



# TRƯỜNG ĐH CÔNG NGHỆ THÔNG TIN



## CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT



# Tài Liệu Tham Khảo

- Trần Hạnh Nhi, Dương Anh Đức. **Giáo trình Cấu Trúc Dữ Liệu 1**, ĐHQG Tp. HCM, 2000.
- Robert Sedgewick. **Cẩm nang thuật toán** (bản dịch của nhóm tác giả ĐH KHTN), NXB Khoa học kỹ thuật, 1994.
- P. S. Deshpande, O. G. Kakde. **C & Data Structures**, 2004.
- Dr. Dobb's. **Algorithms and Data Structures**, 1999
- A.V. Aho, J.E Hopcroft, J.D Ullman. **Data structures and Algorithms**, Addison Wesley, 1983.



# Nội Dung Chương Trình

- Buổi 1: Giới thiệu về CTDL & Giải Thuật.  
Các thuật toán tìm kiếm.
- Buổi 2: Interchange Sort, Selection Sort, Bubble Sort, Insertion Sort.
- Buổi 3: Shaker Sort, Shell Sort, Heap Sort.
- Buổi 4: Quick Sort, MergeSort, Radix Sort.
- Buổi 5: Cấu trúc động, Danh sách liên kết đơn.



# Nội Dung Chương Trình

- Buổi 6: Stack, Queue.
- Buổi 7: Danh sách liên kết kép.
- Buổi 8: Cây, Cây nhị phân, cây nhị phân tìm kiếm.
- Buổi 9: Cây cân bằng (AVL).
- Buổi 10: Các CTDL mở rộng.
- Buổi 11: Ôn tập.



# CHƯƠNG 1

## TỔNG QUAN VỀ CTDL VÀ THUẬT TOÁN



# Nội Dung

- Tổng quan về CTDL và thuật toán
- Các tiêu chuẩn của CTDL
- Vai trò của CTDL
- Độ phức tạp của thuật toán
- Thực hiện và hiệu chỉnh chương trình
- Tiêu chuẩn của chương trình



# Sự Cần Thiết Của Thuật Toán

- Tại sao sử dụng máy tính để xử lý dữ liệu?
  - Nhanh hơn.
  - Nhiều hơn.
  - Giải quyết những bài toán mà con người không thể hoàn thành được.
- Làm sao đạt được những mục tiêu đó?
  - Nhờ vào sự tiến bộ của kỹ thuật: tăng cấu hình máy ⇒ chi phí cao ☹
  - Nhờ vào các thuật toán hiệu quả: thông minh và chi phí thấp ☺

**“Một máy tính siêu hạng vẫn không thể cứu vãn một thuật toán tồi!”**



# Thuật Toán

- **Thuật toán:** Một dãy hữu hạn các chỉ thị có thể thi hành để đạt mục tiêu đề ra nào đó.
- **Ví dụ:** Thuật toán tính tổng tất cả các số nguyên dương nhỏ hơn n gồm các bước sau:

Bước 1:  $S=0, i=1;$

Bước 2: nếu  $i < n$  thì  $s=s+i;$

*Ngược lại:* qua bước 4;

Bước 3:

$i=i+1;$

*Quay lại bước 2;*

Bước 4: *Tổng cần tìm là S.*



# Các Tiêu Chuẩn Của Thuật Toán

- Xác định
- Hữu hạn
- Đúng
- Tính hiệu quả
- Tính tổng quát



# Biểu Diễn Thuật Toán

- Dạng ngôn ngữ tự nhiên
- Dạng lưu đồ (sơ đồ khối)
- Dạng mã giả
- Ngôn ngữ lập trình



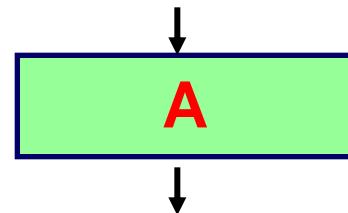
# Biểu Diễn Bằng Ngôn Ngữ Tự Nhiên

- NN tự nhiên thông qua các bước được tuần tự liệt kê để biểu diễn thuật toán.
- Ưu điểm:
  - Đơn giản, không cần kiến thức về về cách biểu diễn (mã giả, lưu đồ,...)
- Nhược điểm:
  - Dài dòng, không cấu trúc.
  - Đôi lúc khó hiểu, không diễn đạt được thuật toán.

