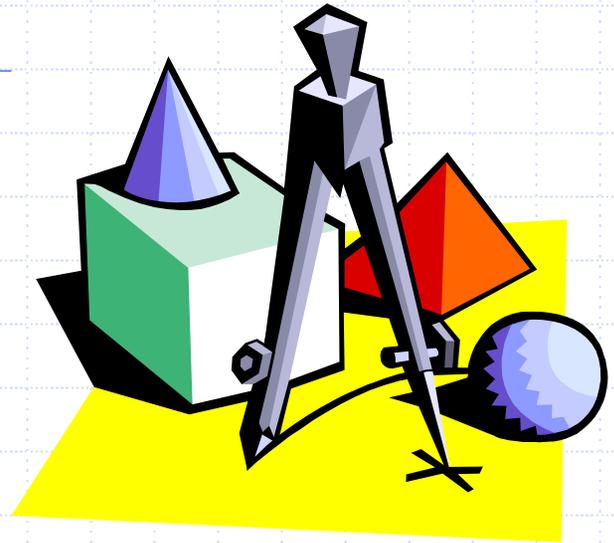# Sorting Lower Bound

Comparison-Based Sorting (§10.4)

◆ 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, $x_1, x_2, ..., x_n$.
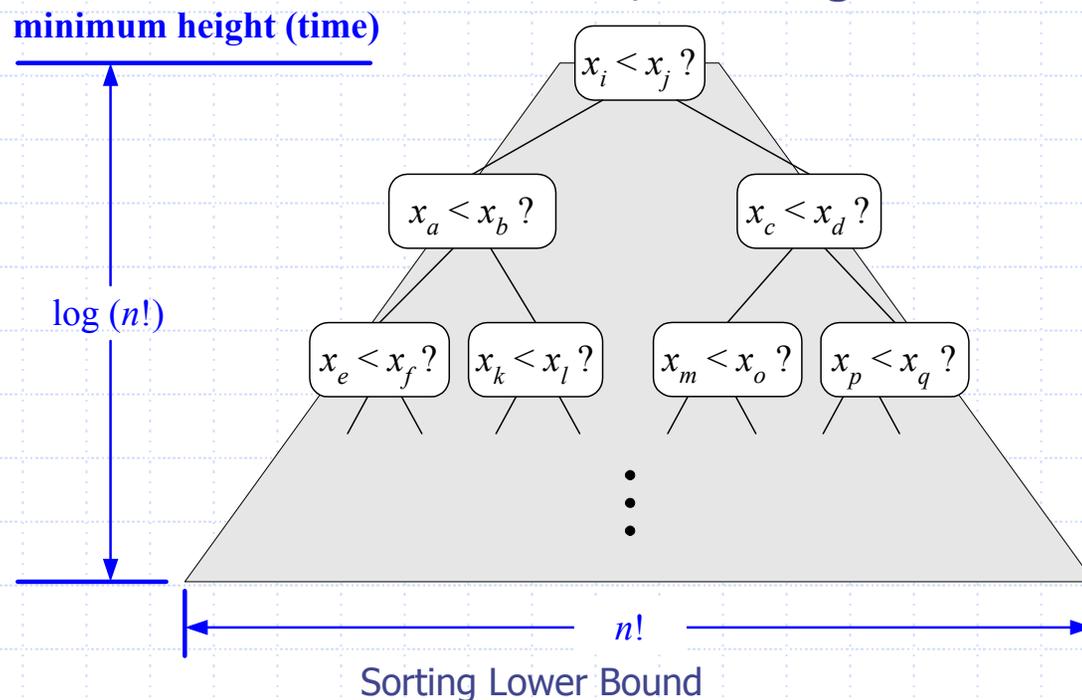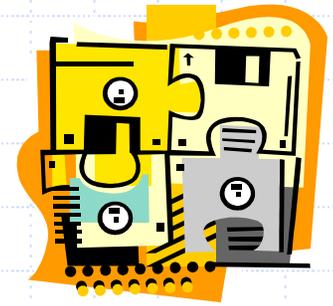
yes

Is $x_i < x_j$?

