This exam is **closed book, closed notes**, 80 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.** (15 points [analysis of iterative pseudocode])

Give a tight bound (using the big-theta notation) on the number of Hi’s produced by the following method as a function of $n$. For simplicity, you can assume $n$ to be a power of two.

```
Algorithm LOOP1 (n : integer)
i ← n
while $i \geq 1$ do
    for $j ← i$ to $2i − 1$ do
        print “Hi”
    end for
    $i ← i/2$
end while
```

Problem 2. (15 points [recurrence relations])

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

\[
T(n) = \begin{cases} 
1 & n = 1 \\
3T\left(\frac{n}{2}\right) + 2 & n > 1 
\end{cases}
\]
Problem 3. (15 points [recurrence relations])

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 4. (15 points [writing/solving recurrence relations])
Consider the following variant for the closest-pair algorithm we discussed in class.

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

Assume each basic operation on points (computing distance, comparing coordinates, \ldots) takes \(O(1)\) time. Define \(T(n)\) to be the worst-case running time of \(\text{Closest-Pair}(A)\) on any array \(A\) of \(n\) points. Write a recurrence relation for \(T(n)\) and give a tight bound on \(T(n)\).
Problem 5. (20 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 6. (20 points [divide & conquer])

Suppose you are given an array $A = \{a_1, a_2, \ldots, a_n\}$ of $n$ distinct integers. You are told that the sequence of values $a_1, a_2, \ldots, a_n$ is unimodal, that is for some index $p \in [1, n]$, the values in the array increase up to position $p$, and then decrease the remainder of the way until position $n$. Give an algorithm to find the position $p$ in $O(\log n)$ time. You can assume $n$ to be a power of 2.