**Maximum Sum Combination: -**

Medium Accuracy: 49.69% Submissions: 19K+ Points: 4

Given two integer array **A** and **B** of size **N** each.  
A **sum combination** is made by adding one element from array **A** and another element of array **B**.  
Return the **maximum K valid sum combinations** from all the possible sum combinations.

Note : Output array must be sorted in **non-increasing** order.

**Example 1:**

**Input:**  
N = 2  
K = 2  
A [ ] = {3, 2}  
B [ ] = {1, 4}  
**Output:** {7, 6}  
**Explanation:**   
7 -> (A : 3) + (B : 4)  
6 -> (A : 2) + (B : 4)

**Example 2:**

**Input:**  
N = 4  
K = 3  
A [ ] = {1, 4, 2, 3}  
B [ ] = {2, 5, 1, 6}  
**Output:** {10, 9, 9}  
**Explanation:**   
10 -> (A : 4) + (B : 6)  
9 -> (A : 4) + (B : 5)  
9 -> (A : 3) + (B : 6)

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **maxCombinations()** which takes the interger **N**,integer **K** and two integer arrays **A [ ]**and **B [ ]**as parameters and returns the **maximum K valid distinct sum combinations** .

**Expected Time Complexity:** O(Nlog(N))  
**Expected Auxiliary Space:** O(N)

**Constraints:**  
1 ≤ N ≤  105  
1 ≤ K ≤  N  
1 ≤ A [ i ] , B [ i ] ≤ 104

**Code: -**

//{ Driver Code Starts

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

class Solution {

public:

vector<int> maxCombinations(int N, int K, vector<int> &A, vector<int> &B) {

// code here

sort(A.begin(), A.end(), greater<int>());

sort(B.begin(), B.end(), greater<int>());

map<pair<int,int>, bool> vis;

priority\_queue<pair<int, pair<int,int>>> pq;

vector<int> ans;

vis[{0,0}] = true;

pq.push( { A[0]+B[0], {0,0} } );

while(K-- and pq.size()){

ans.push\_back(pq.top().first);

int fa = pq.top().second.first;

int fb = pq.top().second.second;

pq.pop();

if(fa+1 < N and vis[{ fa+1, fb }] == false){

vis[{ fa+1, fb }] = true;

pq.push({ A[fa+1]+B[fb], {fa+1,fb} });

}

if(fb+1 < N and vis[{ fa, fb+1 }] == false){

vis[{ fa, fb+1 }] = true;

pq.push({ A[fa]+B[fb+1], {fa,fb+1} });

}

}

return ans;

}

};

//{ Driver Code Starts.

int main() {

int t;

cin >> t;

while (t--) {

int N, K;

cin >> N >> K;

vector<int> A(N), B(N);

for (int i = 0; i < N; i++) {

cin >> A[i];

}

for (int i = 0; i < N; i++) {

cin >> B[i];

}

Solution obj;

vector<int> ans = obj.maxCombinations(N, K, A, B);

for (auto &it : ans) cout << it << ' ';

cout << endl;

}

return 0;

}

// } Driver Code Ends

**T.C: - O(N\*log N)**

**S.C: - O(N)**