no
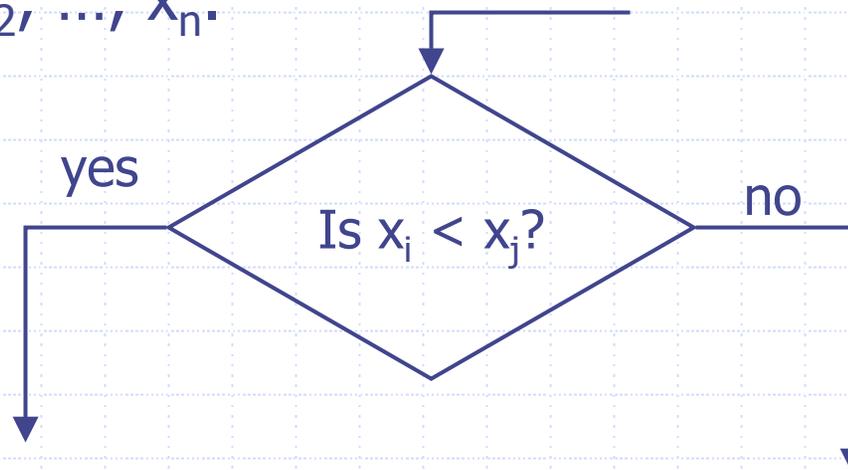
Counting Comparisons

◆ Let us just count comparisons then.

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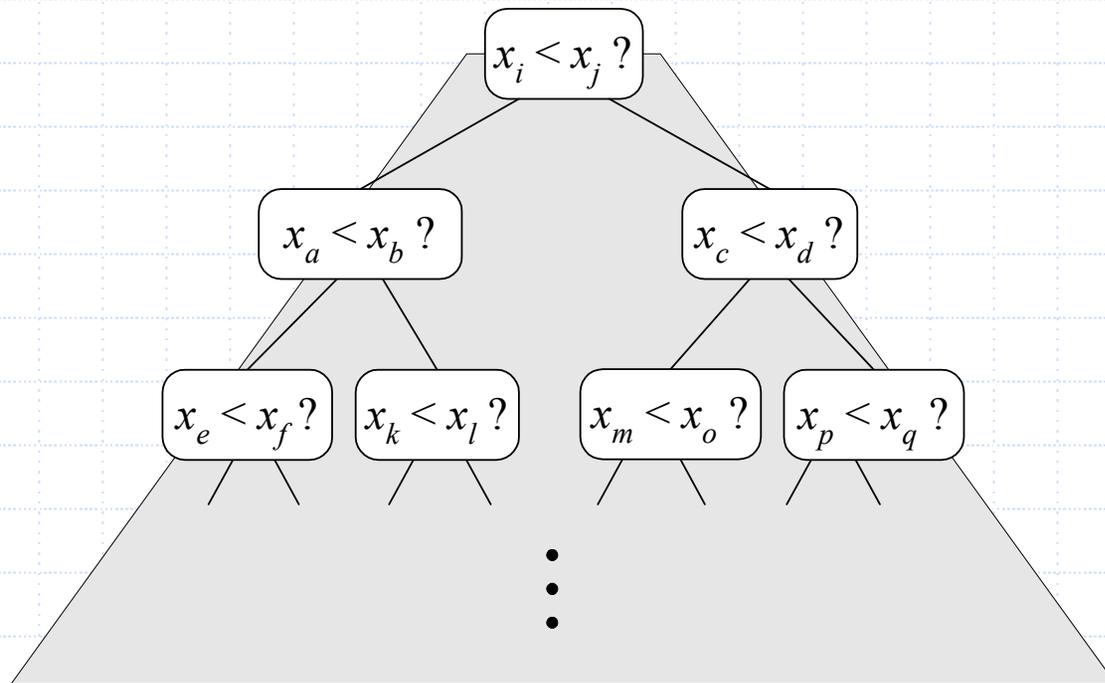
Decision Tree Height

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

minimum height (time)

$$x_i < x_j \;?$$

$$x_a < x_b \;?$$ $$x_c < x_d \;?$$

$\log (n!)$

$$x_e < x_f \,?$$ $$x_k < x_l \,?$$ $$x_m < x_o \;?$$ $$x_p < x_q \;?$$

$n!$

The Lower Bound

- Any comparison-based sorting algorithms takes at least log (n!) time

- Therefore, any such algorithm takes time at least

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

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