Duration: **50 minutes**
Aids Allowed: **NONE** (in particular, no calculator)

Student Number: |_|_|_|_|_|_|_|_|_|

Last (Family) Name: ____________________

First (Given) Name(s): ____________________

Tutorial Section: (cirle one)	UC–87 Hamed Hatami	UC–144 Tobi Kral	UC–244 Ady Ecker	UC–A101 Alfredo Gabaldon	LM–161 Stephanie Horn

*Do **not** turn this page until you have received the signal to start.*
(In the meantime, please fill out the identification section above,
and read the instructions below *carefully*.)

This term test consists of 3 questions on 7 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete.*

Answer each question directly on the test paper, in the space provided, and use the reverse side of the pages for rough work. If you need more space for one of your solutions, use the reverse side of a page and *indicate clearly the part of your work that should be marked.*

In your answers, you may use without proof any fact covered in lecture, tutorial, or on assignments. You must justify all other facts required for your solution.

If you are unable to answer a question (or part of a question), you will get 20% of the marks for the question (or part of the question) if you state clearly that you do not know how to answer. Note that you will *not* get those marks if your answer contains contradictory statements (such as "I do not know how to answer" followed or preceded by parts of a solution that have not been crossed off).

General Hint: We were careful to leave ample space on the test paper to answer each question.

Marking Guide

1: ______/10

2: ______/20

3: ______/20

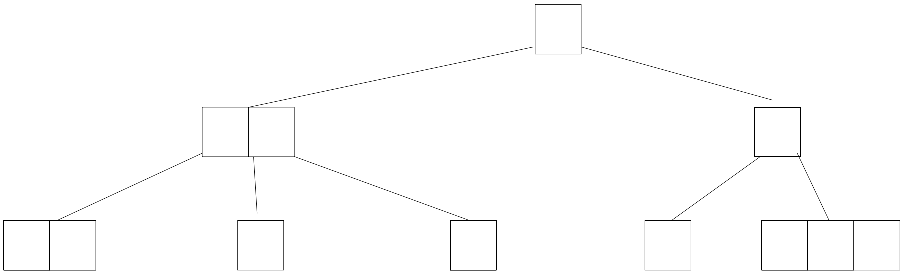
TOTAL: ______/50

Good Luck!

Question 1. [10 marks]

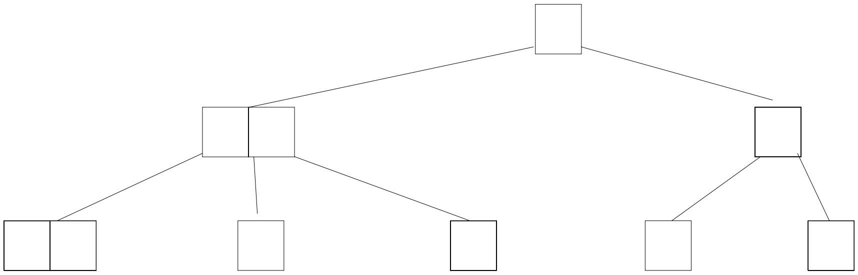
Part (a) [2 marks]

Draw the resulting 2-3-4 tree after performing DELETE(R) on the tree below.



Part (b) [2 marks]

Draw the resulting 2-3-4 tree after performing DELETE(U) on the tree below.



Question 1. (CONTINUED)

Part (c) [6 MARKS]

Show that it is possible to perform only 10 insertions into an empty 2-3-4 tree to get a resulting tree of height 3. (For example, the trees on the previous page have height 3.)

Question 2. [20 marks]

Consider the following algorithm that computes the number of 1's in a one-dimensional array A that contains some number of 1's (possibly none) followed by some number of 0's (possibly none).

The idea of the algorithm is to break up A into approximately $\sqrt{n}$ sections, each one of length approximately $\sqrt{n}$ (where $n = \text{length}(A)$), and to perform the search in two phases: first look at the last element in each section to determine the last section that contains 1's, then look at each element in that section to determine the position of the last 1 in A.

```
NumOfOnes($A$) :
    $m := \left\lceil \sqrt{\text{length}(A)} \right\rceil$
    $p := m - 1$
    while $p < \text{length}(A)$ and $A[p] = 1$ :
        $p := p + m$
    $p := p - m + 1$
    while $p < \text{length}(A)$ and $A[p] = 1$ :
        $p := p + 1$
    return $p$
```

For each part of this question, measure complexity by counting the number of array accesses. To simplify your analysis, assume that the length of A is a perfect square *i.e.*, $\text{length}(A) = m^2$ for some $m \in \mathbb{N}$.

Part (a) [2 marks]

Perform a best-case analysis of the NumOfOnes algorithm.

Part (b) [6 marks]

Perform a worst-case analysis of the NumOfOnes algorithm.

Question 2. (CONTINUED)

Part (c) [12 MARKS]

Perform an average-case analysis of the NUMOFONES algorithm. Assume that A is equally likely to contain any number of 1's (anywhere from 0 to $n = \text{length}(A)$).
(HINT: Let R_i denote the probability that element $A[i]$ is accessed, for $0 \leq i < \text{length}(A)$. Compute your answer as a function of the R_i's, then compute R_i separately by considering two cases: when $i = km - 1$ for some $k \in \mathbb{N}$, and when it is not. You do not have to simplify your final answer.)

Question 3. [20 marks]

For this question, you will augment 2-3-4 trees to perform one new operation, NumGreater(k): return the number of elements in the tree that are strictly greater than k (for an arbitrary key k that may or may not be in the tree).

Part (a) [4 marks]

Describe any new information that will be stored in the tree.

Part (b) [6 marks]

Describe how to maintain the new information during the regular INSERT operation. Briefly justify that the performance of INSERT is still $\mathcal{O}(\log n)$.

Question 3. (CONTINUED)

Part (c) [10 MARKS]

Describe how to perform operation NUMGREATER(k) using the new information. Briefly justify that the worst-case performance of your operation is $\mathcal{O}(\log n)$.

Total Marks = 50