

Banner number:

Name:

Midterm Exam
CSCI 3110: Design and Analysis of Algorithms
 June 26, 2008

Group 1		Group 2		Group 3		Σ
Question 1.1		Question 2.1		Question 3.1	<input type="checkbox"/>	
Question 1.2		Question 2.2		Question 3.2	<input type="checkbox"/>	
		Question 2.3				
Σ		Σ		Σ		

Instructions:

- The questions are divided into three groups. 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. Keep 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 9 pages, including this title page. Notify me immediately if your copy has fewer than 9 pages.**

Question 1.1 (Algorithm design paradigms)

10 marks

a. Explain what characterizes a greedy algorithm.

b. List the three main steps in a divide-and-conquer algorithm.

1.

2.

3.

Question 1.2 (Analysis of algorithms)

5 marks

a. Give the formal definition of the condition functions f and g have to satisfy so that $f(n) = o(g(n))$.

b. Now assume that $f(n) = o(g(n))$, that you have an algorithm A with running time $f(n)$ and an algorithm B with running time $g(n)$, and that you run both algorithms on the same, sufficiently large, input. Assume further that algorithm A runs on a slower computer than algorithm B . Which of the two algorithms will finish first? Justify your answer.

Question 2.1 (Asymptotic growth)

5 marks

a. Order the following functions by increasing order of growth:

$$n^2 \quad \lg n \quad n \lg n \quad \sqrt{n} \quad 2^{\lg n}$$

b. Prove that you have arranged the last two functions in the sorted sequence in the right order; that is, if the sorted sequence is $f_1(n), f_2(n), \dots, f_5(n)$, prove that $f_4(n) = o(f_5(n))$.

Question 2.2 (Recurrence relations)

5 marks

Solve the following recurrences using the Master theorem. To justify your answer, state which case applies and show that $n^{\log_b a}$ and $f(n)$ satisfy the conditions that need to be satisfied for this case to apply.

1. $T(n) = 3T(n/2) + \Theta(n^2)$

2. $T(n) = 4T(n/2) + \Theta(1)$

3. $T(n) = 3T(n/3) + \Theta(n)$

Question 2.3 (Correctness proofs)

10 marks

Consider the algorithm for merging two sorted sequences L and R to produce a new sequence S containing all their elements in sorted order:

