Birla Institute of Technology & Science, Pilani

Second Semester 2017-2018, DSA (CS F211)

Lab Assignment #3

1.A set of Towers with heights t1,t2,t3,….tn becomes a tourist spot only when all their heights are same. Assume all the towers are made of bricks stacked vertically and all are of same dimension. To increase/decrease the height of the tower you must add/remove one brick. Adding/Removing a brick from a tower incurs a cost and is different for different towers. You are assigned with the task of making a set of towers a tourist spot such that the cost incurred is as minimum as possible.

Input: First line consists of an integer T which is followed by 3*T lines, 3 lines per test case. First line denotes the Number of towers N. Second line denotes the height of towers[t1,t2,t3,…tn] and third line denotes the cost incurred for adding/removing a brick[c1,c2,…cn]

Output: Minimum cost at which you can make the set of towers a tourist spot.

Constraints: T <= 10; n<=20000; 0<=Ci<=30000; 0<=Hi<=20000

Example:

Sample-Input:

1

3

1 2 3

10 100 1000

Sample-Output:

120

2. Let A be an array of n distinct positive integers. If i < j and A[i] > A[j] then the pair (i, j) is called an inversion of A. Given n and an array A your task is to find the number of inversions of A in O(n logn). Also print all pairs of inversions. **(Note: You cannot use global variables.)**

3.Raju is searching for a book in the Library, he needs a specific book which is stored in one of the filing cabinets arranged in a line against the wall of the Library. He wants his friends to find the book from the filing cabinets. Each filing cabinet has different number of books. He doesn't want his friends to get in each other's way, nor does he want books from different filing cabinets getting mixed up, so he has decided to partition the cabinets, and assign a specific section to each friend. Each friend will have at least 1 cabinet to search through. More specifically, he wants to divide the filing cabinets into N sections (N is the number of friends) and assign a section to each friend. Also, he wants to minimize the number of folders that a friend would have to look through. For example, suppose there were three friends and nine filing cabinets with the following number of folders:

10 20 30 40 50 60 70 80 90

He would divide up the filing cabinets into the following sections:

10 20 30 40 50 | 60 70 | 80 90

The person assigned to the first section would have to look through 150 folders. The person assigned to the second section would have to search through 130 folders, and the last person would filter through 170 folders. In this partitioning, the maximum number of folders that a person looks through is 170. No other partitioning has less than 170 folders in the largest partition.

Input:

First line contains M(Number of Filing Cabinets) and N(Number of friends) separated by space.

Second line contains M integers separated by space denoting the Number of folders in each Filing Cabinet.

Output:

The Maximum number of folders that a person would have to look through in an optimal partitioning of the filing cabinets (Minimizing the number of folders that a friend would have to look through).

Constraints:

-Folders will contain between 2 and 15 elements, inclusive

-Each element of folders will be between 1 and 1000, inclusive

-No of friends will be between 1 and the number of elements in folders, inclusive

Example:

Sample-Input:

9 5

10 20 30 40 50 60 70 80 90

Sample-Output:

110

Explanation:

10 20 30 40 | 50 60 | 70 | 80 | 90

4. Game x is played in a country y. There are N states in country y and you are given the number of players from each of them. There is a rule that for any team to participate in that game, there should be exactly M players in a team and no two players should be from same state. How many maximum teams can you form?

Constraints: 1 <= N <= 50, 2 <= M <= 100, 1 <= nop[i] <= 109

Input:

N M

Second line contains the number of players from ith state

Sample Input-1:

4 3

2 4 3 6

Output:

4

Explanation:

N = 4 and M = 3

Let 4 states be s1, s2, s3, s4. Given that we have 2 from s1, 4 from s2, 3 from s3 and 6 from s4. Each team should contain 3 players from different states. We can form the

following team: {s1, s2, s3}, {s1, s2, s4}, {s2, s3, s4}, {s2, s3, s4}. Note that even though there are 15 players in total we can't form more than 4 teams.

