CS213 Minor - Assignment1

Total Marks: 100

Submission Instructions

Submission Instructions: Create a zip file containing the solution for all problems. Zip file should be named <**RollNo.**>_**Assignment1.zip**.
Each problem should be named as <**RollNo.**>_<**ProblemID**>**.zip** (Example: 180070050_*P*1.*cpp*).

Problems

1. You are provided with a list of students with their marks in the DSA course. The task is to find all the students having marks between the specified range (x,y) (inclusive of x and y), the final output should be printed in decreasing order of marks. You may make use of STL for this question.

   (a) Create a data structure to take input of students and their marks.

   (b) Create a function for **fetchMarksQuery(list_of_students,x,y)** which takes input the range (x,y) and returns the list of students having marks in the specified range. (Hint: Use Searching)

   (c) The main function should have a loop over the queries

   (d) The complexity of the sorting algorithm should not exceed **O(nlogn)** and the query part should not exceed **O(p+logn)**, where **p** is the number of students to be returned for that particular query.

   Input:
   The first line contains number of students n. $1 <= n <= 10^6$
   The next n lines contains each students ID along with his corresponding marks.
   The line following this would be the number of queries say q.
   The next q lines should contain pairs x space seperated with y, the range of marks for each query

   Output:
   For each query, the first line prints x seperated with y.
   The next p lines prints the list of students having marks in this range.

   Sample:
   Input:
   4
   214050009 89
   214050006 90
   214050001 78
   214050002 99
   2
   98 99
   80 90
   Output:
   98 99
   214050002
   80 90
   214050009
   214050006

2. You are given prices for a particular stock for the next n days, say $p_1, p_2, ...p_n$. You want to buy **one stock** on one of the upcoming days and sell it on a later day. Both buying and selling should be done within the next n days. You want to find out which buying and selling days will maximize your profit. If multiple of these options are possible print with the earliest buying date. The time complexity should not exceed $O(n)$.

   Input:
   The first line contains number of days n. $1 <= n <= 10^8$
   The second line contains prices separated by space
   Output:
   The first line prints maximum profit possible.
   The second line prints the earliest buy date possible such that profit is maximised.

   Sample:
   Input:
   5
   100 10 1110 5000 4000
   Output:
   4990
   2

3. The **chocolate-packing** problem is the following. You are a shopkeeper. On any given day, you are able to produce n chocolates. Using these n chocolates, you must create different packets, where each packet can contain any number of chocolates. You are given a table of prices , where the i^{th} entry represents the price of selling i chocolates in one packet. Using this, determine the maximum revenue obtainable by distributing the n chocolates into different packets. You can assume that all packets will be sold (There will be no packet that does not generate any revenue).

   Input:
   The first line contains the number of chocolates n. $1 <= n <= 10^4$
   The second line contains n space separated integers where the i^{th} number represents the revenue obtained when a packet containing i chocolates is sold. $1 <= i <= n$; $1 <= p_i <= 10^3$
   Output:
   The maximum amount of revenue obtained after packaging the chocolates.

   Sample:
   Input:
   2
   3 4
   Output:
   6

   Hint: If you are getting the correct answer but your program is taking too long as n increases, think of some things you are computing more number of times than required. Is there any way you can make this process more efficient?

4. **Alternate approach to sorting:** Most of you have learned some basic sorting algorithms in CS 101. Here we discuss an alternate approach to sorting an array.
   We will not be giving the algorithm, but an approach to do so is by trying to first solve this subproblem:
   Suppose I represent a set of integers using an array that holds the integers in sorted order. Given two such arrays representing two sets. Can you think of an algorithm to combine the elements into a single sorted array?
   Do not implement any other algorithm for sorting.

   Input:
   The first line number of integers n. $1 <= n <= 10^8$
   The second line contains n space separated integers
   Output:
   The same n integers in sorted order

   Sample:
   Input:
   3
   3 1 2

   Output:
   1 2 3

5. Suppose you are given k sorted arrays, each with n elements, and you want to combine them into a single array of kn elements.
   Consider the following approach. Using the solution to the problem described above, you merge the first 2 arrays, then merge the 3^{rd} given array with this merged version of the first two arrays, then merge the 4^{th} given array with the merged version of the first three arrays, and so on until you merge in the final (k^{th}) input array. What is the running time taken by this successive merging algorithm, as a function of k and n?
   The above method, for the given problem specification turns out to be too slow. Lets try something else. Let us make a bigger array of size nk and fill it with the elements of all the arrays. Now lets apply the sorting algorithm on this array. What would be the running time of this algorithm? While this would be faster than the above method in some cases, it still is not good enough to meet the constrains of the problem.
   Can you think of a faster way to do the k-way merge procedure?
   Implement this optimised merge procedure on C++.

   Constraints:
   $1 <= n <= 10^8$
   $1 <= k <= 10^6$
   $1 <= nk <= 10^8$

   Input:
   First line - n seperated with k
   Each of the next k lines contain a sorted list of integers of size n
   Output:
   kn integers in a single line in sorted order

   Sample:
   Input:
   4 3
   1 2 7 9
   3 4 11 14
   3 6 8 10
   Output:
   1 2 3 3 4 6 7 8 9 10 11 14