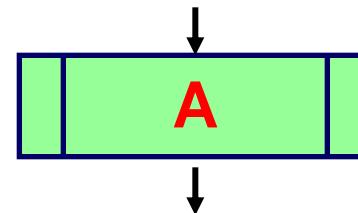


# Lưu Đồ

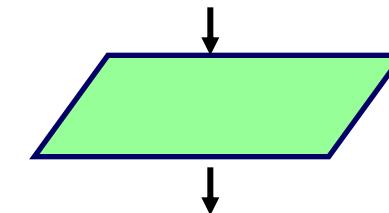
- Là hệ thống các nút, cung hình dạng khác nhau thể hiện các chức năng khác nhau.



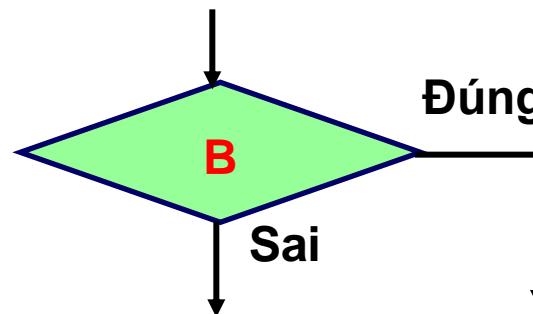
Thực hiện A



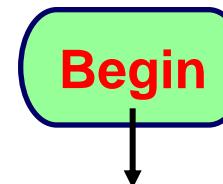
Gọi hàm A



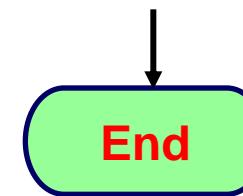
Vào / Ra dữ liệu



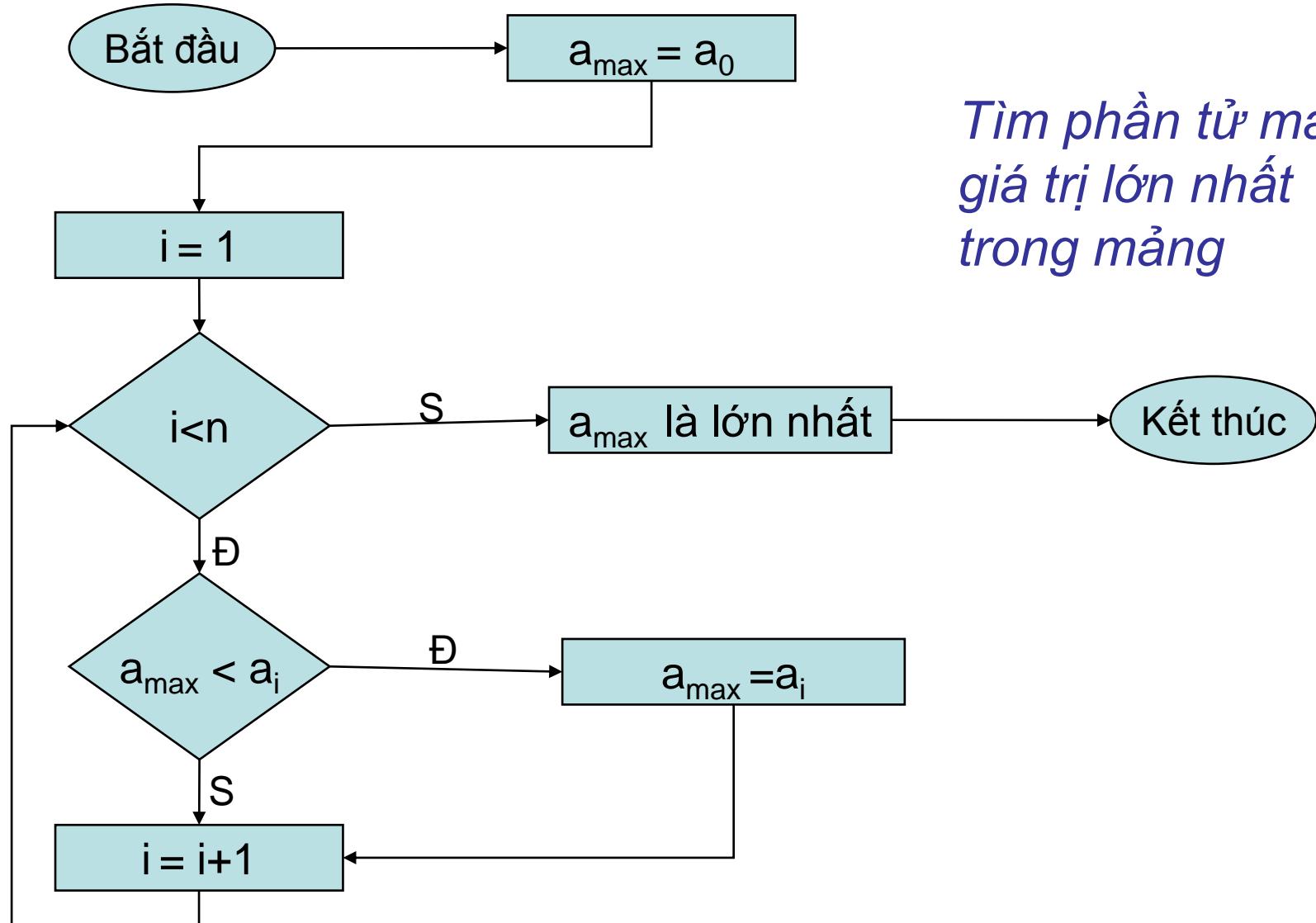
Điều kiện rẽ nhánh B



Nút giới hạn bắt đầu /  
kết thúc chương trình



# Biểu Diễn Bằng Lưu Đồ



# Biểu Diễn Bằng Mã Giả

- Ngôn ngữ tựa ngôn ngữ lập trình:
  - Dùng cấu trúc chuẩn hóa, chẵng hạn tựa Pascal, C.
  - Dùng các ký hiệu toán học, biến, hàm.
- Ưu điểm:
  - Đỡ cồng kềnh hơn lưu đồ khối.
- Nhược điểm:
  - Không trực quan bằng lưu đồ khối.



# Biểu Diễn Bằng Mã Giả

## ➤ Một số quy ước

1. Các biểu thức toán học

2. Lệnh gán: “=” ( $A \leftarrow B$ )

3. So sánh: “==”, “!=”

4. Khai báo hàm (thuật toán)