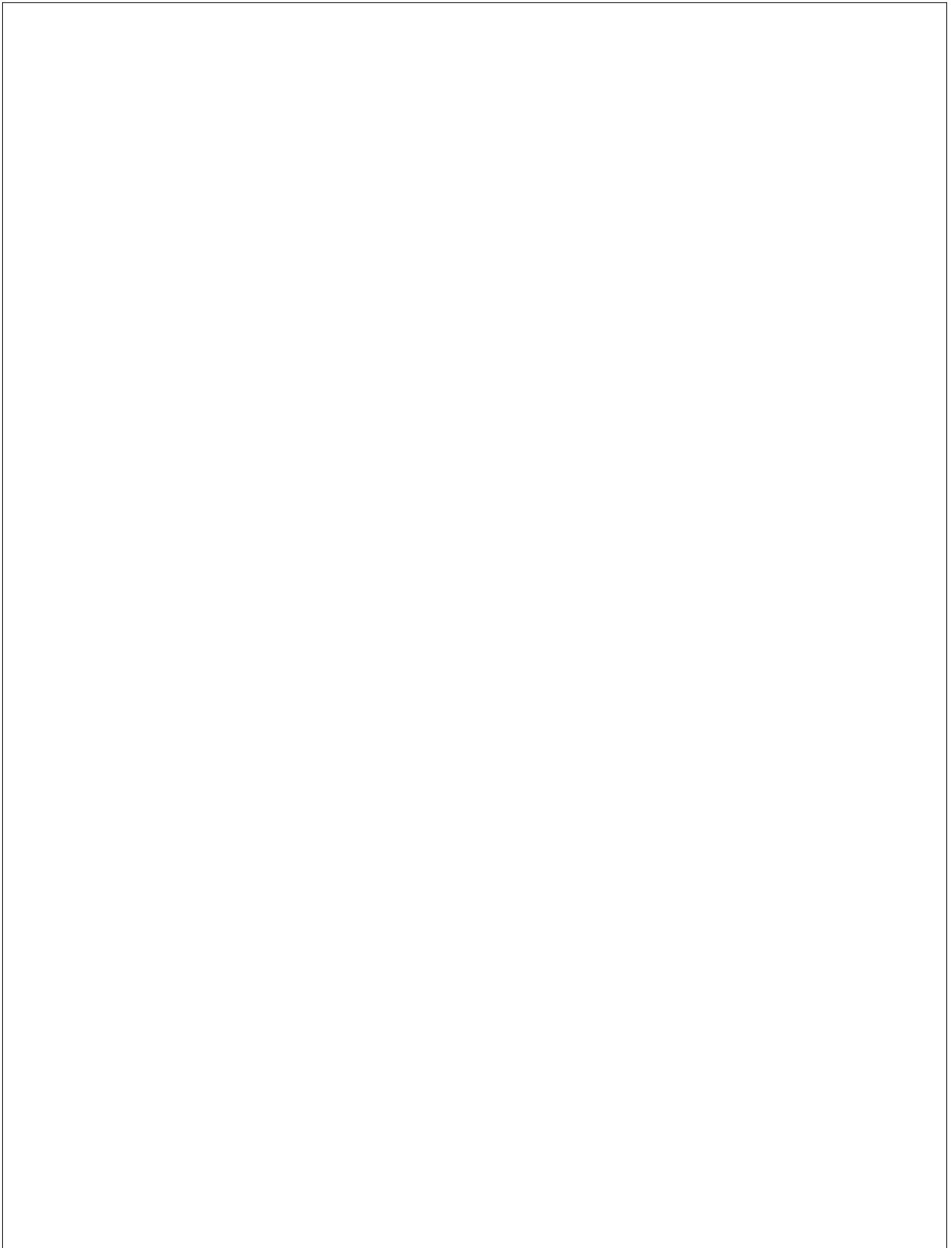
MERGE(L, R)

```
1   $S \leftarrow \emptyset$ 
2  while  $L \neq \emptyset$  and  $R \neq \emptyset$ 
3      do  $x \leftarrow$  first element of  $L$ 
4           $y \leftarrow$  first element of  $R$ 
5          if  $x < y$ 
6              then remove  $x$  from  $L$ 
7                  append  $x$  to  $S$ 
8              else remove  $y$  from  $R$ 
9                  append  $y$  to  $S$ 
10 if  $L \neq \emptyset$ 
11     then append  $L$  to  $S$ 
12     else append  $R$  to  $S$ 
13 return  $S$ 
```

Your task is to prove that this algorithm is correct, that is, that the returned sequence S contains the elements of $L \cup R$ in sorted order. While you are free to do this whichever way you want, one way to structure the proof is to prove the following claims by induction on the number of iterations of the while-loop executed so far and argue that they imply the correctness of the algorithm:

- (i) Let L_0 and R_0 denote the contents of L and R at the beginning of the algorithm. Then at any point in time $S \cup L \cup R = L_0 \cup R_0$.
- (ii) At any point in time, $x \leq y$ holds for all pairs (x, y) such that $x \in S$ and $y \in L \cup R$.
- (iii) At any point in time, S is sorted.

Extra space for Question 2.3

A large, empty rectangular box with a thin black border, intended for providing an answer to Question 2.3. The box occupies most of the page's vertical space.

Question 3.1 (Greedy algorithms)