Sample Input-2:

6 4

10 11 12 13 14 15

Output:

18

5.Hari is assigned with the task of finding the mth smallest number from a list of number i.e. number at the mth index when the list is sorted in non-descending order. The Numbers will be given in intervals. For example, the intervals (1, 3) and (5, 7) represent the list of numbers { 1, 2, 3, 5, 6, 7 }. A number may be present in more than one interval, and it appears in the list once for each interval it is in. For example, the intervals (1, 4) and (3, 5) represent the list of numbers { 1, 2, 3, 3, 4, 4, 5 }. **(Note: Indexing is 0-based.)**

You will be given Two Arrays LowerBound and UpperBound. The ith element of LowerBound denotes the smallest number of ith interval and ith element of UpperBound denotes the largest number of ith interval.

Constraints:

-LowerBound will contain between 1 and 50 elements, inclusive.

-UpperBound will contain the same number of elements as LowerBound.

-Each element of LowerBound and UpperBound will be between -2,000,000,000 and 2,000,000,000, inclusive.

-The i-th element of LowerBound will be less than or equal to the i-th element of UpperBound.

-n will be a non-negative integer less than the total number of elements in the list, but no greater than 2,000,000,000.

Input:

N(Number of Intervals) M

N space separated integers denoting LowerBound

N space separated integers denoting UpperBound

Output:

Output the Mth smallest Number.

Example:

Sample-Input:

2 4

1 5

3 7

Sample-Output:

4

Explanation: The numbers are 1 2 3 5 6 7. 4^{th} smallest Number is 6 (0-based Indexing).

6.You will be provided with arrays X[] and Y[].You are assigned with the task of sorting X[] so that the order of numbers in X[] after sorting is dictated by the order in which numbers occurred in Y[].Numbers which didn't occur in Y[] are to be present at the end of X[] in sorted order.

Note: Don't use the sort in-built function.

Example:

X[] = {2,5,3,8,1,7,8,32}

Y[] = {8,1,2,7}

Output = {8,8,1,2,7,3,5,32}

Input:

N(Size of X) M(Size of Y)

N space separated Integers denoting X

M space separated Integers denoting Y

Output:

X[] after Sorting

Constraints: N <= 100000; M <= 100000

Sample-Input:

9 3

8 3 3 5 53 6 3 1 3

6 3 1

Sample-Output:

6 3 3 3 3 1 5 8 53

7. Young Sheldon bought a new computer operating on the Intel 8086. He decides to test it out and see how fast it can process his codes. He wants to input a long array of integers and see how quickly merge sort can be performed. But he would require an array which would maximize the number of comparisons in the sorting algorithm. Because Sheldon is too young to figure out the order in which the numbers have to be presented for this, you are required arrange the numbers for him.

Input: Two lines

Line 1: Number of elements in the array

Line 2: Sequence of numbers to be ordered

Output: One line showing the sequence in which the array has to be presented.

Example: [1,9,5,13,3,11,7,15,2,10,6,14,4,12,8,16] takes 153 comparisons while the sorted one takes only 30 comparisons.

Time Complexity: O(nlogn)

8.Tony Stark is building his new Hulkbuster suit with the help of Bruce Banner. He goes for a battle with Hulk every day to assess their work that day. Tony maintains a score for how well he did against the Hulk. Every time his score crosses 3 times the median of last m days, he considers that day's work a success. Given the past n day's data, help Jarvis develop a code to determine whether that day's work was a success or not.

Don't consider the first m days for evaluation. Output -1 for all such cases.

Input: m n

Array of size n containing the scores for n days.

Output: Array of size n containing 0s and 1s (and -1s initially) denoting 0 for failure and 1 for success on that day.

Time Complexity: O(nmlogm). Explore the algorithm for O(nm).

9. In a normal sorting algorithm, we are given an array of integers. Suppose we are given an array of strings where each string represents a numeric value. These strings can be big for 'long long int' to store their values. Write a program to sort the array in an ascending order.

Input: n – denoting size of the array

Followed by n numeric strings

Output: n lines of numeric strings sorted in ascending order.

Example:

Sample-Input:

3

123

1

33584523167436435345

Sample-Output:

1

123

33584523167436435345

10.Given an Array sort it using Bubble Sort, Insertion Sort, Selection Sort and output the name of the Algorithm for which the number of swaps is minimum along with the total number of swaps the algorithm performed. If any two Algorithms have the same number of swaps output the Lexicographically Smallest Algorithm Name.