

## Practice Exam 2

- Do not open this exam booklet until you are directed to do so. Read all the instructions on this page.
- When the exam begins, write your name on every page of this exam booklet.
- This exam is closed book. You may use one handwritten  $8\frac{1}{2} \times 11$  or A4 crib sheet. No calculators or programmable devices are permitted.
- Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem. Do not put part of the answer to one problem on the back of the sheet for another problem, since the pages may be separated for grading.
- Do not waste time and paper rederiving facts that we have studied. It is sufficient to cite known results.
- Do not spend too much time on any one problem. Read them all through first, and attack them in the order that allows you to make the most progress.
- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Be neat.
- Good luck!

**Problem 1.** The administrators at Pen 'n' Slate University, located in Ecstatic Valley, have decided to move into the 20th century and computerize their records. You've been asked to design a database containing current and former students. Each student's record consists of ID number, name, start date, graduation date (if applicable), GPA.

In each of the following, state which abstract data type addresses the required operations (if applicable), which data structure you will use to store the student records, what information will be used as keys, which operations you will support and how long these operations take asymptotically, assuming that your data structure holds  $n$  student records.

- (a) You are required to add students to the database, remove them from the database, and search for their records by ID in  $O(1)$  time.

**Abstract data type :**

**Data structure:**

**Keys:**

**Operations and their running time:**

- (b) The Dean now wants to be able to produce a Dean's list from time to time. You should be able to tell him, at any time, the list of all currently enrolled students with GPA After a given threshold  $a$ . Does the insert/delete time change?

**Data structure:**

**Keys:**

**Operations and their running time:**

- (c) The Dean now changes his mind and says that he wants to get the list of the top 5% of currently enrolled students.

**Data structure:**

**Keys:**

**Operations and their running time:**

Name: \_\_\_\_\_

CSE 465 Practice Exam 2, page 3

- (d) The Registrar's office wants to be able to find all students who were enrolled at any point during a give time period (for example, from Fall 1999 to Spring 2003).

**Data structure:**

**Keys:**

**Operations and their running time:**

**Problem 2** (Short Answers). Give a short answer and a brief justification for each of the following questions. The better your argument, the higher your grade. **No points will be given even for a correct answer if no justification is presented.**

- (a) True or false?  $\frac{n}{2} = o(n)$ .
- (b) How long does it take to find an element of rank  $\frac{n}{4}$  in an unsorted list?
- (c) True or false? There exists an algorithm that, in  $O(n)$  expected time, determines whether an array of  $n$  integers has repeated elements, i.e., whether there are two distinct indices  $i, j$  s.t.  $A[i] = A[j]$ .
- (d) State the recurrence for the *expected* running time of QUICKSORT.
- (e) Recall that in the randomized partition algorithm, a pivot is “ok” if it induces a split where both pieces are at least  $1/4$  of the array size. What is the probability of getting two ok splits in a row in the randomized order statistics algorithm?
- (f) Can the max-heap property be used to print out the keys of an  $n$ -node heap in sorted order in  $O(n)$  time?
- (g) You want to modify a standard hash table so it supports the following operations: COUNT, which returns the number of elements currently in the hash table (in  $O(1)$  time); LIST-ALL, which lists all the elements currently in the hash table. If the table currently contains  $k$  elements, LIST-ALL should run in  $O(k)$  time. State the changes to the data structure and explain briefly why the operations take the desired time.
- (h) Write pseudocode which takes a binary search tree node  $x$  as input and returns the height of the tree rooted at  $x$ .

**Problem 3.** Your goal is to understand the following algorithm and analyze its running time.

Initial call: MYSTERY-ALG( $A, 1, \text{length}(A)$ )

MYSTERY-ALG( $A, p, q$ )

