

Name:
ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.

Advice 2: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solution:

- The solutions **should be typed**, we cannot accept hand-written solutions. Here's a short intro to [Latex](#).
- In this homework we denote the asymptomatic *Big-O* notation by \mathcal{O} and *Small-O* notation is represented as o .
- We recommend using online Latex editor [Overleaf](#). Download the **.tex** file from Canvas and upload it on overleaf to edit.
- You should submit your work through [Gradescope](#) only.
- If you don't have an account on it, sign up for one using your CU email. You should have gotten an email to sign up. If your name based CU email doesn't work, try the identikey@colorado.edu version.
- Gradescope will only accept **.pdf** files (except for code files that should be submitted separately on Canvas if a problem set has them) and **try to fit your work in the box provided**.
- You cannot submit a pdf which has less pages than what we provided you as Gradescope won't allow it.

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

Piazza threads for hints and further discussion

Piazza Threads

Question 1

Question 2

Question 3

Question 4

Recommended reading

Divide Conquer; Recurrence Relations: Ch. 2 2.3; Ch. 4 4.1, 4.2, 4.3, 4.4, 4.5

Quicksort: Chapter 7 complete

Efficient Data Structures: Hash Tables: Ch. 11 Complete

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

1. (20 pts) Let $A = \langle a_1, a_2, \dots, a_n \rangle$ be an array of numbers. Let's define a 'flip' as a pair of distinct indices $i, j \in \{1, 2, \dots, n\}$ such that $i < j$ but $a_i > a_j$. That is, a_i and a_j are out of order.

For example - In the array $A = [1, 3, 5, 2, 4, 6]$, $(3, 2)$, $(5, 2)$ and $(5, 4)$ are the only flips i.e. the total number of flips is 3. (Note that in this example the indices are the same as the actual values)

Design a divide-and-conquer algorithm with a runtime of $\mathcal{O}(n \log(n))$ for computing the number of flips. Your algorithm has to be a divide and conquer algorithm that is modified from the Merge Sort algorithm. Explain how your algorithm works, including pseudocode. You can add a picture of your pseudo-code or properly commented code.

