CS6013: Advanced Data Structures and Algorithms IIT Hyderabad, Fall 2020

Programming Assignment 2 (Dynamic 2-SUM)

- Out on: **17th Nov 2020**. Due Date: **11:59pm, 27th Nov 2020**.
- The total marks for this assignment is 50 (scaled down to 10 for final weightage).
- For submission and upload instructions, see the section after the questions.

---

1 Dynamic 2-SUM

1.1 Problem Description

Consider the 2-SUM problem: given an array of n integers (possibly with repetitions), and a target integer t find if there exist two distinct elements x, y in the array such that $x + y = t$. There are multiple approaches to find a $O(n \log n)$ solution to this problem.

We would like to implement a *dynamic* version of this problem. In this version, you do not know the number of elements in advance. The input arrives as a sequence of operations. We start with the empty multiset S. Operations are of three types:

1. Insert(k) : Inserts a number k into S.
2. Delete(k): Deletes an instance of the key k from S. If k is not present, no change is made to the data structure. If k is present multiple times, any one instance is deleted.
3. Query (a, b): Prints the *number* of target values in the closed interval $[a, b]$ such that there are *distinct* elements x, y in the multiset with $x + y = t$.

Note that by "distinct" above, we mean two distinct elements of the multiset, which could have the same integer value. E.g. if $S = \{5, 5, 5\}$, then then `Query (10,15)` returns the answer 1, whereas `Query(17,20)` returns the answer 0.

Space Constraint: Your program should use at most $O(n)$ space at any point of time, where n is the number of elements in the multiset *at that point in time*.

You are to make your choice of data structure and implementation algorithm, so that the above can be handled efficiently. All data structures used must be implemented from scratch by you, and no pre-existing libraries of data structures are allowed to be used.

You should include a single pdf file named `report.pdf` in your submission that contains an explanation of the choice of data structure, and what are the time bounds (expected/amortized/worst-case) for each operation on it.

1.2 Programming Specifications

Input

Each operation is specified by two lines: one giving the command (a single character), and second giving the argument. All inputs are separated by a newline. For e.g. Consider the sequence of operations: Insert(10), Insert (-21), Insert(2), Insert(2), Delete(-20), Delete (-21), Query(-20, 40), Query(100,200). This is specified by the sequence:

```
I
10
I
-21
I
2
I
2
D
-20
D
-21
Q
-20 40
Q
100 200
E
```

We use `I` for insert, `D` for delete, and `Q` for query. Arguments to query are separated by a single space. End of commands is marked by the character `E`, which, if encountered, the program can quit.

You may assume that the input sequence will always be in a valid/correct format. Inputs will be given on **Standard Input**.

Output

Outputs should be printed to **Standard Output**. For every Query Q in the input stream, you should immediately print a single integer to standard output specifying the answer to that query, followed by a newline. Each query has exactly one line of output.

For instance, the output on the above input should be:

```
2
0
```

1.3 Programming Language

1. You may use one of C, C++, Java or Python for this.

2. You should not use in-built structures like Lists in Python (will be considered $O(n)$), ArrayLists in Java or the Vector class in C++ to implement your data structure. Similarly, HashSets or equivalents are not permitted. If you have a query about something being used or not, ask on canvas before using it.

3. Instructions (for programming and submissions) specific to each language will be updated soon. Some generic guidelines are: for C/C++, the programs should compile in gcc with the –std=c++14 switch. Python programs should compile on python3 . **Do not submit python notebooks**.

2 Submission Guidelines

1. Your submission will be one zip file named `<roll-number>.zip` , where you replace `roll-number` by your roll number (e.g. `cs20mtech11003.zip`), all in small letters. The compressed file should contain the below mentioned files:

   (a) Programming files (please do not submit python notebooks or IDE files). More detailed instructions will follow.

   (b) A description of your solutions in **pdf** format named `report.pdf`. This file should describe the algorithms you use, Pseudo-code is the preferred way to write out algorithmic concepts. If it is not an algorithm described in class, then you have to both describe it and argue that it is correct. You may state and used results proven in class without proof.

   (c) Upload your zip files on canvas. No delays permitted.

2. The preferred way to write your report is using LaTeX(Latex typesetting). However it is okay to submit pdf files not created in latex. **No handwritten files please!**

3. Failure to comply with instructions (filenaming, upload, input/output specifications) will result in your submission not being evaluated (and you being awarded 0 for the assignment). Do not print a single extra character over what is needed for the output.

4. **Plagiarism policy:** If we find a case of plagiarism in your assignment (i.e. copying of code, either from the internet, or from each other, in part or whole), you will be awarded a **zero** and will lead to a **FR** grade for the course in line with the department Plagiarism Policy (https://cse.iith.ac.in/academics/plagiarism-policy.html). Note that we will not distinguish between a person who has copied, or has allowed his/her code to be copied; both will be equally awarded a zero for the submission.

3 Evaluation

You will be marked for the following aspects:

1. Efficiency (time/space)
2. Correctness of Algorithm
3. Code clarity and comments
4. Report presentation