Banner number: Name:

Midterm Exam

CSCI 3110: Design and Analysis of Algorithms

June 23, 2014

Group 1		Group 2		Group 3		$\sum$
Question 1.1		Question 2.1		Question 3.1	☐	
Question 1.2		Question 2.2		Question 3.2	☐	
$\sum$		$\sum$		$\sum$		

Instructions:

- The questions are divided into three groups: Group 1 (18 marks = 36%), Group 2 (20 marks = 40%), and Group 3 (12 marks = 24%). You have to answer **all questions in Groups 1 and 2** and **exactly one question in Group 3**. In the above table, put a check mark in the **small** box beside the one question in Group 3 you want me to mark. If you select 0 or 2 questions in Group 3, I will mark neither.
- Provide your answer in the box after each question. If you absolutely need extra space, use the backs of the pages; but try to avoid it. Keey your answers short and to the point.
- **You are not allowed to use a cheat sheet.**
- If you are asked to design an algorithm and you cannot design one that achieves the desired running time, design a slower algorithm that is correct. A correct and slow algorithm earns you 50% of the marks for the algorithm. A fast and incorrect algorithm earns 0 marks.
- When designing an algorithm, you are allowed to use algorithms and data structures you learned in class as black boxes, without explaining how they work, as long as these algorithms and data structures do not directly answer the questions.
- **Read every question carefully before answering. In particular, do not waste time on an analysis if none is asked for, and do not forget to provide one if it is required.**
- **Do not forget to write your banner number and name on the top of this page.**
- **This exam has 8 pages, including this title page. Notify me immediately if your copy has fewer than 8 pages.**

Question 1.1 (Asymptotic growth of functions) — 9 marks

$O(f(n))$, $\Omega(f(n))$, $\Theta(f(n))$, $o(f(n))$, and $\omega(f(n))$ are sets of functions.

a. Define the set $O(f(n))$.

b. Define the set $\Theta(f(n))$.

c. Define the set $\omega(f(n))$.

Question 1.2 (Running time of algorithms) — 9 marks

a. Define what the **worst-case running time** of an algorithm is.

b. Define what the **average-case running time** of an algorithm is.

Question 2.1 (Asymptotic growth of functions) 10 marks

a Order the following functions by increasing order of growth:

$$(\lg n)^2 \quad n^{1/\lg n} \quad \frac{n}{\lg n} \quad 4^{\lg n} \quad \sqrt{n}$$

b Prove that you have arranged the second and third functions in the sorted sequence in the right order; that is, if the sorted sequence is $f_1(n), f_2(n), \ldots, f_5(n)$, prove that $f_2(n) \in o(f_3(n))$.

c Prove that $n \in o(n^2)$ without using limits.

Question 2.2 (Correctness proofs) **10 marks**

Euclid's algorithm can be used to find the greatest common divisor (GCD) of two integers:

```
EUCLID(a, b)
    // You may assume that both numbers are non-negative and that at least one is strictly positive.
    if a < b
        swap a and b
    if b == 0
        return a
    return EUCLID(a − b, b)
```

Prove that this algorithm is correct, that is, that it does indeed return the GCD of the two numbers.

Question 3.1 (Graph algorithms) **12 marks**

You are given a graph G each of whose edges is either red or blue and an integer parameter $k \geq 0$. Your task is to develop an algorithm that decides whether G has a spanning tree with exactly k red edges. (The algorithm does not have to find such a spanning tree. This can be done and is not particularly hard but requires more time to figure out than you have in this exam.) The running time of your algorithm should be $O(n+m)$, where n is the number of vertices of G and m is the number of edges of G. To make your task easier, here are the three parts of the answer you have to figure out:

(a) Argue that G has a spanning tree with exactly k red edges if and only if it has a spanning tree with at most k red edges and it has a spanning tree with at least k red edges.

(b) Argue that a minimum spanning tree of a graph whose edges have weight 0 or 1 can be found in $O(n+m)$ time.

(c) Use the linear-time minimum spanning tree algorithm for 0/1 edge weights to decide whether G has two spanning trees as in (a).

Extra space for Question 3.1

Question 3.2 (Greedy algorithms) **12 marks**

In class, we discussed the interval scheduling problem where we tried to maximize the number of non-overlapping intervals we can choose from a given set of intervals. One possible application I mentioned was that the intervals could represent the times when certain classes need to be held and we want to choose the largest subset of classes that can be held in the same classroom without holding two classes in the same classroom simultaneously. Now, simply cancelling the classes we cannot schedule in this fashion is not an option in practice. This leads us to the following problem: You are given a set of classes to be scheduled, each with a start time and an ending time. Your task is to find the minimum number, k, of classrooms such that all classes can be scheduled while ensuring that the time intervals of any two classes scheduled in the same classroom are disjoint. Your algorithm should compute a schedule of the given set of classes in k rooms and should run in $O(n \lg n + kn)$ time, where n is the number of classes provided as input and k is the number of classrooms you determine are needed. Prove that your algorithm does indeed find the minimum k such that it is possible to schedule all classes in these k classrooms.

Hint: Observe that you need at least k classrooms if there are k classes that need to be held simultaneously, that is, if there is a time t that is contained in the time intervals of k classes. Thus, your algorithm should have the property that, if it uses k classrooms to schedule all n classes, then there exists such a set of k simultaneously held classes.

Extra space for Question 3.2