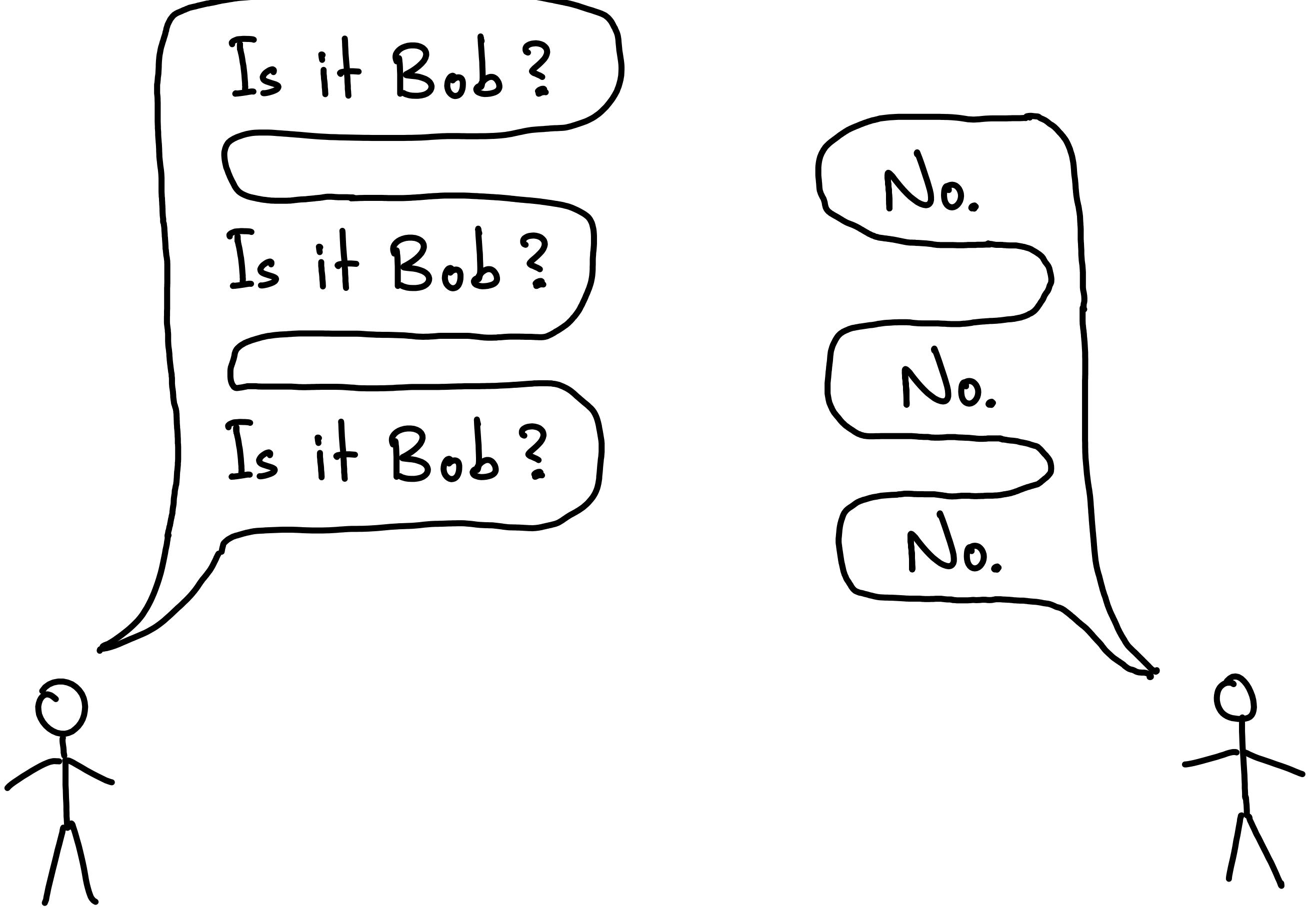
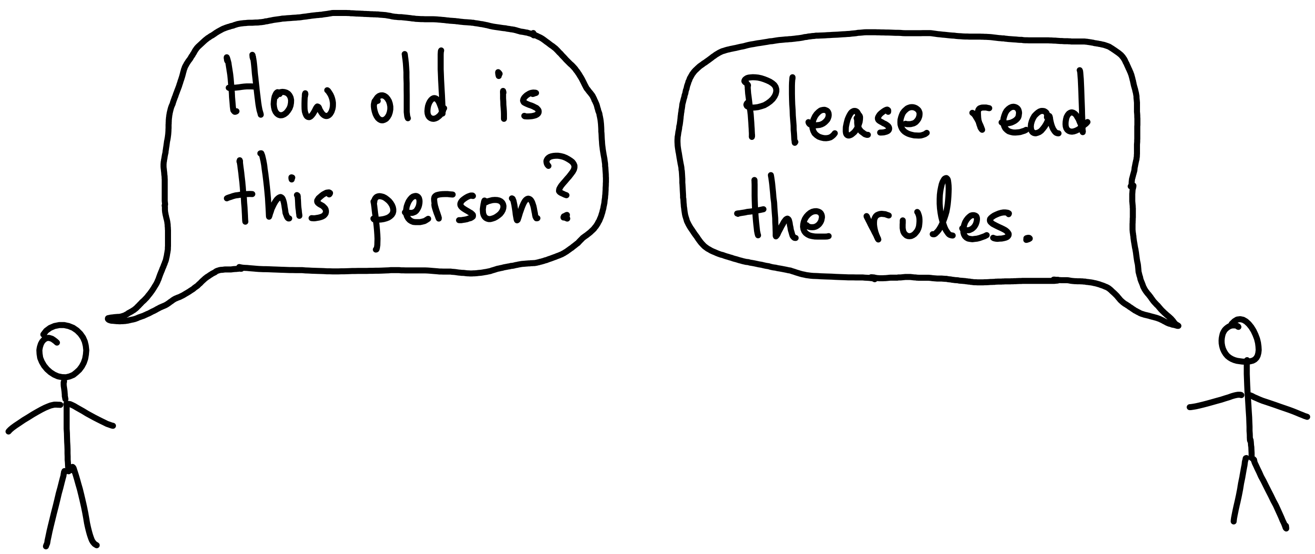
The game of 20 questions

