**Trả lời câu hỏi chương 3:**

**Câu 1: Trong các phương pháp xếp thứ tự đã học, phương pháp nào tối ưu nhất, và kém tối ưu nhất? Tại sao?**

* Phương pháp tối ưu nhất là Quick Sort, có độ phức tạp thuật toán là O(n log n) và thời gian thực thi nhanh đối với đại đa số trường hợp (trường hợp xấu nhất rất hiếm khi xảy ra trong khi trường hợp tổng quát thường gặp trong các ứng dụng thực tế)
* Phương pháp kém tối ưu nhất đó là Bubble Sort bởi vì độ phức tạp thuật toán là O(n­­­­­2)

**Câu 2: Trong 2 phương pháp tìm kiếm đã học, trường hợp nào thì cả 02 phương pháp đều như nhau? Giải thích tại sao?**

* Khi dãy tìm đã được sắp xếp.
* VD: dãy có 6 phần tử, phần tử cần tìm nằm ở vị trí 1 trong dãy, khi đó thuật toán tìm kiếm tuần tự sẽ tốn 2 lần so sánh phân hoạch để tìm được phần tử đó – bằng với số lần so sánh của thuật toán tìm kiếm tuần tự.

**Câu 3: Ngoài các phương pháp xếp thứ tự đã học, hãy tìm hiểu thêm một phương pháp xếp thứ tự khác, giới thiệu sơ và giải thích.**

* Radix Sort (Sắp xếp theo cơ số nhị phân)
* Ý tưởng:
  + Giả sử chúng ta cần sắp xếp danh sách a[1..n], trong đó các phần tử là các số tự nhiên biểu diễn bằng 31 bit. Các bit được đánh số thứ tự từ phải sang trái bằng các số 0..30. Để đọc được bit thứ k của số tự nhiên x cần có một hàm GetBit(x,k).
  + Để sắp xếp (phân chia) danh sách theo bit thứ k của danh sách, ta tìm phần tử đầu tiên bên trái của danh sách có bit thứ k là 1 và phần tử đầu tiên bên phải có bit thứ k là 0 rồi đổi chỗ (Swap) cho nhau. Quá trình tiếp tục cho đến khi con trỏ hai đầu gặp nhau. Khi đó vị trí của hai con trỏ gặp nhau chia danh sách thành hai danh sách con: đanh sách đầu chứa các phần tử có bit thứ k bằng 0, danh sách sau chứa các phần tử có bit thứ k bằng 1.
  + Bằng lời giải đệ quy, tiếp tục phân chia các danh sách con này theo bit thứ k-1.. cho đến bit thứ 0.