

Dry Run:



i = 0 , j = 0;

j < 0

i = 0 ; j = 1;

j < 0

i = 0 ; j = 2

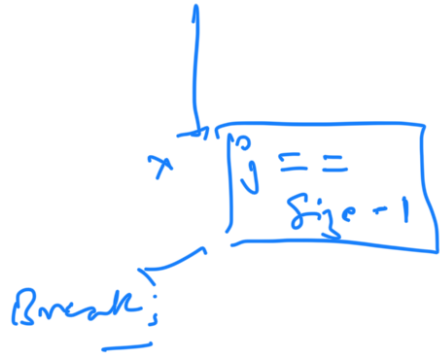
j < 0 ✓ , swap. , i++ , j++

i = 1 , j = 3

j < 0 ✗

i = 1 , j = 4

j < 0 ✓



Time Complexity:

Size of Array = n

n

n

→ O(2n) ✓

n

CODE.

06_26.cpp > main()

```cpp
#include<iostream>
#include<vector>
using namespace std;
int main(){
    vector<int>arr={-1,-2,-2,-1,-5,-3,-8};
    int i=0;
    int j=0;
    while(j<arr.size() && i<arr.size()){
        if(arr[i]<0 && i==j) i++,j++;
        if(arr[j]<0){
            swap(arr[i],arr[j]);
            i++;
            j++;
        }
        else{
            j++;
        }
    }
    //print the array;
    for(int i=0;i<arr.size();i++){
        cout<<arr[i]<<" ";
    }
    return 0;
}
```