Lesson Plan: Number Hunt

Year Group: 9 | **Duration:** 50 minutes | **Topic:** Binary Search vs Linear Search

1. Overview

Core Concept: Binary search — always guessing the midpoint of the remaining range to find a target in O(log n) time, compared to linear search O(n).

Learning Objectives:

- Demonstrate binary search by always guessing the midpoint
- Compare the efficiency of linear search vs binary search
- Interpret a graph showing O(n) vs O(log n) growth
- Explain why binary search requires a sorted list

Key Vocabulary:

Term	Definition
Linear search	Check items one by one from the start
Binary search	Always check the middle item; discard the half that can't contain the target
Midpoint	The middle value of a range
Sorted	Arranged in order (ascending or descending)
O(n)	Linear time — steps grow proportionally with input size
O(log n)	Logarithmic time — steps grow very slowly as input doubles

2. Before the Lesson

Print:

- ☐ `worksheet-results-and-graph.md` — 1 per student

No other materials needed.

3. Timed Lesson Flow

0–5 min — Hook: Guess My Number

1. *"I'm thinking of a number between 1 and 100. Guess it. I'll tell you higher or lower."*
2. Take 3–4 student guesses. How did they approach it?
3. Ask: *"If you started at 1 and counted up, how many guesses could you need?"* (up to 100)
4. *"Is there a smarter way?"*

5–10 min — Two Strategies

1. **Linear search** — start at 1, count up. Worst case: 100 guesses.
2. **Binary search** — always guess the middle of the remaining range. Worst case: 7 guesses.
3. Key question: *"Why does binary search need the list to be sorted?"*

10–20 min — Round 1: Linear Search (Class Demo)

Teacher thinks of a number. Class uses linear search (raise hand to guess: 1, 2, 3...). Record total guesses. This will take a while — that's the point!

20–30 min — Round 2: Binary Search (Pairs)

Pairs play 5 rounds. One thinks, one uses binary search (always guess the midpoint). Record guesses for each round.

30–40 min — Worksheet: Results and Graph

Complete the results table and plot on the pre-drawn axes.

40–47 min — Discussion

For 1,000,000 items: linear worst case = 1,000,000; binary worst case ≈ 20 ($\log_2 1{,}000{,}000 \approx 20$).

47–50 min — Real world: sorted databases, phonebooks, dictionaries, sorted arrays in code.

---

4. Teacher Facilitation Notes

What to look for:

- Students who don't always guess the exact midpoint — enforce: calculate `(low + high) ÷ 2`, round down
- Students who understand "higher/lower" but don't narrow the range correctly — demonstrate: if guess=50 and answer is HIGHER, the new range is 51–100, not 1–100

Common misconceptions:

- Binary search works on any list — NO, it requires a sorted list
- Binary search is always twice as fast — NO, it's logarithmically faster. For n=1,000,000, it's 50,000× faster in the worst case.

---

5. Extension Tasks

1. How many guesses does binary search need for 128 items? 256? 512? 1024? Find the pattern (powers of 2).
2. Write out the binary search algorithm as pseudocode.
3. Why can't you use binary search on an unsorted phone book? What would you need to do first?

---

6. Key Takeaway

> **Binary search halves the problem with each guess. For a million items, it finds the answer in ~20 guesses. Linear search takes up to a million.**