Analysis of Algorithms: Exam 1

February 11, 1999

The exam includes nine regular problems, 10 points each, and a bonus problem. The length of the exam is 70 minutes (11:05 to 12:15).

Sign your name (10 points):

Problem 1 (10 points)

(a) Give a precise mathematical definition of a tight asymptotic bound, $\Theta(g(n))$.

(b) Explain the difference between $O(g(n))$ and $\Omega(g(n))$.

Problem 2 (10 points)

Give the asymptotic (Θ-notation) running times of the following three algorithms. You do *not* need to explain how you determined the running times; just give the answers.

(a) Factorial computation.

$\textsc{Factorial}(n)$
if $n = 0$
  then **return** 1
  else **return** $n \cdot \textsc{Factorial}(n-1)$

(b) Repetitive division by three.

$\textsc{Division}(n)$
while $n > 1$
  do $n \leftarrow \lfloor n/3 \rfloor$

(c) Repetitive square root.

$\textsc{Square-Roots}(n)$
while $n > 2$
  do $n \leftarrow \lfloor \sqrt{n} \rfloor$

Problem 3 (10 points)

Give asymptotic (Θ-notation) solutions to each of the following recurrences. You do *not* need to show the derivation of your solutions; just write the answers.

(a) $T(n) = T(n-1) + n =$

(b) $T(n) = 2T(n/2) + n =$

(c) $T(n) = T(n/2) + n =$

(d) $T(n) = 4T(n/2) + n =$

(e) $T(n) = 2T(n/2) + n^2 =$

Problem 4 (10 points)

Give the *worst-case* time complexity for each of the following sorting algorithms:

(a) Insertion Sort

(b) Merge-Sort

(c) Quick-Sort

(d) Heap-Sort

(e) Counting Sort

Problem 5 (10 points)

(a) Draw a heap that contains the following eight numbers: 1, 2, 3, 4, 5, 6, 7, 8.

(b) Suppose that you apply the Heap-Extract-Max algorithm to your heap. What does the algorithm return? Draw the resulting new heap.

Problem 6 (10 points)

Suppose that we apply the PARTITION algorithm (which is a subroutine of QUICK-SORT) to the following array:

6 4 1 8 5 0 4 9

Show the resulting order of elements in the array, after the application of PARTITION.

Problem 7 (10 points)

Suppose that you work with a database that includes a list of several thousand people and ages of these people; the age of each person is specified as an integer number of years. If you need to sort this list by the ages of people, what sorting algorithm will you use? Why? What is the time complexity (Θ-notation) of this algorithm?

Problem 8 (10 points)

Show the main steps of Radix-Sort (like I did in class) on the following array of three-digit numbers:

117
548
648
927
525
611
143

Problem 9 (10 points)

(a) Give the definition of the Binary-Search-Tree (BST) Property.

(b) Draw a binary search tree that contains the following eight numbers: 1, 2, 3, 4, 5, 6, 7, 8.

Problem 10 (bonus)

This problem is optional and does not affect your grade for the exam; if you solve it, then you get 5 bonus points toward your final grade for the course.

Suppose that you are using a programming language that allows only integer numbers and supports four operations on them: addition, subtraction, multiplication, and integer division; the running time of each operation is constant, that is, $\Theta(1)$. The result of the integer division of n over m is $\lfloor n/m \rfloor$; for example, $\lfloor 10/3 \rfloor = 3$. The language does *not* allow fractional numbers, and does *not* have operations for logarithms and exponentiation.

Write an efficient algorithm $\textsc{Power}(n, m)$ that computes n^m, where n and m are positive integers, and give the asymptotic time complexity (Θ-notation) of your algorithm. Note that computing n^m in $\Theta(m)$ time is *too slow,* and an algorithm with this running time will get you only 1 bonus point; try to design a faster algorithm.