Player 1 thinks of someone that player 2 could name.

Player 2 can ask 20 questions that have a YES or NO answer
& must determine who player 1 is thinking of.


Is it Bob?
Is it Bob?
Is it Bob?
No.
No.
No.


How old is this person?
Please read the rules.

- Suppose there are k possible answers (candidates)

- When player 2 asks a question & gets a reply,
  the remaining candidates are partitioned into 2 groups.

- In the worst case the answer is in the larger group.

  ↳ Player 2 should ask questions that split the set of candidates.

Suppose that player 2 is amazing.

↳ All questions split the set of candidates.

↳ Literally the best player EVER

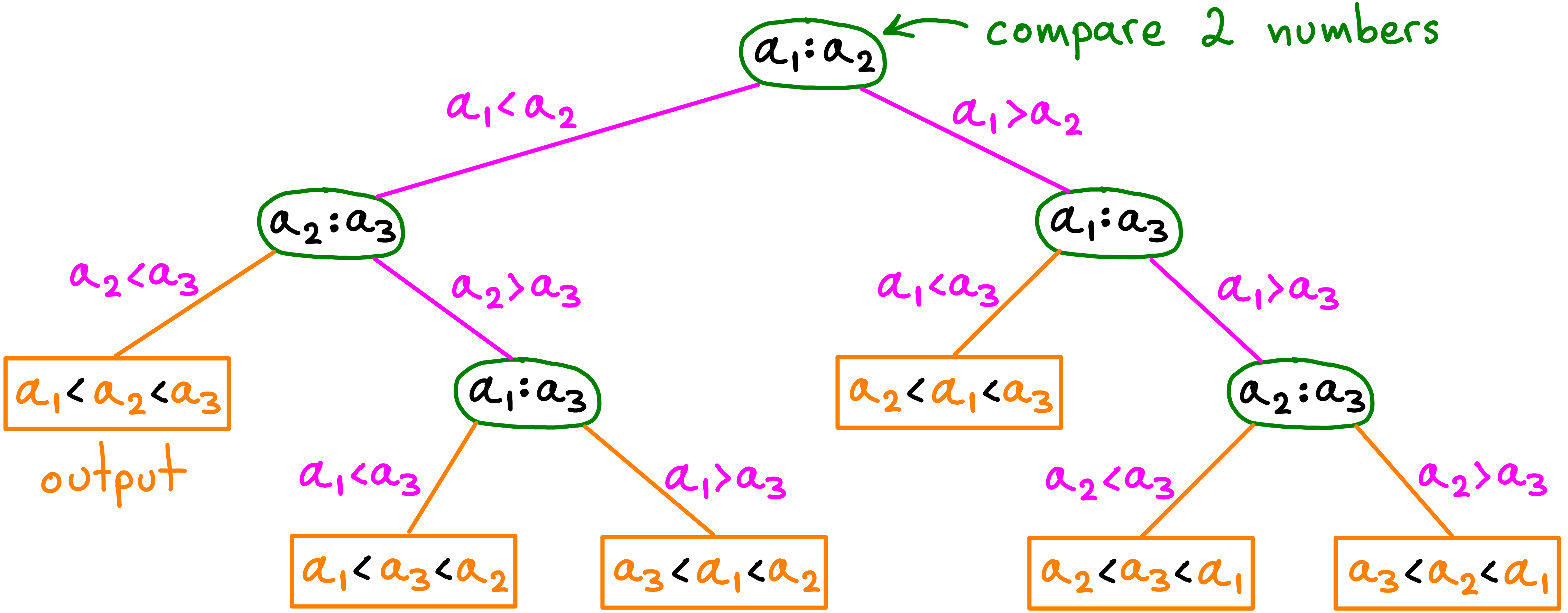
(assuming luck isn't involved)

