Problem:

> Given two integer arrays Arr1 and Arr2 of size N. Use the greatest elements from the given arrays to create a new array of size N such that it consists of only unique elements and the sum of all its elements is maximum.
> The created elements should contain the elements of Arr2 followed by elements of Arr1 in order of their appearance.

Example:

Input:

```
N = 5
Arr1 = {7, 4, 8, 0, 1}
Arr2 = {9, 7, 2, 3, 6}
```

Output:

```
9 7 6 4 8
```

Explaination:

```
9, 7, 6 are from 2nd array
and 4, 8 from 1st array.
```

Approach:

⟶ here first we will add all elements in `set` and then traverse the set from the backwards because it will be sorted so last elements will be big. So we will mark them taken as `True` and also we will make one `count` variable because we will only mark `n` variables from last as taken.

Code will look like this:

```
            set<int>s;
        for(int i=0;i<n;i++){
            s.insert(arr2[i]);
            s.insert(arr1[i]);
        }
        vector<int>ans; // It will contain our ans
        unordered_map<int,bool>mp; // To mark the elements as
taken
        set<int>::reverse_iterator sitr;
        int count = 0; // we only need n elements in our array
            //Traversing set from backwards
        for(sitr=s.rbegin();sitr!=s.rend() && count < n;sitr++){
            count++;
            mp[*sitr] = true;
        }
```

⟶ Now we will traverse both arrays and we will add the elements in `ans` array which are marked as `taken`

we know that if for any `elem`, `mp[elem]` is true then it means it's marked as taken.

```
            for(int i=0;i<n;i++){
                    // If arr2[i] is in map then only we will
proceed.
            auto it = mp.find(arr2[i]);
            if(it != mp.end() && it->second){
                ans.push_back(arr2[i]);
                            // we have to mark taken as false
because we don't want to take it again when we traverse first
array.
                it->second = false;
            }
        }
        for(int i=0;i<n;i++){
            auto it = mp.find(arr1[i]);
```

```
            if(it != mp.end() && it->second){
                ans.push_back(arr1[i]);
                it->second = false;
            }
        }
```

At last we will return `ans`