**Chương I: Tổng quan**  
**1. Nêu các thành phần cơ bản của một chương trình và cho biết mối  
quan hệ của các thành phần đó:**

* Phần Khai báo và phần thân chương trình, trong đó phần thân bắt buộc phải có.

**2. Thành phần cấu trúc dữ liệu có tác động đến những yếu tố nào của  
chương trình**

- **cấu trúc dữ liệu** là một cách lưu [dữ liệu](https://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u) trong [máy tính](https://vi.wikipedia.org/wiki/M%C3%A1y_t%C3%ADnh) sao cho nó có thể được sử dụng một cách hiệu quả.

- Trong thiết kế nhiều loại chương trình, việc chọn cấu trúc dữ liệu là vấn đề quan trọng. Kinh nghiệm trong việc xây dựng các hệ thống lớn cho thấy khó khăn của việc triển khai chương trình, chất lượng và hiệu năng của kết quả cuối cùng phụ thuộc rất nhiều vào việc chọn cấu trúc dữ liệu tốt nhất.  
**3. Thành phần giải thuật có tác động đến yếu tố nào của chương trình**

- Trong [toán học](https://vi.wikipedia.org/wiki/To%C3%A1n_h%E1%BB%8Dc) và [khoa học máy tính](https://vi.wikipedia.org/wiki/Khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh), một **thuật toán**, còn gọi là **giải thuật**, là một [tập hợp hữu hạn](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_h%E1%BB%A3p#L%E1%BB%B1c_l%C6%B0%E1%BB%A3ng_c%E1%BB%A7a_t%E1%BA%ADp_h%E1%BB%A3p_-_H%E1%BB%AFu_h%E1%BA%A1n_v%C3%A0_v%C3%B4_h%E1%BA%A1n) các hướng dẫn [được xác định rõ ràng](https://vi.wikipedia.org/w/index.php?title=%C4%90%C6%B0%E1%BB%A3c_x%C3%A1c_%C4%91%E1%BB%8Bnh_r%C3%B5&action=edit&redlink=1), có thể thực hiện được bằng máy tính, thường để giải quyết một lớp vấn đề hoặc để thực hiện một phép tính. Các thuật toán luôn [rõ ràng](https://vi.wikipedia.org/w/index.php?title=T%E1%BB%91i_ngh%C4%A9a&action=edit&redlink=1) và được sử dụng chỉ rõ việc thực hiện các [phép tính](https://vi.wikipedia.org/w/index.php?title=Ph%C3%A9p_t%C3%ADnh&action=edit&redlink=1), [xử lý dữ liệu](https://vi.wikipedia.org/wiki/X%E1%BB%AD_l%C3%AD_d%E1%BB%AF_li%E1%BB%87u), [suy luận tự động](https://vi.wikipedia.org/w/index.php?title=Suy_lu%E1%BA%ADn_t%E1%BB%B1_%C4%91%E1%BB%99ng&action=edit&redlink=1) và các tác vụ khác.  
**4. Theo bạn khi xây dựng chương trình, thành phần nào của chương  
trình nên được thiết kế trước? vì sao**

**- Xây dựng cấu trúc trước**

- Nên xây dựng thuật toán đầu tiên khi thiết kế chương trình vì Xây dựng mô hình chặt chẽ, chính xác hơn và chi tiết hóa hơn phương pháp đã lựa chọn. Xác định rõ ràng dữ liệu vào, ra cho các bước thực hiện cơ bản và trật tự thực hiện các bước cơ bản đó. Nên áp dụng phương pháp thiết kế có cấu trúc, từ thiết kế tổng thể tiến hành làm mịn dần từng bước.  
**5. Độ phức tạp có thể cài đặt chương trình trên máy tính và độ phức  
tạp không thể cài đặt chương trình mà nên thiết kế lại.**

- Độ phức tập có thể cài đặt chương trình trên máy tính: O(1), O(n),

- Độ phức tập không thể cài đặt chương trình trên máy tính mà nên thiếu kế lại : O(2n),O(log n),n2

**Chương II: Đệ quy**  
**1. Đệ quy là gì, khi nào giải thuật nên được thiết kế theo tiếp cận đệ  
quy**

* Đệ quy là phương pháp dùng trong các chương trình máy tính trong đó có một hàm tự gọi chính nó
* Nếu lời giải của một bài toán X*X*, được thực hiện bằng lời giải của một bài toán X'*X*′ có dạng giống như X*X*, thì đó là một lời giải đệ quy. **Giải thuật tương ứng với lời giải như vậy gọi là giải thuật đệ quy.**  
  **2. Trình bày các phương pháp đệ quy và độ phức tạp tương ứng**

+ Đệ quy tuyến tính. Độ phức tạp:

+ Đệ quy nhị phân. Độ phức tạp:

+ Đệ quy phi tuyến .Độ phức tạp:

+ Đệ quy tương hỗ. Độ phức tạp:  
**3. Đệ quy tuyến tính là gì, khi nào ta sử dụng đệ quy tuyến tính**