- ▷  $A$  is an array of integers (could be negative);
- ▷  $p$  and  $q$  are indices between 1 and  $\text{length}(A)$  such that  $p \leq q$ .

```

1  if  $p = q$  then return  $|A[p]|$ 
   ▷ Here  $|\cdot|$  denotes absolute value.

2   $m \leftarrow \lfloor \frac{p+q}{2} \rfloor$ 
3   $left \leftarrow$  MYSTERY-ALG( $A, p, m$ )
4   $right \leftarrow$  MYSTERY-ALG( $A, m + 1, q$ )

5  Create two new arrays  $B[p..m]$  and  $C[m + 1..q]$ 
   ▷ Initialize  $B$ :
6   $B[m] \leftarrow A[m]$ 
7  for  $i \leftarrow m$  downto  $p$ 
8     do  $B[i] \leftarrow B[i + 1] + A[i]$ 
   ▷ Initialize  $C$ :
9   $C[m + 1] \leftarrow A[m + 1]$ 
10 for  $i \leftarrow m + 1$  to  $q$ 
11    do  $C[i] \leftarrow C[i - 1] + A[i]$ 

12 MERGESORT( $B, p, m$ )
13 MERGESORT( $C, m + 1, q$ )
   ▷ LIGHTEST-SUM takes two sorted arrays of integers and three indices  $p, m, q$ ,
   ▷ and returns  $\min_{\substack{p \leq i \leq m \\ m+1 \leq j \leq q}} |B[i] + C[j]|$  (that is, it returns the absolute value of the closest
   ▷ number to 0 one can get by adding a number from  $B$  to a number from  $C$ ).

14  $middle \leftarrow$  LIGHTEST-SUM( $B, C, p, m, q$ )
15 return  $\min(left, middle, right)$ 

```

- (a) Suppose  $A = [ 2 \quad 3 \quad 4 \quad 5 \quad -15 \quad 2 \quad 3 \quad 4 ]$ . What is the output of the initial call to MYSTERY-ALG?
- (b) What does MYSTERY-ALG compute in general?
- (c) Assume that LIGHTEST-SUM runs in time  $O(n)$  in the worst case. State and solve a recurrence which describes the running time of the MYSTERY-ALG algorithm.

**Problem 4** (Analysis of 3-ary heaps). A **3-ary heap** is like the binary heap we learned about in class, with the exception that non-leaf nodes have 3 children instead of 2.

- (a) How would you represent a 3-ary heap in an array?
- (b) Implement  $\text{PARENT}(i)$  that, given the index  $i$  of a node, returns the index of its parent and  $\text{CHILD}(i, k)$  that, given the index  $i$  of a node, returns the index of its  $k$ th child.
- (c) What are the minimum and the maximum number of elements in a 3-ary heap of height  $h$ ?
- (d) Design an efficient implementation of Extract-Max in a 3-ary max-heap. Give both an English description and pseudocode. Analyze the running time of your algorithm in terms of  $n$ .

**Problem 5** (Augmenting Data Structures). In your summer job, you maintain a database of temperature measurements taken at the Tough Trees resort over many consecutive days. Your employer would like to be able to quickly answer questions of the form: “What was the average (or maximum) temperature at Tough Trees between day  $a$  and day  $b$ ?”

Your task is to create a data structure which supports the following operations efficiently:

- $\text{INSERT}(x, i, temp)$  adds a new day  $i$  to the data structure  $x$ , on which the temperature measured was  $temp$ . Days are numbered  $1, 2, \dots$ . You may assume, for simplicity, that no days are skipped or repeated.
- $\text{MAXINRANGE}(x, a, b)$  returns the maximum temperature measured at any time between day  $a$  and day  $b$  (inclusively).
- $\text{AVGINRANGE}(x, a, b)$  returns the average temperature between days  $a$  and  $b$  inclusively ( $a < b$ ), that is,  $\frac{1}{b-a+1} \sum_{i=a}^b temp_i$ .

One of the caddies at Tough Trees suggests that you create a binary search tree with one node for each day. The node’s key is the day itself. The node also stores that day’s temperature, and some additional information. Your goal is to implement the operations After so they run in time  $O(h)$  where  $h$  is the height of the search tree.

Following the augmentation methodology seen in class:

- (a) What additional information will you store at each node?
- (b) Explain briefly why the  $\text{INSERT}$  operation can be implemented in time  $O(h)$ .
- (c) Give pseudocode for the the operations  $\text{MAXINRANGE}$  and  $\text{AVGINRANGE}$ .

Hint: To write the code for  $\text{MAXINRANGE}$ , first write procedures  $\text{MAXAFTER}(x, a)$  which returns the maximum temperature anytime on or after day  $a$ , and  $\text{MAXBEFORE}(x, b)$  which returns the temperature anytime on or before day  $b$ . For  $\text{AVGINRANGE}$ , first write procedures  $\text{SUMAFTER}(x, a)$ , which returns the sum of temperatures on all days on or after  $a$ , and  $\text{SUMBEFORE}(x, b)$ , which returns the sum of temperatures on all days on or before day  $b$ .