Homework 2

February 13, 2018

In this howework you will use hashing, red-black trees, lists, binary seach, and sorting algorithms. You should follow the rules for writing quaity code.

1. Solve https://leetcode.com/problems/count-of-range-sum/description/ in Python using red–black trees.

2. Suppose that we have an array with 'a's and 'z's. Write a program that finds the subarray of maximum size with the same number of 'a's and 'z's. Expected time complexity must be $O(n)$. Example:

   (a) Input: ['a', 'a', 'z', 'z', 'a']. Output: 0 to 3 Or 1 to 4

3. Suppose that we have a sorted array $[0, .., n-1]$. We choose an element i of the array creating two subarrays: $[0, .., i-1]$ and $[i+1, .., n-1]$. We interchange these subarrays, which gives the new array: $[i+1, .., n-1, i, 0, .., i-1]$. Code a function that finds an element in the modified array in $O(\log n)$ time. Example:

   (a) Input: [15, 16, 17, 18, 19, 20, 11,12, 13], key= 13. Output: Index 8

4. Find the k-th largest element in an array in $O(n)$ expected time.

5. Implement mergesort or quicksort but, when a vector has a size smaller than K in a call, use insertion sort instead. Find the value of K that maximizes empirical performance. (Note that K = 1 is the same as not having a K at all.)