For k candidates, how many questions does player 2 <u>need</u>?

$$\lceil \log_2 k \rceil$$

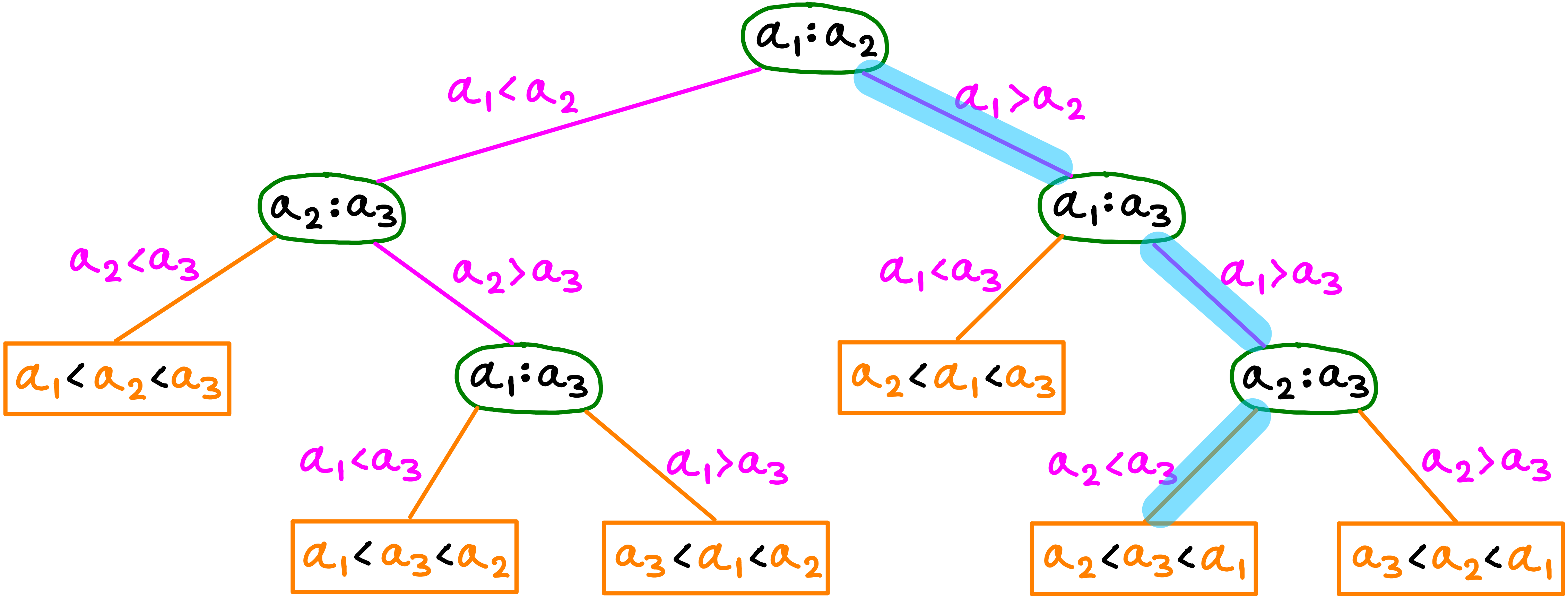
COMPARISON-BASED ALGORITHMS represented as DECISION TREES

example: an algorithm to sort 3 numbers: a_1, a_2, a_3 (no duplicates)