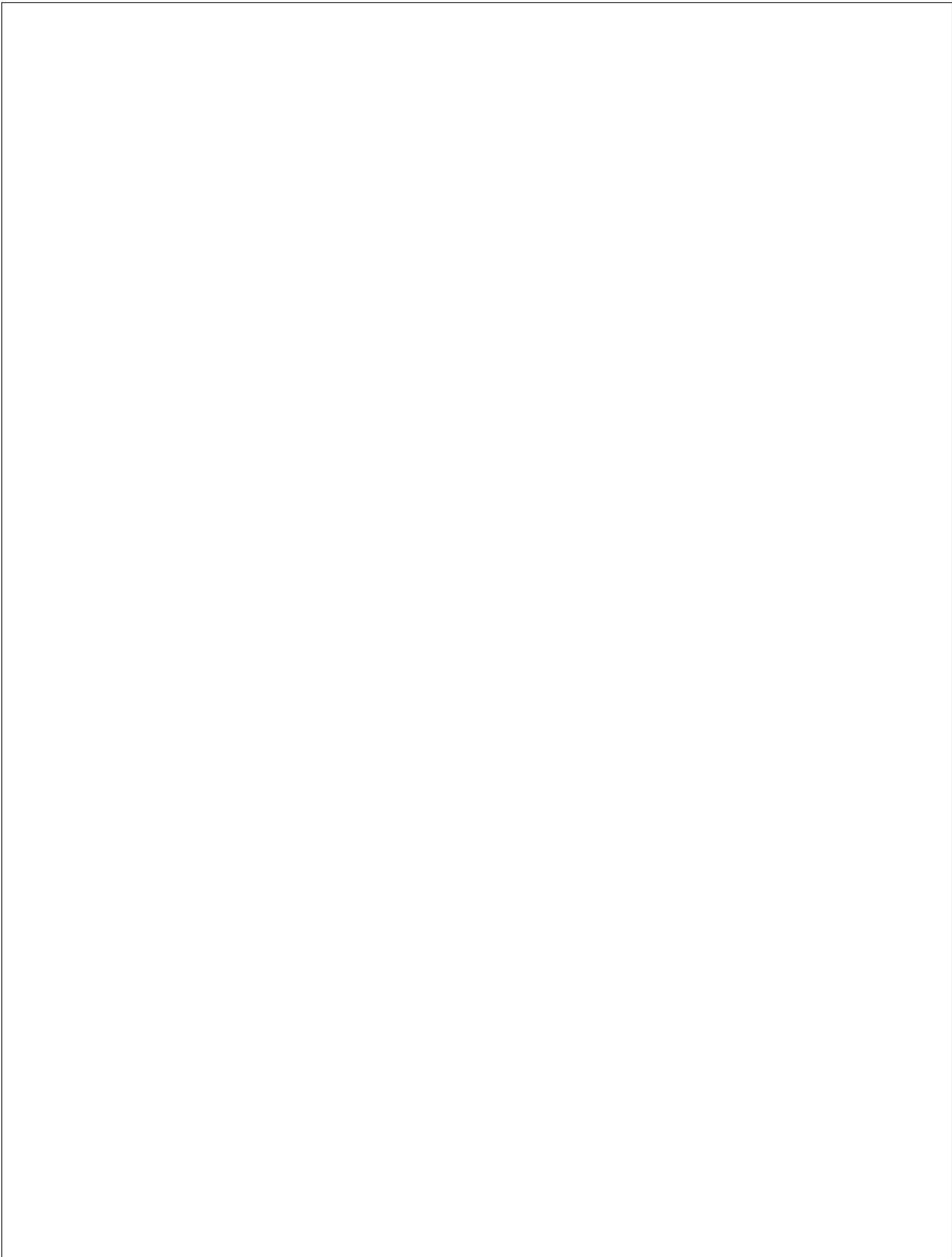
15 marks

You have just opened a new restaurant/café in Halifax. It's called "The Greedy Place". Your restaurant has a sidewalk patio. The patio is quite long, but the sidewalk is narrow; so the tables on the patio are essentially arranged in a straight line. Since the restaurant is in Halifax, you absolutely need to cover the tables with umbrellas to allow your customers to sit outside in spite of the 24/7 rain. Your goal is to cover all tables with as few umbrellas as possible. Here are the constraints and a few simplifying assumptions:

- Every umbrella has a diameter of 8'.
- You may assume that the tables are points; that is, a table is covered when the point where it is placed is contained in the interval covered by an umbrella.
- Assume that the table positions p_1, p_2, \dots, p_n are given by increasing distance from one end of the patio. Let us call this the left-to-right order.

Develop a greedy algorithm that determines a covering of the tables with a minimum number of umbrellas. Your algorithm should take linear time. Prove that the algorithm finds a covering with the minimum number of umbrellas.

Extra space for Question 3.1



Question 3.2 (Divide and conquer)

15 marks

You are on the team of a TV station covering the Tour de France. You would like to provide your audience with different kinds of statistics about each stage of the tour. This includes detailed information about the profile of the stage. One of the pieces of information about the profile you want to provide is the highest elevations the cyclists have to master in different parts of each stage. So you are given the profile of the stage, consisting of n heights, h_1, h_2, \dots, h_n , sampled at a distance of 1km along the route; and you are given m parts P_1, P_2, \dots, P_m of today's stage. Each part P_j is a pair (f_j, t_j) indicating that this part starts at kilometer f_j and ends at kilometer t_j . For each part P_j , you want to report the highest elevation between kilometer f_j and kilometer t_j . Develop a divide-and-conquer algorithm that does this in $O((n + m) \lg n)$ time. Argue that the running time of your algorithm is what you claim.

Extra space for Question 3.2

