Midterm Examination

CS 5114 (Spring 2009)

Assigned: March 19, 2009.
Due: at the beginning of class on March 26, 2009.

Name: ____________________

PID: ____________________

Instructions

1. For every algorithm you describe, prove its correctness and analyse its running time. I am looking for clear descriptions of algorithms and for the most efficient algorithms and analysis that you can come up with. I am not specifying how optimal I want each algorithm to be. I will give partial credit to non-optimal algorithms, as long as they are correct.

2. You may consult the textbook, your notes, or the course web site to solve the problems in the examination. You **may not** work on the exam with anyone else, ask anyone questions, or consult other textbooks or sites on the Web for answers. **Do not use** concepts from Chapters 7 and later in the textbook.

3. You must prepare your solutions digitally and submit a hard-copy.

4. I prefer that you use LaTeX to prepare your solutions. However, I will not penalise you if you use a different system. To use LaTeX, you may find it convenient to download the LaTeX source file for this document from the link on the course web site. At the end of each problem are three commented lines that look like this:

```
% \solution{
%
% }
```

   You can uncomment these lines and type in your solution within the curly braces.

5. Do not forget to staple the hard copy you hand in.

Good luck!

Problem 1 (25 points) Solve the following recurrence relation. You can assume that $T(1) = 0$ and $T(2) = 1$ and c is a positive constant.

$$T(n) = \frac{1}{n} \sum_{i=1}^{n-1} \big(T(i) + T(n-i)\big) + cn.$$

Hint: Unrolling the recurrence may lead to messy expressions. You could guess the solution and prove the guess is correct using induction. To try to prove it directly, compute $nT(n) - (n-1)T(n-1)$ and see if you notice a pattern.

Problem 2 (10 points) You work for a company that operates gas stations. Your company wants to place one gas station on a long country road so as to best serve all the houses on that road. For convenience, assume that the road is a straight line running west to east and that the position of each house on that road is given by the x-coordinate of the house. Suppose there are n houses with x-coordinates $x_1, x_2, \ldots, x_n$ (these coordinates may not be in sorted order). There are various ways to measure the distance from each house to the gas station. Your company wants to find the best placement of the gas station for different cost models. Suppose μ is a proposed location for the gas station. Let $d(r, \mu)$ denote the distance between a house at x-coordinate r and the gas station. Let $C(\mu)$ denote the cost of travelling to the gas station (over all the houses). Devise an algorithm to compute the value of μ that minimises $C(\mu)$ under each of the following definitions of $d(r, \mu)$ and $C(\mu)$ (thus, you should provide two algorithms in total):

(i) $d(r, \mu) = (r - \mu)^2$ and $C(\mu) = \sum_{i=1}^{n} d(x_i, \mu)$.

(ii) $d(r, \mu) = |r - \mu|$ and $C(\mu) = \max_{i=1}^{n} d(x_i, \mu)$.

Problem 3 (25 points) We say that one two-dimensional point $p = (p_x, p_y)$ *looks down on* another two-dimensional point $q = (q_x, q_y)$ if $p_x \geq q_x$ and $p_y \geq q_y$. For example, the point $(2, 10)$ looks down upon the point $(-5, 8)$ but not upon the point $(-5, 12)$. In a set P of n two-dimensional points, a point p is said to be *majestic* if no point in P looks down upon p. Devise an efficient algorithm to compute all majestic points in P.

Problem 4 (20 points) The curriculum in the Department of Computer Science at the University of Draconia consists of n courses. Each course is mandatory. The prerequisite graph G has a node for each course, and an edge from course v to course w if and only if v is a prerequisite for w. In such a case, a student cannot take course w in the same or an earlier semester than she takes course v. A student can take any number of courses in a single semester. Design an algorithm that computes the minimum number of semesters necessary to complete the curriculum. You may assume that G does not contain cycles, i.e., it is a directed acyclic graph. Chapter 3.6 of the textbook discusses directed acyclic graphs. The graph can be split into multiple components. For example, an extreme case is when there are no pre-requisites, in which case each course is in a separate component. Of course, one semester suffices in this trivial case.

If you are concerned about how G is represented, assume that for each course, you have a list of courses for which it is a pre-requisite (the adjacency list representation). If you need, you can assume that the "list" for each course is stored in an array. if you want to use another representation, please describe it. *Do not use an adjacency matrix representation.*

Problem 5 (20 points) Many object-oriented programming language implement a class for manipulating strings. A primitive operation supported by such languages is to split a string into two pieces. This operation usually involves copying the original string. Hence, it takes n units of time for a string of length n, *regardless of the location of the split.* However, if we want to split a string into many pieces, the order in which we make the breaks can affect the total running time of all the splits.

For example, suppose we want to split a 20-character string at positions 3 and 10. If we make the first cut at position 3, the cost of the first cut is the length of the string, which is 20. Now the cut at position 10 falls within the second string, whose length is 17, so the cost of the second cut is 17.

Therefore, the total cost is 20 + 17 = 37. Instead, if we make the first cut at position 10, the cost of this cut is still 20. However, the second cut at position 3 falls within the first string, which has length 10. Therefore, the cost of the second cut is 10, implying a total cost of 20 + 10 = 30.

Design an algorithm that, given the locations of m cuts in a string of length n, finds the minimum cost of breaking the string into $m + 1$ pieces at the given locations.