***Thuật toán <đơn vị> (<tham số>)***

***Input:*** *<dữ liệu vào>*

***Output:*** *<dữ liệu ra>*

*<Các câu lệnh>*

***End***



# Biểu Diễn Bằng Mã Giả

## 5. Các cấu trúc:

Cấu trúc chọn:

**if ... then ... [else ...] fi**

Vòng lặp:

**while ... do**

**do ... while (...)**

**for ... do ... od**

## 6. Một số câu lệnh khác:

Trả giá trị về: **return** [giá trị]

Lời gọi hàm: <Tên>(tham số)



# Biểu Diễn Bằng Mã Giả

- ❖ **Ví dụ:** Tìm phần tử lớn nhất trong mảng một chiều.

$a_{max} = a_0;$

$i = 1;$

**while** ( $i < n$ )

**if** ( $a_{max} < a_i$ )  $a_{max} = a_i;$

$i++;$

**end while;**



# Biểu Diễn Bằng Ngôn Ngữ Lập Trình

- Dùng ngôn ngữ máy tính (C, Pascal,...) để diễn tả thuật toán, CTDL thành câu lệnh.
- Kỹ năng lập trình đòi hỏi cần học tập và thực hành (nhiều).
- Dùng phương pháp tinh chế từng bước để chuyển hóa bài toán sang mã chương trình cụ thể.



