150015-Semester II - 5781

Data Structures I

**Homework Assignment #10 – The last one!!!**

**Question 1**

Suggest a data structure (which might be a combination of data structures) that stores distinct **natural numbers** and supports the following functions all running in O(log n) time. Explain your choice.

Insert(A,x): inserts x in the data structure.

Delete(A,x): Removes x from the data strucutre.

CountInRange(A,a,b) : Returns the number of elements in the the

range [a,b].

**Question2**

Suggest a data structure for saving integers that support the basic stack functions (Push, Pop, Top, isEmpty) in addition to the function GetMax which returns the largest value in the data structure **without removing it from the data stucture.**

For example, after inserting the follwing values (from left to right)

5, 4, 3, 6 and after one removal the remaining values will be 5,4,3. The function Top would return 3 and the function GetMax would return 5.

All the functions must run in constant time – O(1)

Explain your suggested data structure and describe (in pseduo-code or in words) how you would implement all the functions.

**Question 3**

**Birthday** is a record (structure) containing two fields:

name – containing a name of a person.

day – containing a number between 1 and 365 representing the day in the year that the person was born on.

The data structure **Birthdays** contains a collection of Birthday records and the following operations are defined on it:

|  |  |
| --- | --- |
| The function | Description |
| addBirthday( P,b) | The operation adds a Birthday b, to the Birthdays collection, P. |
| printBirthday(P,day) | The operations prints all the names in P with the the birthday, day. |

1. Design an implementation for **Birthdays** so that the run time complexity of the operation addBirthday will be O(1) and the run time complexity of the operation printBirthday will be O(n), where n is the number of birthdays occurring on day.
2. Write the algorithm for the operation addBirthday according to your implementation.

**Question 4**

Write an algorithim that accepts an array of n distinct positive numbers and an integer k, where k <= n, and computes the sum of the k largest numbers in the array.

For example: For the array {8, 14, 7, 12, 42 } and k =3 the algorithim will return 68 (42+12+8).

You can use any data structues and algorithims you learned in the course without needing to implement them.

The algorithim must run in O(n log k) time.

**Question 5**

Suggest a data structure for saving both positive and negative integers. The structure must be able to contain more than one apperance of a number and support the following functions in the given time restraints.

1. Inserting a new number into the structure in O(log n) time.
2. Deleting a number in the structure in O(log n) time.
3. Searching for a number in the structure in O(log n) time.
4. Determine if in the structure there is a pair of opposite numbers (for example 3 and -3) in O(1) time.

**Question 6**

In the relevant cell in the following table, for data stucture X and required data structure Y, determine what is the needed time to convert a structure of type X containg n values to a structure of type Y containg the same n values.

Write the most efficient run-time complexity for the worst case scenario for converting the data structure to the reguired data structure. You can assume that the data in the data structures are numbers.

|  |  |  |  |
| --- | --- | --- | --- |
| The desired data structure 🡪  Given data structure | Sorted Array | AVL tree | Max Heap |
| Single Linked List |  |  |  |
| Binary Tree |  |  |  |
| Min Heap |  |  |  |

Explain your answers.