

3.
1

Largest Element in Array

EASY

■ What You Will Learn

- ✓ How to scan an array once to find the maximum
- ✓ Why one pass is enough
- ✓ Clean Java code with $O(n)$ time and $O(1)$ space

■ Intuition First

Imagine you're watching people walk by and you're trying to spot the tallest. You don't need to gather everyone and compare — you just remember the tallest you've seen so far, and update whenever you spot someone taller. That's it.

■ The Core Idea

- Start with $\text{max} = \text{arr}[0]$
- Loop through rest: if $\text{arr}[i] > \text{max}$, update max
- After one loop — max holds your answer

■ Visual — One Pass Scan

Input array — find the max

3	1	4	1	5	9	2	6
---	---	---	---	---	---	---	---

$\text{max} = 3 \rightarrow 3 \text{ stays} \rightarrow 4 > 3, \text{max}=4 \rightarrow 5 > 4, \text{max}=5 \rightarrow 9 > 5, \text{max}=9 \rightarrow \text{done!}$

max = 9 found at index 5 ✓

3	1	4	1	5	9	2	6
---	---	---	---	---	---	---	---

■ Dry Run

Step 1 max=arr[0]=3, start scan

3	1	4	1	5	9	2	6
---	---	---	---	---	---	---	---

Step 2 arr[2]=4 > 3, update max=4

3	1	4	1	5	9	2	6
---	---	---	---	---	---	---	---

Step 3

arr[4]=5 > 4, update max=5

3

1

4

1

5

9

2

6

Step 4

arr[5]=9 > 5, update max=9

3

1

4

1

5

9

2

6

Step 5

End of array — return 9 ✓

3

1

4

1

5

9

2

6

■ Java Code

```
public static int largestElement(int[] arr) {
    int max = arr[0]; // Start with first element
    for (int i = 1; i < arr.length; i++) {
        if (arr[i] > max) {
            max = arr[i]; // Found a bigger one
        }
    }
    return max;
}
```

■■ Complexity

Case	Time	Space	What happens?
Best	O(n)	O(1)	Must scan all — can't skip any element
Worst	O(n)	O(1)	Same always — no shortcuts exist

■■ Common Mistakes

- Don't initialize max = 0 — what if all elements are negative?
- Always use max = arr[0] as your starting point

■ Pro Tip

- Java has Arrays.stream(arr).max().getAsInt() — but know the manual way for interviews

■ Practice Problems

[Easy] Find max in $[-5, -1, -3]$ — all negatives

[Easy] Find index of max element, not the value

[Med] Find both max and min in a single pass

■ Quick Revision

- Scan once, track $\text{max} = \text{arr}[0]$
- Update max when $\text{arr}[i] > \text{max}$
- $O(n)$ time, $O(1)$ space — no extra array needed
- Never init $\text{max} = 0$ when negatives are possible