# CS 161 W19: Recitation 4 Problems

February 2019

## Exercise 0

You are the head software engineer at a new video-sharing startup called TrickTok. The app hosts videos of magic tricks being performed to copyrighted music.

Obviously, the app is a huge success. You have loads of users. You have $n$ users, to be exact.

Each user has a username. Usernames can only contain lowercase letters, numbers, dashes, underscores, and periods. Also, each username can be at most $k$ characters long, where $k << n$. No two users can have the same username. As of right now, you have all the usernames stored in your company's computers, sorted based on the date they were created. The first username in the list is yours, since you made the first account. The last username in the list is the username of the person who most recently created an account.

You want to implement a search bar that allows users to quickly find certain accounts just by querying the account's username.

a Using the current way the list is sorted, what is the fastest you can possibly find a username?

b Design an $\Theta((nk)logn)$ algorithm that sorts the usernames in alphabetical order.

c Design a $\Theta(n)$ algorithm that sorts the usernames in alphabetical order. Be specific about your algorithms design. If you're using BucketSort, tell us how many buckets. If you're using RadixSort, give us values for $r$ (the number of buckets) and $M$ (The number of possible usernames). Don't forget that usernames can have different lengths!

d Now that the list is sorted, what is the fastest you can find a particular username?

## Exercise 1

Trees! For each of the following scenarios, tell us whether it is possible to design an algorithm under the given constraints. If so, tell us the algorithm. If not, tell us why not.

a Find the maximum value of a given Binary-Search Tree in $O(logn)$ time.

b Find the maximum value of a given Red-Black Tree in $O(logn)$ time.

c Find the second-highest value of a given Binary-Search Tree in $O(n)$ time.

d Find the second-highest value of a given Red-Black Tree in $O(n)$ time.

e Find the maximum value of a given sorted array in $O(logn)$ time.

f Find the second-highest value of a given sorted array in $O(logn)$ time.