Problem statement:

> Given two unsorted arrays **A** of size **N** and **B** of size **M** of distinct elements, the task is to find all pairs from both arrays whose sum is equal to **X**.

Example:

Input:

```
A[] = {1, 2, 4, 5, 7}
B[] = {5, 6, 3, 4, 8}
X = 9
```

Output:

```
1 8
4 5
5 4
```

Explanation:

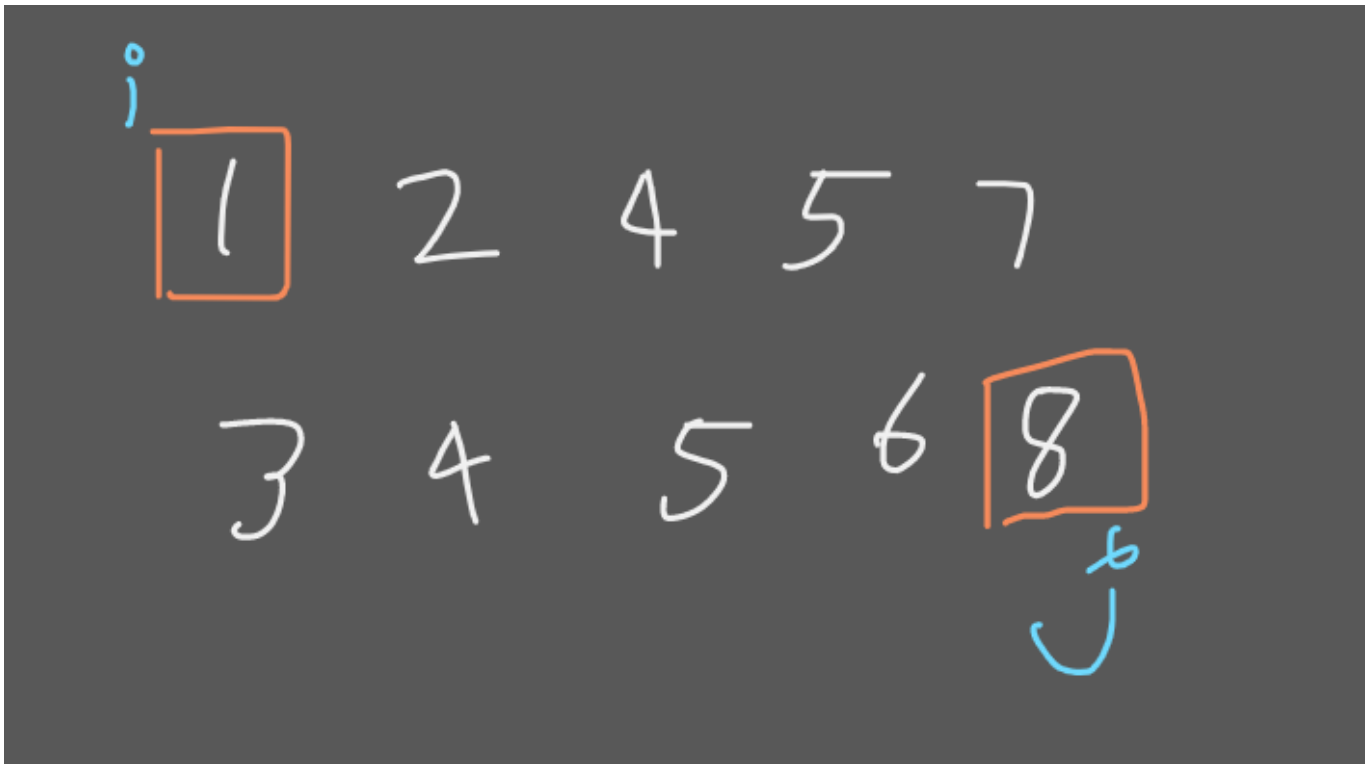
```
(1, 8), (4, 5), (5, 4) are the
pairs which sum to 9.
```

Approach:

⟶ we can use 2 pointer approach here after sorting the both arrays. one pointer will start form 0 and second pointer will start from the end of the second array and then we will have 3 conditions :

```
1. A[i] + B[j] is less than X
2. A[i] + B[j] is greater than X
3. A[i] + B[j] is equal to X
```

⟶ So let's first see the digram for our logic.



⟶ Now if `A[i]+B[j]` is less than X then we will increase the i because if we decrease j then it's gonna be lower because we have already started from right side.

⟶ If `A[i]+B[j]` is greater than X then we will decrease the j because we will find lower values at left side.

If `A[i]+B[j]` is equal to X then we will increase i and decrease j and add the pair in ans vector.

Code:

```
vector<pair<int,int>> allPairs(int A[], int B[], int N, int M, int
X)
    {
        sort(A,A+N);
        sort(B,B+M);
        vector<pair<int,int>>ans;
        int i=0,j=M-1;
        while(i<N && j>=0){
```

```
            if(A[i] + B[j] == X){
                ans.push_back(make_pair(A[i],B[j]));
                i++;
                j--;
            }
            else if(A[i] + B[j] > X){
                j--;
            }else{
                i++;
            }
        }
        return ans;
    }
```