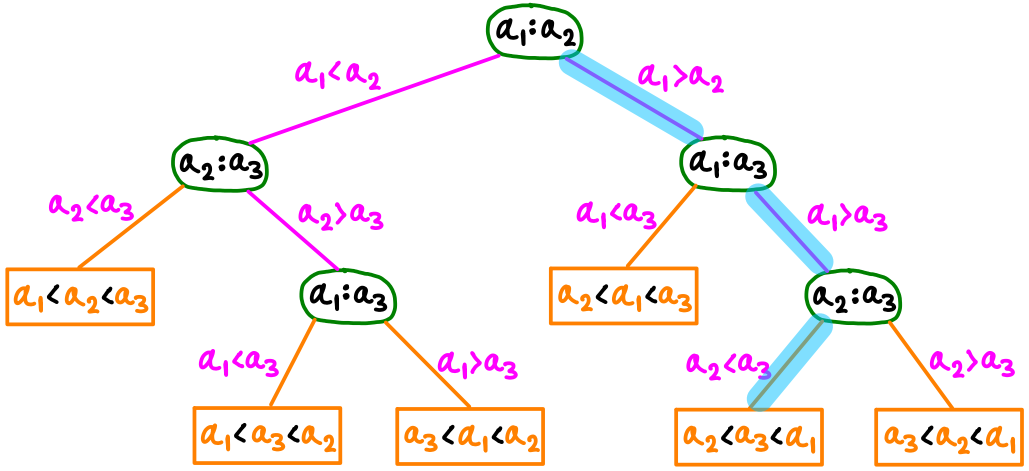
COMPARISON-BASED ALGORITHMS represented as DECISION TREES

example: an algorithm to sort 3 numbers: $\underset{a_1}{9}, \underset{a_2}{4}, \underset{a_3}{6}$



Every comparison-based algorithm has a corresponding decision tree

(not just sorting) Note: different #inputs → different tree!



- Every possible output must be represented by at least one leaf

- Every path from root to leaf represents the execution of the algorithm for specific input } path length ↕ time complexity

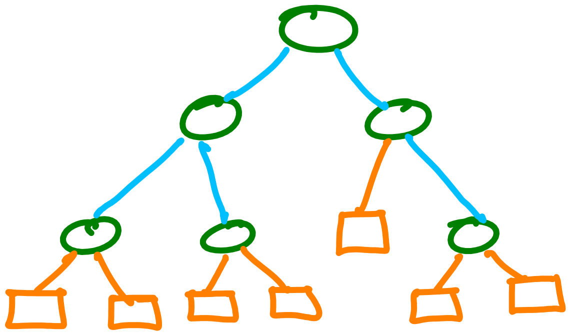
Longest path → worst-case time complexity

the SORTING LOWER BOUND

a lower bound for the worst-case time complexity of all comparison-based sorting algorithms

Consider the decision tree corresponding to any algorithm that can sort n items.

↳ e.g., the best algo EVER.



Every possible output must be represented by at least one leaf.

↳ Every permutation of the input must be represented

↳ # leaves $\geq n!$

To have $n!$ leaves, the tree needs a height of at least $\log_2(n!)$

Conclusion: For every comparison-sort algorithm,

worst-case time complexity $\geq \lceil \log_2 n! \rceil$

Approximating $\log_2 n!$

Stirling's formula: $n! \geqslant \left(\frac{n}{e}\right)^n$

(method 1)

$$\log_2 n! \geqslant \log_2\left(\frac{n}{e}\right)^n$$

$$= n\log_2\left(\frac{n}{e}\right)$$

$$= n\log n - n\log e$$

$$= n\log n - \Theta(n)$$

$$= \Omega(n\log n)$$

$\log(n!) = O(n \log n)$

$\log(n!) \leq \log(n^n) = n \log n$

(method 2, without using a formula)

$\log(n!) = \Omega(n \log n)$

$$\log(n!) = \log(n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdots \cdots 3 \cdot 2 \cdot 1)$$

$$= \log(\underbrace{n \cdot 1} \cdot \underbrace{(n-1) \cdot 2} \cdot \underbrace{(n-2) \cdot 3} \cdot \underbrace{(n-3) \cdot 4} \cdots \cdots \sim \underbrace{(n-\tfrac{n}{2}) \cdot (n-\tfrac{n}{2})})$$

$$\geq \log(n \cdot n \cdot n \cdot n \cdot \cdots n)$$

$$= \log(n^{n/2})$$ (assume n: even otherwise $\lfloor \frac{n}{2} \rfloor$)

$\Rightarrow \log(n!) \geq \frac{n}{2} \log n$

---

so $\frac{1}{2} n \log n \leq \log(n!) \leq n \log n$