2-4 @ indices, not elements
(1,5) (2,5) (3,5) (4,5) (3,4)

5 \{ n, n-1, \ldots, 1 \}\n\frac{n \cdot (n-1)}{2} \text{ inversions.}

6 \Theta(n+k) \text{ where } n \text{ is \# elements and } k \text{ \# inversions}

C\text{OUNT-INVERSIONS}(A, p, r)
1 \text{ if } p < r
2 \text{ then } q \leftarrow \lfloor (p + r)/2 \rfloor
3 \quad l \leftarrow \text{C\text{OUNT-INVERSIONS}(A, p, q)}
4 \quad r \leftarrow \text{C\text{OUNT-INVERSIONS}(A, q, r)}
5 \quad m \leftarrow \text{M\text{ERGE-C\text{OUNT}(A, p, q, r)}}
6 \quad \text{return } l + r + m
7 \text{ else}
8 \quad \text{return 0}

M\text{ERGE-C\text{OUNT}(A, p, q, r)}
1 \quad n_1 \leftarrow q - p + 1
2 \quad n_2 \leftarrow r - q
3 \quad \text{create arrays } L[1..n_1 + 1] \text{ and } R[1..n_2 + 1]
4 \quad \text{for } i \leftarrow 1 \text{ to } n_1
5 \quad \quad \text{do } L[i] \leftarrow A[p + i - 1]
6 \quad \quad \text{for } j \leftarrow 1 \text{ to } n_2
7 \quad \quad \quad \text{do } R[j] \leftarrow A[q + j]
8 \quad L[n_1 + 1] \leftarrow \infty
9 \quad R[n_2 + 1] \leftarrow \infty
10 \quad i \leftarrow 1
11 \quad j \leftarrow 1
12 \quad l \leftarrow 0 \text{ \(\triangle\) for counting inversions}
13 \quad \text{for } k \leftarrow p \text{ to } r
14 \quad \quad \text{do if } L[i] < R[j]
15 \quad \quad \quad \text{then } A[k] \leftarrow L[i]
16 \quad \quad \quad i \leftarrow i + 1
17 \quad \quad \quad \text{else } A[k] \leftarrow R[j]
18 \quad \quad \quad j \leftarrow j + 1
19 \quad \quad \quad l \leftarrow l + (n_1 - i) \text{ \(\triangle\) increment by \# of elements still in } L
3.3(a)

\[
\begin{align*}
1 & \quad n^{1/\lg n} \\
\lg (\lg^* n) & \\
\lg^* (\lg n) & \\
\lg^* n & \\
\lg^* n & \\
2 & \\
\ln \ln n & \\
\sqrt{\lg n} & \\
\ln n & \\
2 & \\
\sqrt[2]{\lg n} & \\
\lg n & \\
(\sqrt{2}) & \\
n & \\
2 & \\
\lg n & \\
\lg(n!) & \\
n & \\
n^2 & \\
n^3 & \\
n^{\lg \lg n} & \\
(\lg n)! & \\
(\lg n) & \\
(\lg n) & \\
(3/2)^n & \\
2^n & < n < 2^n < e^n < n! < (n!)! & < 2^{2^n} < 2^{2^{2n}}
\end{align*}
\]

3.3(b)

\[f(n) = \begin{cases} 
2^{2^n} & \text{if } n \text{ is even} \\
\emptyset & \text{if } n \text{ is odd}
\end{cases}\]
$3.6$

\[
\begin{align*}
&[n] \\
&[\log^*(n)] \\
&[\log n] \\
&[\log \log n] \\
&[\log_2 \log n] \\
&[\log_{\log n} n] \\
&[1 \times 7 + 2]
\end{align*}
\]
4-1.  b) $\Theta(n)$

c) $\Theta(n^2)$

d) $\Theta(\sqrt{n \lg n})$

e) $\Theta(\lg \lg n)$

4-4.  b) $\Theta(n \lg \lg n)$

c) $\Theta(n \lg n)$

d) $\Theta(n \lg n)$

e) $\Theta(n)$

f) $\Theta(n \lg n)$

j) $\Theta(n \lg \lg n)$
The if part: If for all \( i = 1, 2, \ldots, m - 1 \) and \( j = 1, 2, \ldots, n - 1 \), we have

\[
\]  

(1)

then \( A \) is Monge. That is, for all \( i, j, k, \) and \( l \) such that \( 1 \leq i < k \leq m \) and \( 1 \leq j < l \leq n \),

\[
\]  

(2)

Statement (2) is equivalent to (3), where \( a, b > 0 \):

\[
\]  

(3)

If \( a \) and \( b \) are zero, this is obviously true; it amounts to saying that \( 2A[i, j] = 2A[i, j] \). If \( a \) and \( b \) are 1, then this is true too, because it is equivalent to the given hypothesis (1).

Now let us assume temporarily that \( b = 1 \), and make the induction hypothesis that for all \( a' < a \), (3) holds with all occurrences of \( a \) replaced by \( a' \). In particular, when \( a' = a - 1 \), the statement holds. Then we can prove (5) by renaming \( i \) to \( i + a' \); we obtain (1), our given hypothesis. Now we can iterate down through all \( a' < a \) using the same idea, until we reach the base case, in which \( a' = 1 \), in which case the statement clearly holds.

Now let us make the induction hypothesis that for all \( b' < b \), (3) holds with all occurrences of \( b \) replaced with \( b' \). In particular, if \( b' = b - 1 \), then for any \( a \),

\[
\]

by the result of the preceding paragraph. Iterating down through all \( b' < b \) until we reach 1, proves the statement for every \( b \) and every \( a \), which is our goal.

