• This exam is **closed book, closed notes**, 50 minutes long

• Read the questions carefully

• No electronic equipment allowed (cell phones, tablets, computers, ...)

• Write legibly. What can’t be read will not be graded

• Use pseudocode, Python, or English to describe your algorithms (no C/C++/Java)

• When designing an algorithm, you are allowed to use any algorithm or data structure we explained in CS 141 or CS 14, without giving its details, unless the question specifically requires that you give such details

• Always remember to analyze the time complexity of your solution

• If you have a question about the meaning of a question, come to the front of the class
Problem 1. (25 points [writing/solving recurrence relations])

Solve ONE (NOT both) of the following problems. You are NOT required to verify the correctness of your solution.

1. Solve exactly (that is, without using any asymptotic notation) the following recurrence relation by iterative substitutions

\[ T(n) = \begin{cases} 
1 & n = 1 \\
3T\left(\frac{n}{2}\right) + 2 & n > 1 
\end{cases} \]

2. Using the Master method, give an asymptotic tight bound for \( T(n) \) in the following recurrence relation

\[ T(n) = \begin{cases} 
1 & n = 1 \\
T\left(\frac{n}{2}\right) + n \log_2 n & n > 1 
\end{cases} \]
Problem 2. (25 points [writing/solving recurrence relations])
Consider the following variant for the closest-pair algorithm we discussed in class.

Algorithm Closest-Pair (A : array of 2D points)
  if |A| ≤ 2 then return the pair in O(1) time
  sort set A by x coordinate
  $d_L \leftarrow$ Closest-Pair($A[1 \ldots n/2]$)
  $d_R \leftarrow$ Closest-Pair($A[n/2 + 1 \ldots n]$)
  $d \leftarrow \min(d_L, d_R)$
  $S \leftarrow \{ p \in A : |p.x - A[n/2].x| \leq d \}$ be the points in the vertical strip around the median
  $dS \leftarrow$ the closest-pair distance of points in $S$, computed in $O(n \log n)$ time
  return $\min(d, dS)$

Assume each basic operation on points (computing distance, comparing coordinates, ...) takes $O(1)$ time. Define $T(n)$ to be the worst-case running time of Closest_Pair($A$) on any array $A$ of $n$ points. Write a recurrence relation for $T(n)$ and give a tight bound on $T(n)$. 
Problem 3. (25 points [divide & conquer])

Another version of Strassen’s algorithm uses the following identities. To compute

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix} = \begin{bmatrix}
E & F \\
G & H
\end{bmatrix} \begin{bmatrix}
I & J \\
K & L
\end{bmatrix}
\]

first compute the following values:

\[
\begin{align*}
  s_1 &= G + H & m_1 &= s_2 s_6 \\
  s_2 &= s_1 - E & m_2 &= EI \\
  s_3 &= E - G & m_3 &= FK \\
  s_4 &= F - s_2 & m_4 &= s_3 s_7 \\
  s_5 &= J - I & m_5 &= s_1 s_5 \\
  s_6 &= L - s_5 & m_6 &= s_4 L \\
  s_7 &= L - J & m_7 &= H s_8 \\
  s_8 &= s_6 - K
\end{align*}
\]

then

\[
\begin{align*}
  A &= m_2 + m_3 \\
  B &= m_1 + m_2 + m_5 + m_6 \\
  C &= m_1 + m_2 + m_4 - m_7 \\
  D &= m_1 + m_2 + m_4 + m_5
\end{align*}
\]

1. Analyze the time complexity \( T(n) \) of this new algorithm assuming that the input are two \( n \times n \) matrices

2. Compare \( T(n) \) against that of the original Strassen. Is it faster, slower, or the same?
Problem 4. (25 points [divide & conquer])

You are given an unsorted array $A[1 \ldots n]$ of $n$ distinct integers. We say that $A[i]$ is a local maximum if $A[i]$ is bigger than its neighbors, that is, $A[i] > A[i - 1]$ (if $i \neq 1$) and $A[i] > A[i + 1]$ (if $i \neq n$). For instance in $A = \{3, 4, 1, 2, 5, 6, 0, 8, 9\}$, 4 is a local maximum, 6 is a local maximum, and 9 is a local maximum. Give an $O(\log n)$-time algorithm to find any of the local maximum in $A$. If there is more than one local maximum in $A$, we are OK with any one of them. Explain briefly how the algorithm works, and why it runs on $O(\log n)$ time. You can assume that $n$ is a power of two. Hint: If $A[i]$ is not a local maximum because $A[i] < A[i + 1]$, then must there be some local maximum $A[j]$ with $j > i$?