* Đệ quy tuyến tính là hàm đệ quy chỉ gọi chính nó một lần trong thân hàm. Hiểu đơn giản là trong một hàm, nếu có duy nhất một câu lệnh gọi chính hàm đó thì được gọi là hàm đệ quy tuyến tính.
* Ví dụ hàm tính giai thừa chính là một hàm đệ quy tuyến tính, vì nó chỉ gọi lại chính nó một lần duy nhất.  
  **4. Đệ quy nhị phân là gì, khi nào ta sử dụng đệ quy nhị phân**
* Đệ quy nhị phân là dạng đệ quy gọi hai lần chính nó. Hiểu đơn giản là trong một hàm đệ quy, mà có dòng lệnh gọi chính hàm đó hai lần.
* Ta có một hàm tìm dãy số Fibonacci sử dụng đệ quy.  
  **5. Khi nào ta cần khử đệ quy cho chương trình**
* Cứ mỗi lần gọi đệ quy máy phải dành một số vùng nhớ để lưu trữ các trị, các thông số và biến cục bộ. Do đó, đệ quy tốn nhiều vùng nhớ, thời gian truyền đối mục, thiết lập vùng nhớ trung gian và trả về kết quả Vậy để khắc phục hạn chế đó người ta nói đến khử đệ quy. Khử đệ quy ở đây có nghĩa là biến một thủ tục đệ quy thành một thủ tục chỉ chứa vòng lặp mà không ảnh hưởng gì đến các yếu tố khác, chứ không phải là thay đổi thuật toán.

**Chương III: Tìm kiếm**  
**1. Tìm kiếm là gì, có nhưng phương pháp tìm kiếm nào và độ phức tạp  
của mỗi phương pháp**

* Tìm kiếm là việc thu thập một số thông tin nào đó từ một khối thông tin lớn đã được lưu trước đó
* Tìm kiếm tuyến tính.Độ phức tạp O(n)
* Tìm kiếm Nhị phân.Độ phức tập O(log(n))

**2. Tìm kiếm tuyến tính (TKTT) là gì, khi nào ta sử dụng phương pháp  
TKTT. Phân tích ưu và nhược điểm của TKTT**

- Tìm kiếm tuyến tính : là một giải thuật tìm kiếm rất cơ bản. Trong kiểu tìm kiếm này, một hoạt động tìm kiếm liên tiếp được diễn ra qua tất cả từng phần tử.

- Nếu danh sách được liên kết là cấu trúc dữ liệu bất kể các phần tử được sắp xếp như thế nào, tìm kiếm tuyến tính được chấp nhận do không có khả năng thực hiện trực tiếp thuật toán tìm kiếm nhị phân.

Ưu điểm:

- Giải thuật này tỏ ra khá hiệu quả **khi cần tìm kiếm trên một danh sách đủ nhỏ hoặc một danh sách chưa sắp thứ tự đơn giản**.

- Tìm kiếm tuyến tính rất dễ sử dụng và không cần bất kỳ yếu tố nào

Nhược điểm:

- tìm kiếm tuyến tính không yêu cầu các yếu tố được sắp xếp

**3. Tìm kiếm nhị phân (TKNP) là gì, khi nào ta sử dụng phương pháp  
TKNP. Phân tích ưu và nhược điểm của TKNP**

* Tìm kiếm nhị phân: là một giải thuật tìm kiếm nhanh với độ phức tạp thời gian chạy là Ο(log n). Giải thuật tìm kiếm nhị phân làm việc dựa trên nguyên tắc chia để trị .
* Khi một mảng là cấu trúc dữ liệu và các phần tử được sắp xếp theo thứ tự được sắp xếp, thì tìm kiếm nhị phân được ưu tiên để tìm kiếm **nhanh**

Ưu điểm:

-Tìm kiếm nhị phân yêu cầu mảng được sắp xếp là lý do Nó yêu cầu xử lý để chèn vào vị trí thích hợp của nó để duy trì danh sách được sắp xếp

Nhược điểm:

- thuật toán tìm kiếm nhị phân tuy nhiên rất khó và các yếu tố nhất thiết phải được sắp xếp theo thứ tự.

**Chương IV: Sắp xếp**  
**1. Sắp xếp là gì, có những phương pháp sắp xếp cơ bản nào và độ phức tạp  
tương ứng của mỗi phương pháp**

* Sắp xếp là quá trình bố trí lại các phần tử của một tập hợp theo thứ tự nào đó.
* Sắp xếp kiểu lựa chọn ( Selection Sort). Độ phức tạp O (n2)
* Sắp xếp theo kiểu thêm dần (Insertion Sort).Độ phức tạp O(n2)
* Sắp xếp Bong Bóng (Bubble Sort ). Độ phức tạp O (n2)
* Sắp xếp Nhanh (Quick Sort). Độ phức tạp trung bình O(nlog2n)
* Sắp xếp vun đống (heap Sort). Độ phức tạp trung bình O(nlog2n)2

**2. Phân tích sự giống và khác nhau giữa hai thuật toán selection sort và  
bubble sort**

- Giống nhau: Sắp xếp dựa trên so sánh,Hoán đổi vị trí, Độ phức tạp trung bình O(n2)

- Khác nhau:

+ Bubble sort : so sánh các phần tử để đặt các phần tử lớn nhất vào vị trí cuối cùng.