The only if part

It is easy to show the only if part; the given hypothesis is a special case of the constraint as viewed in (3), with \( a \) and \( b \) as 1.

\[ b. \]

The violation of the constraint

\[
\]

for \( 1 \leq i < k \leq m \) and \( 1 \leq j < l \leq n \) is violated when \( i = 1, k = 2, j = 2, \) and \( l = 3 \), because \( A[1, 3] = 22, A[2, 2] = 6, A[2, 3] = 7, \) and \( A[1, 2] = 23 \), so

\[
\]

This can be fixed by changing \( A[1, 3] \) to 24. Then \( A[1, 3] + A[2, 2] = 30 \).

We must also make sure that this change doesn’t cause another upset to the Monge array, which can be ensured by making sure that \( A[1, 3] + A[2, 4] \) (32) is less than \( A[1, 4] + A[2, 3] \) (39), which is so.
We would like to prove that if \( f(i) \) is the index of the column containing the leftmost minimum element of row \( i \), then \( f(1) \leq f(2) \leq \ldots f(m) \) for any \( m \times n \) Monge array. Equivalently, \( f(i) \leq f(i+1) \) for all \( i \) between 1 and \( m \).

Assume \( 1 \leq i < k \leq m \) and \( 1 \leq j < l \leq n \), and \( f(i) = l \). Because of the last assumption, we know also that \( A[i, j] > A[i, l] \). The constraint on Monge arrays tells us that

\[
\]

In order for this equation to be satisfied, \( A[k, l] \) must be less than \( A[k, j] \), by at least the amount that \( A[i, j] \) is larger than \( A[i, l] \). More formally, suppose that \( A[i, j] = A[i, l] + d \), where \( d \) is positive.

\[
\Rightarrow d + A[k, l] \leq A[k, j]
\]

If \( A[k, l] \) were greater than or equal to \( A[k, j] \), the last equation would be contracted, no matter how small \( d \) is, as long as \( d \) is positive. Therefore \( A[k, l] < A[k, j] \), so \( A[k, j] \) could not be the minimum element in row \( k \), i.e. \( f(k) \) could not be \( j \), for any \( j \) less than \( l \).

To summarize, if \( k > i \), then \( f(i) \leq f(k) \). In particular, letting \( k = i + 1 \), we have \( f(i) \leq f(i + 1) \).

d.

Let \( f(0) = 1 \) and \( f(m + 1) = n \), for simplicity. For each odd row \( i \), consider the elements \( A[i, f(i - 1)] \) through \( A[i, f(i + 1)] \), and take the least one. We are guaranteed that the leftmost minimum element for this row lies in this range, because \( f(i - 1) \leq f(i) \leq f(i + 1) \). There are \( f(i + 1) - f(i - 1) + 1 \) many elements in that range, so we can represent the number of elements to consider with the following sum:

\[
\sum_{i=1}^{m} f(i + 1) - f(i - 1) + 1 \\
= \sum_{i=1}^{m} f(i + 1) - f(i - i) + \sum_{i=1}^{m} 1 \\
= f(m + 1) - f(0) + \sum_{i=1}^{m} 1 \\
= f(m + 1) + \lceil m/2 \rceil \\
= n + \lceil m/2 \rceil \\
= O(m + n)
\]

e.

We showed in part \((d)\) that the cost of computing the leftmost minimum element in each of the odd rows for an \( m \times n \) Monge array when the leftmost minimum elements of the even rows were known was \( O(m + n) \).
This is the final stage of computing the leftmost minimum element in each row of an $m \times n$ Monge array; there is an earlier stage in which these elements for the even rows are computed. This is done by taking only the even rows and creating a subarray $A'$, with $\lfloor m/2 \rfloor$ rows and $n$ columns. Hence the recurrence to describe the cost of this algorithm is this:

$$T(m, n) = T\left(\frac{m}{2}, n\right) + O(m + n)$$

To solve this recurrence, I use the telescoping method. Consider the following sequence of equations as a sum, with $O(m + n)$ replaced with $c(m + n)$, $c$ a constant.

$$T(m, n) = T\left(\frac{m}{2}, n\right) + O(m + n)$$
$$T\left(\frac{m}{2}, n\right) = T\left(\frac{m}{2^2}, n\right) + c\left(\frac{m}{2} + n\right)$$
$$T\left(\frac{m}{2^2}, n\right) = T\left(\frac{m}{2^3}, n\right) + c\left(\frac{m}{2^2} + n\right)$$
$$\vdots$$
$$T\left(\frac{m}{2^{k-1}}, n\right) = T\left(\frac{m}{2^k}, n\right) + c\left(\frac{m}{2^{k-1}} + n\right)$$

The first term on the right-hand side of the last equation can be rewritten as $T(1, n)$, which is $O(n)$; it is the cost of finding the leftmost minimum in a matrix with one row of size $n$. Adding the equations, we have

$$T(m, n) = O(n) + c \sum_{i=1}^{k} 2^{-i}m + n$$
$$= O(n) + cm \sum_{i=1}^{k} 2^{-i}m + n \log m$$

Since $\sum_{i=1}^{k} 2^{-i} = k < 1$, that expression multiplied by $m$ is less than $m$. So $cm \sum_{i=1}^{k} 2^{-i} = O(m)$

$$T(m, n) = O(n) + O(m) + O(n \log n)$$
$$= O(m) + O(n \log n).$$