Final Exam

- Do not open this exam booklet until you are directed to do so. Read all the instructions on this page.

- When the exam begins, write your name on every page of this exam booklet.

- This exam contains 11 problems, some with multiple parts. You have 180 minutes to earn 180 points.

- This exam booklet contains 20 pages, including this one. Two extra sheets of scratch paper are attached. Please detach them before turning in your exam at the end of the examination period.

- This exam is closed book. You may use two handwritten A4 or $8\frac{1}{2}'' \times 11''$ crib sheets. No calculators or programmable devices are permitted.

- Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem. Do not put part of the answer to one problem on the back of the sheet for another problem, since the pages may be separated for grading.

- Do not waste time and paper rederiving facts that we have studied. It is sufficient to cite known results.

- Do not spend too much time on any one problem. Read them all through first, and attack them in the order that allows you to make the most progress.

- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Be neat.

- Good luck!

Name: _____

Grade sheet: Do not write in the spaces below.

Problem	*Title*	*Parts*	*Points*	*Grade*	*Grader*
1	Recurrences	3	15		
2	Algorithms and running times	1	9		
3	Substitution method	1	10		
4	True or False, and Justify	7	35		
5	Red-black trees	3	15		
6	Wiggly arrays	1	10		
7	Difference constraints	2	12		
8	Amortized increment	1	12		
9	Minimum spanning trees	1	12		
10	Multithreaded scheduling	1	10		
11	Transposing a matrix	4	40		
	Total		180		

Problem 1.   Recurrences [15 points] (3 parts)

Give a tight asymptotic upper bound (O notation) on the solution to each of the following recurrences. You need not justify your answers.

(a) $T(n) = 2T(n/8) + \sqrt[3]{n}$.

(b) $T(n) = T(n/3) + T(n/4) + 5n$

(c) $T(n) = \begin{cases} 8T(n/2) + \Theta(1) & \text{if } n^2 > M, \\ M & \text{if } n^2 \leq M; \end{cases}$

where M is a variable independent from n.

Problem 2. Algorithms and running times [9 points]

Match each algorithm below with the tightest asymptotic upper bound for its worst-case running time by inserting one of the letters A, B, . . ., I into the corresponding box. For sorting algorithms, n is the number of input elements. For matrix algorithms, the input matrix has size $n \times n$. For graph algorithms, the number of vertices is n, and the number of edges is $\Theta(n)$.

You need not justify your answers. Some running times may be used multiple times or not at all. Because points will be deducted for wrong answers, do not guess unless you are reasonably sure.

☐   Insertion sort   A: $O(\lg n)$

☐   Heapsort   B: $O(n)$

☐   BUILD-HEAP   C: $O(n \lg n)$

☐   Strassen's   D: $O(n^2)$

☐   Bellman-Ford   E: $O(n^2 \lg n)$

☐   Depth-first search   F: $O(n^{2.5})$

☐   Floyd-Warshall   G: $O(n^{\lg 7})$

☐   Johnson's   H: $O(n^3)$

☐   Prim's   I: $O(n^3 \lg n)$

Problem 3.   Substitution method [10 points]

Use the substitution method to prove a tight asymptotic lower bound (Ω-notation) on the solution to the recurrence

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

Problem 4. True or False, and Justify [35 points] (7 parts)

Circle **T** or **F** for each of the following statements to indicate whether the statement is true or false, respectively. If the statement is correct, briefly state why. If the statement is wrong, explain why. The more content you provide in your justification, the higher your grade, but be brief. Your justification is worth more points than your true-or-false designation.

T F Let A_1, A_2, and A_3 be three sorted arrays of n real numbers (all distinct). In the comparison model, constructing a balanced binary search tree of the set $A_1 \cup A_2 \cup A_3$ requires $\Omega(n \lg n)$ time.

T F Let T be a complete binary tree with n nodes. Finding a path from the root of T to a given vertex $v \in T$ using breadth-first search takes $O(\lg n)$ time.

T F Given an unsorted array $A[1 \mathinner{.\,.} n]$ of n integers, building a max-heap out of the elements of A can be performed asymptotically faster than building a red-black tree out of the elements of A.

T F Suppose we use a hash function h to hash n distinct keys into an array T of length m. Assuming simple uniform hashing, the expected number of colliding pairs of elements is $\Theta(n^2/m)$.

T F Every sorting network with n inputs has depth $\Omega(\lg n)$.

T F If a dynamic-programming problem satisfies the optimal-substructure property, then a locally optimal solution is globally optimal.

T F Let $G = (V, E)$ be a directed graph with negative-weight edges, but no negative-weight cycles. Then, one can compute all shortest paths from a source $s \in V$ to all $v \in V$ faster than Bellman-Ford using the technique of reweighting.