+ Selection sort: so sánh các phần tử để đặt các phần tử nhỏ nhất vào vị trí phía trước.

**3. Nêu cấu trúc dữ liệu được sử dụng cho thuật toán insert sort và đánh giá  
độ phức tạp của thuật toán trong ba trường hợp phân bố của dữ liệu**

Thuật toán sắp xếp chèn (**Insertion Sort**) là thuật toán sắp xếp hoạt động tương tự cách chúng ta sắp xếp các quân bài trên tay trong một trò chơi bài.

Để sắp xếp theo đúng trật tự, người chơi sẽ rút lần lượt từ quân bài thứ 2, sau đó so với các quân bài đứng trước nó để chèn vào vị trí thích hợp.

Tóm lại, sắp xếp chèn là thuật toán sắp xếp đặt một phần tử chưa được sắp xếp vào vị trí thích hợp của nó trong mỗi lần lặp.

* **Trường hợp xấu nhất:** O(n2)  
  Giả sử, một mảng có thứ tự tăng dần và bạn muốn sắp xếp nó theo thứ tự giảm dần. Trong trường hợp này, trường hợp xấu nhất sẽ xảy ra.  
  Mỗi phần tử phải được so sánh với mỗi phần tử khác, do đó, đối với mỗi phần tử thứ n sẽ có (n-1) số phép so sánh được thực hiện. Do đó, tổng số phép so sánh = n\*(n-1) ~ n2
* **Trường hợp tốt nhất:** O(n)  
  Khi mảng đã được sắp xếp, vòng lặp bên ngoài chạy n lần trong khi vòng lặp bên trong hoàn toàn không chạy. Vì vậy, chỉ có n số phép so sánh. Do đó, độ phức tạp là tuyến tính.
* **Trường hợp trung bình:** O(n2)  
  Nó xảy ra khi các phần tử của mảng có thứ tự lộn xộn (không tăng dần cũng không giảm dần).

**4. Nêu cấu trúc dữ liệu được sử dụng cho thuật toán Quick sort, phương pháp thiết kế của thuật toán?, phần tử chốt là gì và được dùng để làm gì.Có những phương pháp chọn phần tử chốt là gì. Phân tích độ phức tạp của thuật toán do bị ảnh hưởng bởi việc chọn phần tử chốt.**

- Thuật toán sắp xếp Quick – Sort được thực hiện theo mô hình chia để trị.Thuật toán được thực hiện xung quanh một phần tử gọi là chốt (key).Mỗi phần tử chốt ở vị trí khác nhau sẽ cho ta một phiên bản khác nhau của thuật toán.

- Phương pháp chọn chốt:

+ Luôn lựa chọn phần tử đầu tiên trong dãy làm chốt

+ Luôn lựa chọn phần tử cuối cùng trong dãy làm chốt

+ Luôn lựa chọn phần tử ở giữa dãy làm chốt

+ Lựa chọn phần tử ngẫu nhiên trong dãy làm chốt

- Mỗi lần phân hoạch ta chọn phải phần tử có giá trị cực đại hoặc cực tiểu làm mốc.Khi đó dãy bị phân hoạch thành 2 phần không đều: 1 phần chỉ có một phần tử,phần còn lại có n-1 phần tử.Do đó, ta cần tới n lần phân hoạch mới xếp xong. Độ phức tạp O(n2)

Trường hợp tốt nhất,trung bình. Độ phức tạp O(nlogn)

**5. Nêu cấu trúc dữ liệu được sử dụng cho thuật toán Heap sort, phương  
pháp thiết kế của thuật toán?, quá trình sắp xếp phụ thuộc vào chức  
năng nào của đống, độ phức tạp trung bình của thuật toán là bao nhiêu.**

**-** Thực hiện sắp xếp thông qua việc tạo các heap,trong đó heap là 1 cây nhị phân hoàn chỉnh có tính chất là khóa ở nút cha bao giờ cũng lớn hơn khóa ở các nút con.

Việc thực hiện giải thuật này được chia làm 2 giai đoạn

+ Đầu tiên là việc tạo heap từ dãy ban đầu.Theo định nghĩa của heap thì nút cha bao giờ cũng lớn hơn các nút con.Do vậy,nút gốc của heap bao giờ cũng là phần tử lớn nhất

+ Giai đoạn 2: Sắp xếp dựa trên heap tạo được. Phần tử bé nhất của dãy và được đặt lên đầu

* Một cây nhị phân hoàn toàn chỉnh có n nút thì chiều cao của cây đó là [log2(n+1)]. Khi tạo đống cũng như khi vun đống trong giai đoạn sắp xếp, trường hợp xấu nhất thì số lượng phép so sánh cũng chỉ tỷ lệ với chiều cao của cây
* Do đó có thể suy ra, trong trường hợp xấu nhất, cấp độ lớn của thời gian thực hiện Heap Sort là O(nlog2n).Việc đánh giá thời gian thực hiện trung bình giải thuật Heap Sort cũng là O(nlog2n)