### **Design & Analysis of Algorithms**

Midterm

Date: Thursday, Oct. 15, 2020

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

* You have 150 minutes for this exam.
* It consists of 5problems worth 50 points each, plus a problem 6 for extra 30 points credit.
* The extra credit will be recorded separately, so make sure you have answered all questions from first 5 problems before moving on to problem 6.
* You need to collect 200 points out of 280 to have an A.

## **Instructions**

Work as many problems as possible. All problems have the same value, but subparts of a problem may have different values (depending on their difficulties, importance, etc). Provide a short preliminary explanation of how an algorithm works before running an algorithm or presenting a formal algorithm description, and use examples or diagrams if they are needed to make your presentation clear. Please be concise and give well-organized explanations. Long, rambling, or poorly organized explanations, which are difficult to follow, will receive less credit.

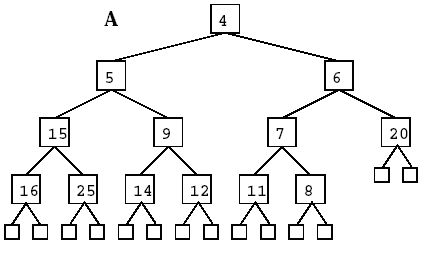
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Problem 1  (50) | Problem 2  (50) | Problem 3  (50) | Problem 4  (50) | Problem 5  (50) | Extra Credit  (30) | Total  (200/280) |
|  |  |  |  |  |  |  |

Problem 1 [Algorithm Analysis].

For each of the following algorithms, indicate their worst-case running time complexity using Big-Oh notation, and give a brief (2-3 sentences each) summary of the  
worst-case running time analysis.  
(a) (10 points) Selection-sort on a sequence of size n.  
(b) (10 points) Merge-sort on a sequence of size n.  
(c) (10 points) Bucket sort on a sequence of n integer keys, each in the range of [0; N]  
(d) (10 points) Find an element in a binary search tree that has n distinct keys.  
(e) (10 points) Find an element in a red-black tree that has n distinct keys.

**Problem 2 [Heap].**

1. (5 points) Give the definition of a (min) heap.
2. (10 points) What is the worst-case complexity of heap-sort? Explain.
3. (30 pts) Perform sequence of ***insert(10),* *removeMin, insert(3)*** operations on heap ***A.*** Draw the heap after each operation.



**Problem 3 [Hash Tables].**

You have a hash table of size *m* = 11 and two hash functions *h*1 and *h*2. Here  
are some precomputed hash values:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| word | ape | bat | bird | cow | dog | goat | hare | koala | mule | panda |
| h1 | 5 | 0 | 5 | 6 | 0 | 2 | 0 | 6 | 1 | 3 |
| h2 | 2 | 5 | 5 | 7 | 1 | 2 | 2 | 4 | 3 | 8 |

1. Draw a picture of the resulting hash-table after inserting the following words in order: dog, hare, ape, mule, bat, panda, cow, koala, goat
2. Identify cells that are probed when trying to find the word: bird.  
   Do (1) and (2) using each of the following techniques:
   1. (25 points) Linear probing with *h*1 as your hash function.
   2. (25 points) Double hashing with *h*1 as your first hash function and *h*2 as your second hash function.

**Problem 4 [Binary Search Variants].** (50 points)

Let *A* be an array of *n* distinct integers sorted in ascending order, and let *x* be an integer which may or may not be in *A*. Describe an algorithm running in *O*(log *n*) time that determines the **number** of integers in *A* that are strictly less than *x*. Analyze the running time of your algorithm to justify that it runs in *O*(log *n*) time.

**Problem 5** [Partial Sort]. (50 points)

Suppose the entering freshman class at some university has nstudents. The information pertinent to each student in the class (name, identification number, employment status, etc.) can be found in some element of A, an array of records indexed from 1 to n. Assume the records are in some random order, and that we wish to rearrange the array in-place so that all the unemployed student records precede all the employed student records. Describe and give the pseudocode for an in-place algorithm running in O(n)-time which performs this “partial sort" on A. Assume “status" is a field in each record, with value “employed" or “unemployed".

**“Challenging” (extra credit) problem (30 pts):** *A ‘true’ medieval story…* 

Sir Arthur de Templar was paid 27 gold coins for the information he provided on the last known location of the Holly Grail. Soon, from the highly trusted sources, he has learned that one of the 27 gold coins is not fully gold (it has some admixture of iron and copper and hence its weight differs (lighter or heavier) from the weight of the other 26 true gold coins). Sir Arthur paid a visit to his closest friend drug-maker Mr. Drugless and in a few seconds they identified the fake coin. Legend says that they used just an old weighing device of Mr. Drugless (depicted in Figure above) and could identify the fake coin in only 4 weightings. Recall that the fake coin maybe lighter or heavier (that is unknown) than the true-gold one.

1. Repeat the procedure used by Sir Arthur de Templar and Mr. Drugless for identifying the fake coin among 27 gold coins in just 4 weightings. (15 pts)

2. Extend it to an algorithm for identifying a single fake coin among *N* given gold coins using minimum number of weightings. How many weightings do you need? (15 pts)

(A weighting is an operation of putting *k* coins on one plate of the device and *k* other coins on the other plate (*k=1,2,3,…*) and checking if the device is in equilibrium, and if not, which plate is heavier, lighter.)