# Độ Phức Tạp Của Thuật Toán

- Một thuật toán hiệu quả:
  - Chi phí cần sử dụng tài nguyên thấp: Bộ nhớ, thời gian sử dụng CPU, ...
- Phân tích độ phức tạp thuật toán:
  - **N** là khối lượng dữ liệu cần xử lý.
  - Mô tả độ phức tạp thuật toán qua một hàm **f(N)**.
  - Hai phương pháp đánh giá độ phức tạp của thuật toán:
    - Phương pháp thực nghiệm.
    - Phương pháp xấp xỉ toán học.



# Phương Pháp Thực Nghiêm

- Cài thuật toán rồi chọn các bộ dữ liệu thử nghiệm.
- Thống kê các thông số nhận được khi chạy các bộ dữ liệu đó.
- Ưu điểm: Dễ thực hiện.
- Nhược điểm:
  - Chịu sự hạn chế của ngôn ngữ lập trình.
  - Ảnh hưởng bởi trình độ của người lập trình.
  - Chọn được các bộ dữ liệu thử đặc trưng cho tất cả tập các dữ liệu vào của thuật toán: khó khăn và tốn nhiều chi phí.
  - Phụ thuộc vào phần cứng.



# Phương Pháp Xấp Xỉ

- Đánh giá giá thuật toán theo hướng tiệm xấp xỉ tiệm cận qua các khái niệm  $O()$ .
- Ưu điểm: Ít phụ thuộc môi trường cũng như phần cứng hơn.
- Nhược điểm: Phức tạp.
- Các trường hợp độ phức tạp quan tâm:
  - ➡ Trường hợp tốt nhất (phân tích chính xác)
  - ➡ Trường hợp xấu nhất (phân tích chính xác)
  - ➡ Trường hợp trung bình (mang tính dự đoán)



# Sử Phân Lớp Theo Độ Phức Tạp Của Thuật Toán

## ➤ Sử dụng ký hiệu BigO

- ↳ Hằng số :  $O(c)$
- ↳  $\log N$  :  $O(\log N)$
- ↳  $N$  :  $O(N)$
- ↳  $N \log N$  :  $O(N \log N)$
- ↳  $N^2$  :  $O(N^2)$
- ↳  $N^3$  :  $O(N^3)$
- ↳  $2^N$  :  $O(2^N)$
- ↳  $N!$  :  $O(N!)$



*Độ phức tạp tăng dần*

# Dữ Liệu

- Theo *từ điển Tiếng Việt*: số liệu, tư liệu đã có, được dựa vào để giải quyết vấn đề
- *Tin học*: Biểu diễn các thông tin cần thiết cho bài toán.



# Cấu Trúc Dữ Liệu

- Cách tổ chức lưu trữ dữ liệu.
- Các tiêu chuẩn của CTDL:
  - Phải biểu diễn đầy đủ thông tin.
  - Phải phù hợp với các thao tác trên đó.
  - Phù hợp với điều kiện cho phép của NNLT.
  - Tiết kiệm tài nguyên hệ thống.



# Vai Trò Của Cấu Trúc Dữ Liệu

- Cấu trúc dữ liệu đóng vai trò quan trọng trong việc kết hợp và đưa ra cách giải quyết bài toán.
- CTDL hỗ trợ cho các thuật toán thao tác trên đối tượng được hiệu quả hơn



# Thực Hiện Và Hiệu Chỉnh Chương Trình

- Chạy thử.
- Lỗi và cách sửa:
  - Lỗi thuật toán.
  - Lỗi trình tự.
  - Lỗi cú pháp.
- Xây dựng bộ test.
- Cập nhật, thay đổi chương trình theo yêu cầu (mới).



# Tiêu Chuẩn Của Một Chương Trình

- Tính tin cậy
  - Giải thuật + Kiểm tra cài đặt
- Tính uyển chuyển
- Tính trong sáng
  - Dễ hiểu và dễ chỉnh sửa
- Tính hữu hiệu.
  - Tài nguyên + giải thuật

