Tóm tắt slide:

Việc viết một chương trình, giải một bài toán được kết hợp bởi 2 yếu tố, lựa chọn một cấu trúc dữ liệu phù hợp, sau đó tìm ra phương hướng kết hợp mọi thứ bằng giải thuật để có thể giải được bài toán.

Cấu trúc dữ liệu + Thuật toán = Chương trình

Giải thuật là gì? (xem slide)

Những điều mà 1 giải thuật cần chú ý là:

* Giải được bằng giải thuật
* Tối ưu hóa giải thuật
* Triển khai giải thuật

Đặc trưng của giải thuật bao gồm:

* Tính xác định : Giải thuật nên rõ ràng và không mơ hồ. Mỗi một giai đoạn (hay mỗi bước) nên rõ ràng và chỉ mang một mục đích nhất định.
* Tính dừng: Các giải thuật phải kết thúc sau một số hữu hạn các bước.
* Tính đúng đắn
* Tính phổ dụng: Một giải thuật có tính phổ biến nếu giải thuật này có thể giải quyết được một lớp các vấn đề tương tự.
* Tính khả thi

Có nhiều cách diễn đạt các giải thuật, ví dụ như:

* Dạng lưu đồ (Dùng các hình khối quy ước sẵn và biểu diễn)
* Dạng ngôn ngữ tự nhiên (Sử dụng ngôn ngữ của bản thân và diễn giải thuật toán theo cách mình hiểu)
* Dạng giả mã (là mô tả thuật toán một cách ngắn gọn - thường là loại bỏ các thứ không cần thiết kèm theo sử dụng một số quy ước của ngôn ngữ lập trình)

Cấu trúc dữ liệu (Data Structure) (Đ/n xem slide

Mối quan hệ giữa Giải thuật và Cấu trúc dữ liệu:

* Đối tượng xử lí của giải thuật chính là dữ liệu
* Với một cấu trúc dữ liệu, sẽ có những giải thuật tương ứng.
* Khi các trúc dữ liệu thay đổi thường giải thuật cũng phải thay đổi theo.

Chúng ta cần đưa ra một chiến lược giải quyết bài toán một cách hợp lí như:

* Divide and Conquer
* Dynamic Programming
* Back-tracking
* Greedy method

Hãy cùng xem xét chiến thuật Divide and Conquer:

* Thường có hai cách như sau:
* Top-Down Design

Nó được hiểu nôm na là ban đầu phân tích bài toán và chia nhỏ từng vấn đề của bài toán. Sau đó xử lí từng phần và kết hợp nó lại

* Tinh chỉnh từng bước:
* Biểu diễn ý tưởng bằng ngôn ngữ tự nhiên
* Cụ thể từng phần, thay đổi bằng ngôn ngữ chương trình
* Cuối cùng ta có chương trình

Tham khảo các ví dụ slide để hiểu thêm

Một thuật toán thường được xây dưng trên các mục tiêu sau:

* Thuật toán dễ hiểu, cài đặt và gỡ lỗi
* Thuật toán sử dụng hiệu quả các tài nguyên của máy, thiên về tốc độ xử lí và tiết kiệm bộ nhớ
* Điều đó được đánh giá trên các khía cạnh sau:
  + Dữ liệu đầu vào
  + Tốc độ xử lí của phần cứng
  + Trình biên dịch: Gồm có Interpreted (như Python, PHP, JavaScript,…) và Compiled language (C, C++, Haskell, Rust,…). Chính vì có 2 đặc điểm trên nên tôi nghĩ là nó đã sinh ra 2 trường phái trên.