**⚡ CHUYÊN SÂU VỀ QUICK SORT**

**✅ 1. Ý tưởng**

Chọn một phần tử gọi là **pivot**, sau đó chia mảng thành 2 phần:

* Phần bên trái: nhỏ hơn pivot
* Phần bên phải: lớn hơn pivot  
  → Đệ quy sắp xếp từng phần

**📦 Cách hoạt động:**

1. Chọn pivot (thường là phần tử cuối, đầu, hoặc ngẫu nhiên).
2. Phân chia mảng sao cho phần tử nhỏ hơn pivot nằm bên trái, lớn hơn nằm bên phải.
3. Đệ quy lặp lại trên 2 phần.

**🧠 2. Ví dụ minh họa:**

cpp

Sao chépChỉnh sửa

Mảng ban đầu: [8, 3, 5, 4, 7, 6, 1, 2]

Chọn pivot = 2

Sau phân tách: [1] 2 [8,3,5,4,7,6]

→ Tiếp tục đệ quy trái và phải.

**🔁 3. Code cơ bản của Quick Sort**

cpp

Sao chépChỉnh sửa

int partition(int a[], int l, int r) {

int pivot = a[r];

int i = l - 1;

for (int j = l; j < r; j++) {

if (a[j] <= pivot) {

i++;

swap(a[i], a[j]);

}

}

swap(a[i + 1], a[r]);

return i + 1;

}

void quickSort(int a[], int l, int r) {

if (l < r) {

int pi = partition(a, l, r);

quickSort(a, l, pi - 1);

quickSort(a, pi + 1, r);

}

}

**📊 4. Độ phức tạp**

| **Trường hợp** | **Độ phức tạp** |
| --- | --- |
| Best Case | O(n log n) ✅ |
| Average Case | O(n log n) ✅ |
| Worst Case | O(n²) ❌ |

**📌 Vì sao có O(n²)?**

* Nếu luôn chọn pivot **xấu** (phần tử lớn nhất/nhỏ nhất), mảng không chia đều → mất hiệu quả.

Ví dụ: [1,2,3,4,5] mà chọn pivot luôn là phần tử cuối → chia kiểu 1 vs 4 → đệ quy sâu như selection sort

**✅ 5. In-place và không ổn định**

| **Thuộc tính** | **Giá trị** |
| --- | --- |
| In-place | ✅ Có (chỉ dùng con trỏ, không cần mảng phụ) |
| Stable (ổn định)? | ❌ Không (có thể hoán vị lung tung) |

**⚖️ 6. Ưu – nhược điểm**

**✅ ƯU:**

* Thường chạy rất nhanh (O(n log n) trung bình).
* Không tốn bộ nhớ thêm (O(log n) do đệ quy).
* Cài đặt gọn, dễ cải tiến.

**❌ NHƯỢC:**

* Worst case là O(n²) nếu chọn pivot tệ.
* Không ổn định.
* Phân tích logic phân chia hơi khó hơn Merge Sort.

**🧠 7. Pivot chọn thế nào thì tốt?**

| **Cách chọn pivot** | **Ảnh hưởng** |
| --- | --- |
| Phần tử đầu/cuối | Dễ code nhưng dễ sai |
| **Ngẫu nhiên** | Tránh worst case |
| Trung vị (median of three) | Chia đều hơn, hiệu năng tốt hơn |

✅ std::sort trong C++ thường dùng quick sort cải tiến với chọn pivot trung vị.

**📦 8. Ứng dụng**

* Thư viện chuẩn của nhiều ngôn ngữ.
* Cần sắp xếp nhanh, không yêu cầu ổn định.
* Các bài toán: chọn phần tử thứ k, partition, đếm cặp nghịch thế,...

**💼 9. Mẹo khi đi phỏng vấn**

“Quick Sort có ổn định không?” → ❌ Không  
“Quick Sort có in-place không?” → ✅ Có  
“Tại sao Quick Sort nhanh?” → Vì chia đều – hoạt động cache tốt  
“Khi nào Quick Sort thành O(n²)?” → Khi pivot chia lệch liên tục

**🎯 10. So sánh Merge Sort vs Quick Sort**

| **Đặc điểm** | **Merge Sort** | **Quick Sort** |
| --- | --- | --- |
| Độ phức tạp | Luôn O(n log n) | Trung bình O(n log n) |
| In-place | ❌ Không | ✅ Có |
| Ổn định | ✅ Có | ❌ Không |
| Ứng dụng lớn | File, ổn định | Tốc độ trong RAM |
| Thường dùng | Khi cần ổn định | Khi cần hiệu năng cao |