**TRẢ LỜI CÂU HỎI CHƯƠNG 3**

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

* Trong các phương pháp đã học phương pháp sắp xếp Quick Sort là tối ưu nhất. Vì có độ phức tạp trung bình O(n log( n)).
* Phương pháp sắp xếp Selection Sort là phương pháp kém tối ưu nhất. Vì Giải thuật này không phù hợp với tập dữ liệu lớn khi mà độ phức tạp trường hợp xấu nhất và trường hợp trung bình là O(n^2).

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

* Trong hai phương pháp tìm kiếm đã học trường hợp cả hai phương pháp đều như nhau trong trường hợp: có 2 phần tử (số lần so sánh bằng nhau).
* Giải thích: giả sử cho 2 phần tử đã sắp xếp: 90, 50. Sử dụng 2 phương pháp tìm kiếm và đếm số phép so sánh của 2 phương pháp. X = 50.

+ Phương pháp tìm kiếm tuần tự:

|  |  |
| --- | --- |
| 90 | 50 |

a [0] a [1]

so sánh giá trị x với a [i] (i = 0):

a[0] = 90 != X, i++;

so sánh giá trị x với a [i] (i = 1):

a[1] = 50 = X, tìm thấy X tại vị trí i = 3, tìm kiếm kết thúc.

* Vậy số phép so sánh của phương pháp này là 2 phép.

+ Phương pháp tìm kiếm nhị phân:

|  |  |
| --- | --- |
| 90 | 50 |

a [0] a [1]

Với left = 0, right = 1;

So sánh phần tử X tại vị trí a[(left + right)/2]

a [(0+1)/2]=a[0]

a[0] = 90 > X.

Giới hạn so sánh phần tử X từ vị trí a[(left + right)/2 +1] đến right = 1

a [(0+1)/2 +1]= a[1]

a[1]=50 = X, tìm thấy x.

* Số phép so sánh của phương pháp này là 2 phép

Vậy trong trường hợp tìm kiếm trong danh sách có 2 phần tử thì cả hai phương pháp sẽ như nhau do có số lần so sánh giống nhau.

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

Shell Sort là một giải thuật sắp xếp mang lại hiệu quả cao dựa trên giải thuật **sắp xếp chèn (Insertion Sort)**. Giải thuật này tránh các trường hợp phải tráo đổi vị trí của hai phần tử xa nhau trong giải thuật sắp xếp chọn (nếu như phần tử nhỏ hơn ở vị trí bên phải khá xa so với phần tử lớn hơn bên trái).

Đầu tiên, giải thuật này sử dụng giải thuật sắp xếp chọn trên các phần tử có khoảng cách xa nhau, sau đó sắp xếp các phần tử có khoảng cách hẹp hơn. Khoảng cách này còn được gọi là **khoảng (interval)** – là số vị trí từ phần tử này tới phần tử khác. Khoảng này được tính dựa vào công thức Knuth như sau:

h = h\* 3 + 1

h: là khoảng (interval) với giá trị ban đầu là 1.

Giải thuật này khá hiệu quả với các tập dữ liệu có kích cỡ trung bình khi mà độ phức tạp trường hợp xấu nhất và trường hợp trung bình là O(n), với n là số phần tử.