**Trả lời câu hỏi chương 3.**

Câu 1:

-Phương Pháp tối ưu nhất: Quick Sort.

+Vì Quick Sort dựa vào một phần tử chốt và so sánh nó với các phần tử còn lại sau đó đổi chổ các thành phần đó cho phù hợp, việc này sẽ chia mảng thành từng đoạn nhỏ và không cần phải duyệt hết tất cả các phần tử trong các lần lặp sau.

-Phương Pháp kém tối ưu nhất: Selection Sort.

+Vì ta phải lặp lại việc xét từng phần tử để chọn ra phần tử bé nhất đưa lên đầu danh sách. Việc này lặp lại cho đến khi sắp xếp kết thúc nên thời gian thực hiện sẽ dài hơn.

Câu 2:

Khi ta thực hiện tìm kiếm trên 1 danh sách chưa có thứ tự, do đó khi dung Search Binary thì ta phải sắp xếp lại danh sách trước nên thời gian thực hiện của thuật toán sẽ kéo dài ra tương đương với Search Sequence.

Câu 3:

Phương pháp ShakerSort:

Giải thuật sắp xếp ShakerSort cũng dựa trên nguyên tắc đổi chỗ trực tiếp, nhưng tìm cách khắc phục các nhược điểm của BubleSort với những ý tưởng cải tiến chính như sau :

Trong mỗi lần sắp xếp, duyệt mảng theo 2 lượt từ 2 phiá khác nhau :

+ Lượt đi: đẩy phần tử nhỏ về đầu mảng

+ Lượt về: đẩy phần tử lớn về cuối mảng

Ghi nhận lại những đoạn đã sắp xếp nhằm tiết kiệm các phép so sánh thừa.

Các bước tiến hành như sau :

Ý tưởng chính của giải thuật là xuất phát từ cuối (đầu) dãy, đổi chỗ các cặp phần tử kế cận để đưa phần tử nhỏ (lớn) hơn trong cặp phần tử đó về vị trí đúng đầu (cuối) dãy hiện hành, sau đó sẽ không xét đến nó ở bước tiếp theo, do vậy ở lần xử lý thứ i sẽ có vị trí đầu dãy là i . Lặp lại xử lý trên cho đến khi không còn cặp phần tử nào để xét. Các bước tiến hành như sau :

Bước 1 :

l = 1; r = n; //từ l đến r là đoạn cần được sắp xếp

k = n; // ghi nhận vị trí k xảy ra hoán vị sau cùng

// để làm cơ sở thu hẹp đoạn l đến r

Bước 2a :

j = r ; // đẩy phần tử nhỏ về đầu mảng

Trong khi (j > l) :

Nếu a[j]<a[j-1]: swap(a[j] ,a[j-1])

k = j;//lưu lại nơi xảy ra hoán vị

j = j - 1;

l = k; //loại các phần tử đã có thứ tự ở đầu dãy

Bước 2b :

j = l ; // đẩy phần tử lớn về cuối mảng

Trong khi (j < r) :

Nếu a[j]>a[j+1]: swap(a[j] , a[j+1])

k = j;//lưu lại nơi xảy ra hoán vị

j = j+1;

r = k; //loại các phần tử đã có thứ tự ở cuối dãy

Bước 3 : Nếu l < r: Lặp lại Bước 2.