Problem 5. Red-black trees [15 points] (3 parts)

(a) Assign the keys $2, 3, 5, 7, 11, 13, 17, 19$ to the nodes of the binary search tree below so that they satisfy the binary-search-tree property.
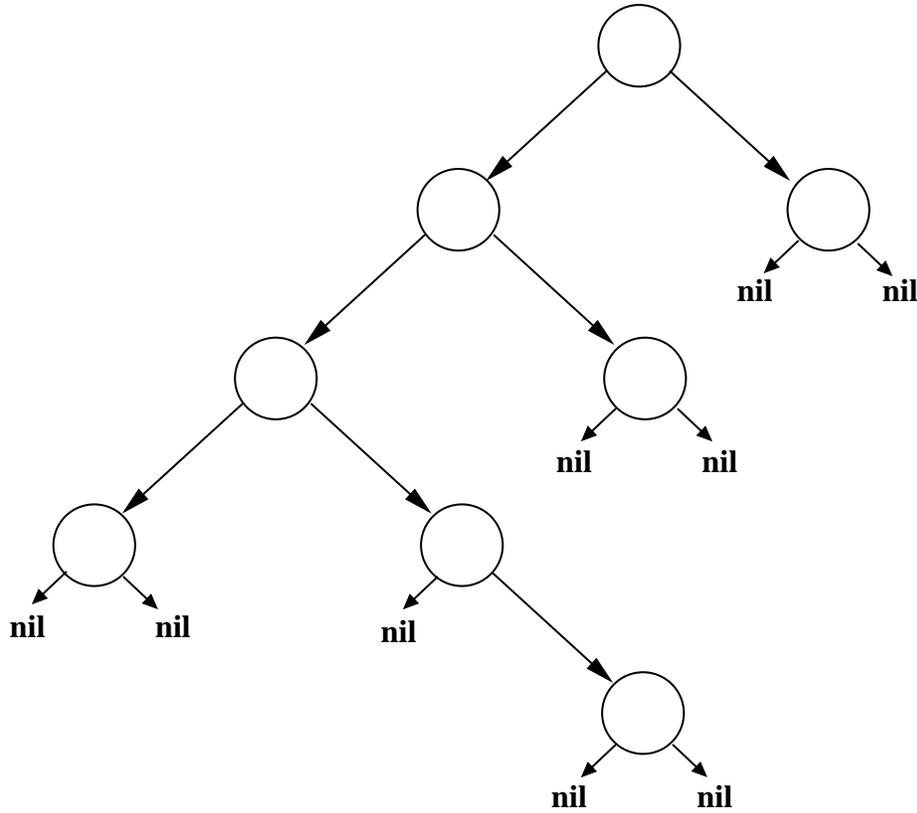
(b) Explain why this binary search tree cannot be colored to form a legal red-black tree.

(c) The binary search tree can be transformed into a red-black tree by performing a single rotation. Draw the red-black tree that results, labeling each node with "red" or "black." Include the keys from part (a).

Problem 6.   Wiggly arrays [10 points]

An array $A[1 .. 2n + 1]$ is **wiggly** if $A[1] \leq A[2] \geq A[3] \leq A[4] \geq \ldots \leq A[2n] \geq A[2n + 1]$. Given an unsorted array $B[1 .. 2n+1]$ of real numbers, describe an efficient algorithm that outputs a permutation $A[1 .. 2n + 1]$ of B such that A is a wiggly array.

Problem 7. Difference constraints [12 points] (2 parts)

Consider the following linear-programming system of difference constraints (note that one constraint is an equality):

$$
\begin{aligned}
x_1 - x_4 &\leq -1 \\
x_1 - x_5 &\leq -4 \\
x_2 - x_1 &\leq -4 \\
x_2 - x_3 &= -9 \\
x_3 - x_1 &\leq 5 \\
x_3 - x_5 &\leq 2 \\
x_4 - x_3 &\leq -3 \\
x_5 - x_1 &\leq 5 \\
x_5 - x_4 &\leq 1
\end{aligned}
$$

(a) Draw the constraint graph for these constraints.

(b) Solve for the unknowns x_1, x_2, x_3, x_4, and x_5, or explain why no solution exists.

Problem 8. Amortized increment [12 points]

An array $A[0 \mathinner{\ldotp\ldotp} k-1]$ of bits (each array element is 0 or 1) stores a binary number $x = \displaystyle\sum_{i=0}^{k-1} A[i] \cdot 2^i$.

To add 1 (modulo 2^k) to x, we use the following procedure:

Increment(A, k)
1 $i \leftarrow 0$
2 **while** $i < k$ and $A[i] = 1$
3   **do** $A[i] \leftarrow 0$
4           $i \leftarrow i + 1$
5 **if** $i < k$
6   **then** $A[i] \leftarrow 1$

Given a number x, define the potential $\Phi(x)$ of x to be the number of 1's in the binary representation of x. For example, $\Phi(19) = 3$, because $19 = 10011_2$. Use a potential-function argument to prove that the amortized cost of an increment is $O(1)$, where the initial value in the counter is $x = 0$.

