Sorting Lower Bound
Comparison-Based Sorting

Many sorting algorithms are comparison based.
- They sort by making comparisons between pairs of objects
- Examples: bubble-sort, selection-sort, insertion-sort, heap-sort, merge-sort, quick-sort, ...

Let us therefore derive a lower bound on the running time of any algorithm that uses comparisons to sort n elements, $x_1, x_2, \ldots, x_n$. 

Is $x_i < x_j$?

- yes
- no
Counting Comparisons

Let us just count comparisons then.

Each possible run of the algorithm corresponds to a root-to-leaf path in a decision tree.
Decision Tree Height

- The height of the decision tree is a lower bound on the running time.
- Every input permutation must lead to a separate leaf output.
- If not, some input ...4...5... would have the same output ordering as ...5...4..., which would be wrong.
- Since there are $n! = 1 \cdot 2 \cdot ... \cdot n$ leaves, the height is at least $\log(n!)$.

$$\text{minimum height (time)} = \log(n!)$$
The Lower Bound

- Any comparison-based sorting algorithms takes at least \( \log(n!) \) time.
- Therefore, any such algorithm takes time at least \( \Omega(n \log n) \).

\[
\log (n!) \geq \log \left( \frac{n}{2} \right)^{n/2} = (n/2) \log (n/2).
\]

That is, any comparison-based sorting algorithm must run in \( \Omega(n \log n) \) time.