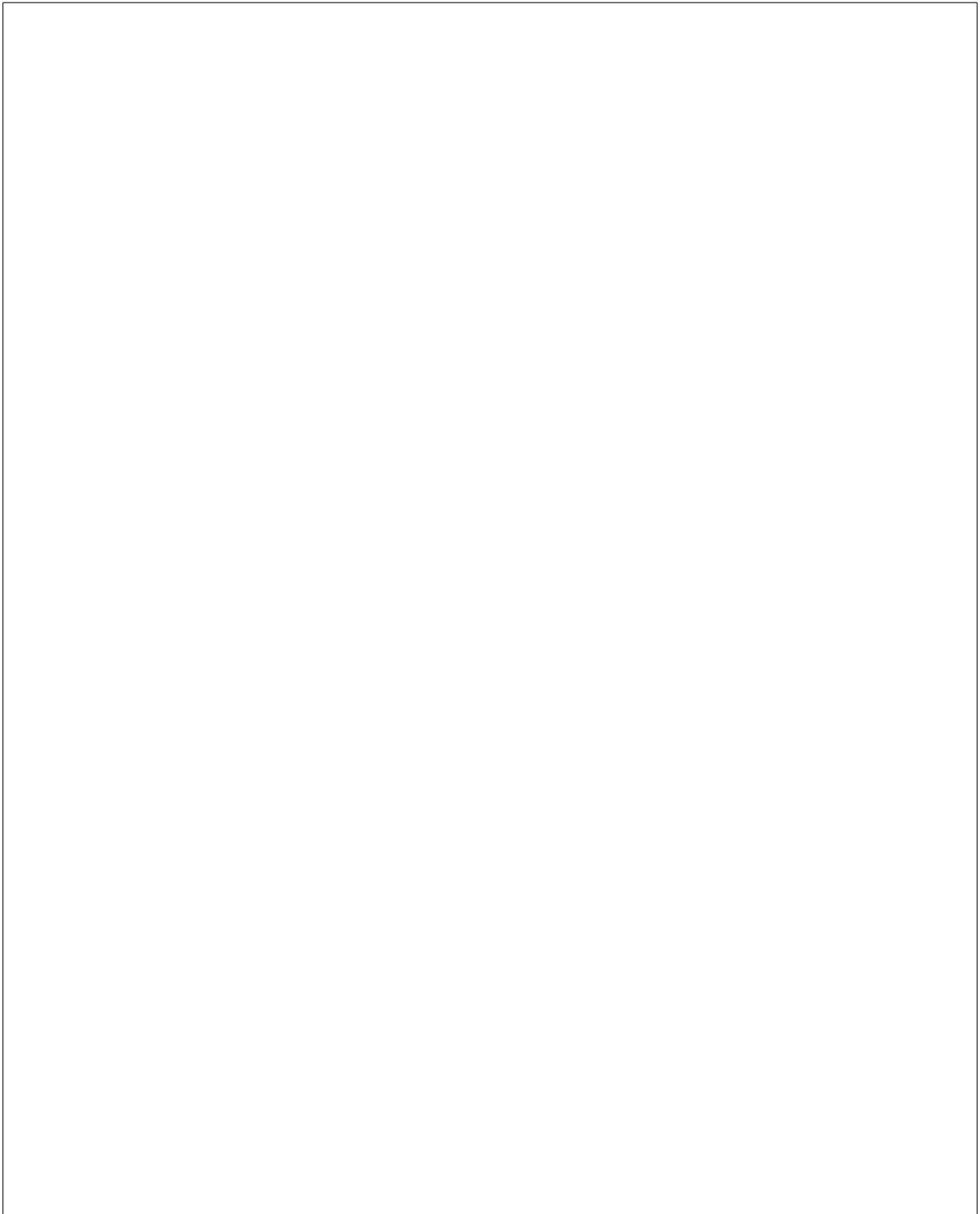


Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder



Name:
ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

2. (10 pts) For the following problems, you must use the pseudocode for *QuickSort* and *Partition* in Section 7.1 of the *Introduction to Algorithms 3rd edition (CLRS)* and the array $A = [22, 40, 67, 55, 10, 92, 66]$.

(a) (3 pts) What is the value of the pivot in the call $\text{Partition}(A; 1; 7)$? (Array indexing starts at 1 as per the convention in the book)

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

(b) (3 pts) *What is the index of that pivot value at the end of that call to Partition?*

(c) (4 pts) *On the next recursive call to Quicksort, what sub-array does Partition evaluate? (Give the indices specifying the subarray.)?*

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

3. (15 pts) Suppose in quicksort, we have access to an algorithm which chooses a pivot such that, the ratio of the size of the two subarrays divided by the pivot is a **constant** k . i.e an array of size n is divided into two arrays, the first array is of size $n_1 = \frac{nk}{k+1}$ and the second array is of size $n_2 = \frac{n}{k+1}$ so that the ratio $\frac{n_1}{n_2} = k$ a constant.

(a) (3 pts) Given an array, what value of k will result in the best partitioning?

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

- (b) (9 pts) Write down a recurrence relation for this version of QuickSort, and solve it asymptotically using **recursion tree** method. For this part of the question assume $k = 3$. Show your work, write down the first few levels of the tree, identify the pattern and solve. Assume that the time it takes to find the pivot is $\Theta(n)$ for lists of length n .

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

- (c) (3 pts) *Provide a verbal explanation of how this Partition algorithm affects the running time of QuickSort, in comparison with the case where the best possible pivot is always used. Does the value of k affect the running time?*

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

4. (10 pts) Consider the hash function $h(k) = \lfloor 100k \rfloor$ for all keys k for a table of size 100. You have three applications.

- **Application 1:** Keys are generated uniformly at random from the interval $[0.3, 0.8]$.
- **Application 2:** Keys are generated uniformly at random from the interval $[0.1, 0.4] \cup [0.6, 0.9]$.
- **Application 3:** Keys are generated uniformly at random from the interval $[0, 1]$.

(a) (2 pts) For which application does the hash function $h(k)$ perform worse? Please explain/justify adequately your answer.

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

- (b) (3 pts) *In each of the three applications does the hash function satisfy the uniform hashing property? Please explain/justify adequately your answer.*

Name:

ID:

CSCI 3104, Algorithms
Homework 2B (55 points)

Escobedo & Jahagirdar
Summer 2020, CU-Boulder

- (c) (5 pts) Suppose you have n keys in total for each application. What is the resulting load factor α for each application? Assume that in each application only slots of the hash table that have a chance of being filled are considered when calculating α .

5. **Extra Credit (5% of total homework grade)** For this extra credit question, please refer the leetcode link provided below or click [here](#). Multiple solutions exist to this question ranging from brute force to the most optimal one. Points will be provided based on Time and Space Complexities relative to that of the most optimal solution.

Note: The brute force approach in this question would be to create an array of size 10^6 and directly access the key using the index of the array. For this question, we expect you to perform collision handling once you obtain a hash value for a particular key.

Please provide your solution with proper comments which carries points as well.

<https://leetcode.com/problems/design-hashmap/>

Replace this text with your source code inside of the .tex document