Quick Sort

Selects a pivot, partitions the list around it, then recursively sorts the sublists on either side of the pivot, progressively moving towards the middle.

```python
def partition(arr: list, low: int, high: int) -> int:
    pivot = arr[high]
    lowest_index = low - 1
    for i in range(low, high):
        if arr[i] < pivot:
            lowest_index += 1
            arr[i], arr[lowest_index] = arr[lowest_index], arr[i]
    arr[lowest_index + 1], arr[high] = arr[high], arr[lowest_index + 1]

    return lowest_index + 1


def quick_sort(arr: list, low: int, high: int) -> None:
    if low < high:
        p = partition(arr, low, high)
        quick_sort(arr, low, p - 1)
        quick_sort(arr, p + 1, high)
```

Property	Details
How it works	Quick sort selects a pivot and rearranges the elements so that those smaller than the pivot are on the left and those larger are on the right. The pivot is then placed between the two groups. This creates a partition with two sublists: one with elements less than the pivot (left) and one with elements greater than the pivot (right). These sublists are then sorted recursively using the same process, with the sorting progressing from both sides towards the middle.
Time Complexity	O(n * log n) on average, O(n^2) in the worst case
Space Complexity	O(n) for recursive calls, O(log n) with optimised recursion
Stable	No
Type	in-place, recursive

- Follow on Bluesky
- Join Discord