Understanding Array Indexing and Loop Ranges

Understanding Array Indexing and Loop Ranges in Java (and C++/Python)

Arrays are 0-indexed in most languages like Java, C++, Python.

If an array has 'n' elements, the valid indices are:

0, 1, 2, ..., n - 1

Loop Analysis:

Case 1: for (int i = 0; i < nums.length; i++)

- Covers all elements from index 0 to length - 1.
- Total iterations = length.
- This is the standard loop.

Case 2: for (int i = 0; i <= nums.length; i++)

- When i = nums.length, it's out of bounds.
- This causes a runtime error.

Case 3: for (int i = 1; i < nums.length; i++)

- Skips the first element (index 0).
- Valid, but only when you want to skip the first item.

Case 4: for (int i = 0; i < nums.length - 1; i++)

- Skips the last element (index n - 1).
- Valid, but only when you want to exclude the last item.

Understanding Array Indexing and Loop Ranges

Final Summary:

Loop Condition	What It Does	Correct?	Use Case
i = 0; i < length	Full array traversal	Yes	Standard full loop
i = 0; i <= length	One step extra -> Error	No	Causes index out of bounds
i = 1; i < length	Skips index 0	Yes	When you want to skip first item
i = 0; i < length - 1	Skips last index	Yes	When you want to exclude last

---

TL;DR:

i = 0; i < length is the correct way to loop over the full array safely.