Problem 9.   Minimum spanning trees [12 points]

Let $G = (V, E)$ be a connected, undirected graph with edge-weight function $w : E \to \mathbb{R}$, and assume all edge weights are distinct. Consider a cycle $\langle v_1, v_2, \ldots, v_k, v_{k+1} \rangle$ in G, where $v_{k+1} = v_1$, and let (v_i, v_{i+1}) be the edge in the cycle with the largest edge weight. Prove that (v_i, v_{i+1}) does *not* belong to the minimum spanning tree T of G.

Problem 10. Multithreaded scheduling [10 points]

A greedy scheduler runs a multithreaded computation in 260 seconds on 4 processors and in 90 seconds on 32 processors. Is it possible that the computation has work $T_1 = 1024$ and critical-path length $T_\infty = 64$? Justify your answer.

Problem 11.   Transposing a matrix [40 points] (4 parts)

Let X be an $N \times N$ matrix, where N is an exact power of 2. The following code computes $Y = X^\mathsf{T}$:

$\text{TRANS}(X, Y, N)$
1 **for** $i \leftarrow 1$ **to** N
2   **do for** $j \leftarrow 1$ **to** N
3   **do** $Y[j, i] \leftarrow X[i, j]$

Consider the cache-oblivious two-level memory model with a cache of M elements and blocks of B elements. Assume that both matrices X and Y are stored in row-major order, that is, the linear order of X in memory is $X[1, 1], X[1, 2], \ldots, X[1, N], X[2, 1], X[2, 2], \ldots, X[2, N], \ldots, X[N, 1], X[N, 2], \ldots, X[N, N]$, and similarly for Y.

(a) Analyze the number $MT(N)$ of memory transfers incurred by TRANS when $N \gg M$.

Now, consider the following divide-and-conquer algorithm for computing the transpose:

R-TRANS(X, Y, N)
1   **if** $N = 1$
2   **then** $Y[1, 1] \leftarrow X[1, 1]$
3   **else** Partition X into four $(N/2) \times (N/2)$ submatrices X_{11}, X_{12}, X_{21}, and X_{22}.
4   Partition Y into four $(N/2) \times (N/2)$ submatrices Y_{11}, Y_{12}, Y_{21}, and Y_{22}.
5   R-TRANS$(X_{11}, Y_{11}, N/2)$
6   R-TRANS$(X_{12}, Y_{21}, N/2)$
7   R-TRANS$(X_{21}, Y_{12}, N/2)$
8   R-TRANS$(X_{22}, Y_{22}, N/2)$

Assume that the cost of partitioning is $O(1)$.

  (b) Give and solve a recurrence for the number $MT(N)$ of memory transfers incurred by
       R-TRANS when $N \gg M$.

The following multithreaded algorithm computes the transpose in parallel:

P-TRANS(X, Y, N)
1 **if** $N = 1$
2   **then** $Y[1, 1] \leftarrow X[1, 1]$
3   **else** Partition X into four $(N/2) \times (N/2)$ submatrices X_{11}, X_{12}, X_{21}, and X_{22}.
4   Partition Y into four $(N/2) \times (N/2)$ submatrices Y_{11}, Y_{12}, Y_{21}, and Y_{22}.
5   **spawn** P-TRANS$(X_{11}, Y_{11}, N/2)$
6   **spawn** P-TRANS$(X_{12}, Y_{21}, N/2)$
7   **spawn** P-TRANS$(X_{21}, Y_{12}, N/2)$
8   **spawn** P-TRANS$(X_{22}, Y_{22}, N/2)$
9   **sync**

  (c) Give and solve recurrences describing the work $T_1(N)$ and critical-path length $T_\infty(N)$ of P-TRANS. What is the asymptotic parallelism of the algorithm?

Professor Kellogg inadvertantly places two additional **sync** statements into his code as follows:

K-TRANS(X, Y, N)
1   **if** $N = 1$
2   **then** $Y[1, 1] \leftarrow X[1, 1]$
3   **else** Partition X into four $(N/2) \times (N/2)$ submatrices X_{11}, X_{12}, X_{21}, and X_{22}.
4   Partition Y into four $(N/2) \times (N/2)$ submatrices Y_{11}, Y_{12}, Y_{21}, and Y_{22}.
5   **spawn** K-TRANS$(X_{11}, Y_{11}, N/2)$
6   **sync**
7   **spawn** K-TRANS$(X_{12}, Y_{21}, N/2)$
8   **sync**
9   **spawn** K-TRANS$(X_{21}, Y_{12}, N/2)$
10   **spawn** K-TRANS$(X_{22}, Y_{22}, N/2)$
11   **sync**

(d) Give and solve recurrences describing the work $T_1(N)$ and critical-path length $T_\infty(N)$ of K-TRANS. What is the asymptotic parallelism of this algorithm?

SCRATCH PAPER — Please detach this page before handing in your exam.

SCRATCH PAPER — Please detach this page before handing in your exam.