**Chương 3 sắp xếp và tìm kiếm**

Câu 1: trong các thuật toán đã học thuật toán sếp thứ tự tối ưu nhất là: QuickSort

Kém tối ưu nhất là : SELECTION SORT

Vì : QuickSort : thuật toán chạy nhanh, độ phức tạp của thuật toán là O(log n)

SELECTION SORT: thuật toán này có độ phức tạm là 0(n), thuật toán tìm phần tử nhỏ nhất hóa vị nó…..các phần tử trong mảng đều phỉa so sánh với nhau, thực hiện bước lặp này cho đến khi đoạn danh sách đang xét còn một phần tử.

Câu 2:

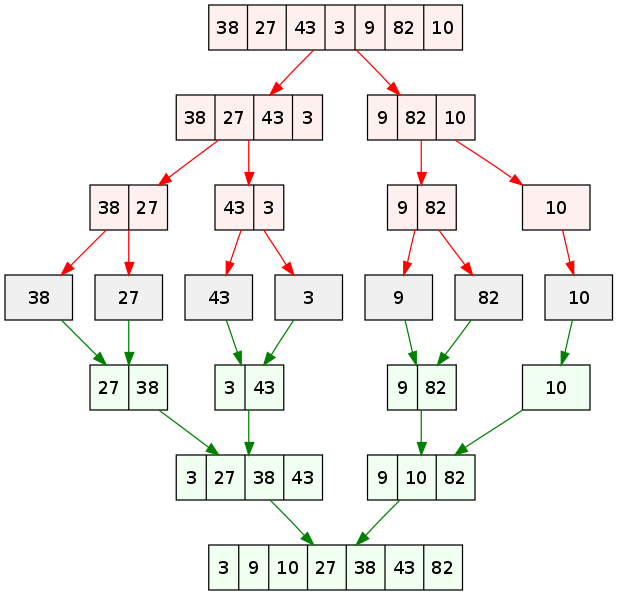
Hai phương pháp đã học là tìm kiếm nhị phân và tìm kiếm tuần tự trong trường hợp danh sách chưa có thứ tự chung sẻ nhau nhau vì nếu phương pháp tìm kiếm nhị phân chưa được sắp thếm thì phải sắp xếp ới dùng được.. do vậy thời gian tìm kiếm nhị phân và sắp xếp phân tử=>> như sắp xếp tuần tự

Câu 3: Thuật toán merge sort

## Ý tưởng của thuật toán merge sort

Giống như Quick sort, Merge sort là một thuật toán chia để trị. Thuật toán này chia mảng cần sắp xếp thành 2 nửa. Tiếp tục lặp lại việc này ở các nửa mảng đã chia. Sau cùng gộp các nửa đó thành mảng đã sắp xếp. Hàm merge() được sử dụng để gộp hai nửa mảng. Hàm merge(arr, l, m, r) là tiến trình quan trọng nhất sẽ gộp hai nửa mảng thành 1 mảng sắp xếp, các nửa mảng là arr[l…m] và arr[m+1…r] sau khi gộp sẽ thành một mảng duy nhất đã sắp xếp.

Hình ảnh dưới đây từ wikipedia sẽ hiển thị cho bạn toàn bộ sơ đồ tiến trình của thuật toán merge sort cho mảng {38, 27, 43, 3, 9, 82, 10}. Nếu nhìn kỹ hơn vào sơ đồ này, chúng ta có thể thấy mảng ban đầu được lặp lại hành động chia cho tới khi kích thước các mảng sau chia là 1. Khi kích thước các mảng con là 1, tiến trình gộp sẽ bắt đầu thực hiện gộp lại các mảng này cho tới khi hoàn thành và chỉ còn một mảng đã sắp xếp.

[](https://nguyenvanhieu.vn/wp-content/uploads/2018/07/thuat-toan-sap-xep-merge-sort-minh-hoa-code-su-dung-c.png)