CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT 1

Số tiết lý thuyết: **45**

Số tiết thực hành: **30**

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

- Xem đề cương

Hình Thức Thi

- Quá trình: **10%**
- Giữa kỳ: **20% (đồ án)**
- Cuối kỳ: **70%**
  - Lý thuyết: **40%**
  - Seminar: **10%**
  - Thực hành: **20%**

TỔNG QUAN VỀ CTDL VÀ THUẬT TOÁN

Nội Dung

- Khái niệm CTDL & thuật toán
- Các kiểu dữ liệu
- Vai trò của CTDL & thuật toán
- Các tiêu chuẩn đánh giá CTDL & thuật toán
- Độ phức tạp của thuật toán
- Thực hiện và hiệu chỉnh chương trình
- Tiêu chuẩn của chương trình

Khái Niệm Về CTDL Và Thuật Toán

- Niklaus Wirth:

  CTDL + Thuật toán = Chương trình

- Cần nghiên cứu về thuật toán và CTDL!

Khái Niệm Về CTDL Và 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.*

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!"

Các Tiêu Chuẩn Của Thuật Toán

- Tính rõ ràng: Thuật toán được thể hiện bằng các câu lệnh minh bạch; các câu lệnh được sắp xếp theo thứ tự nhất định.
- Hữu hạn: một số hữu hạn các bước tính toán
- Đúng:
- Tính hiệu quả
  - thực hiện nhanh, tốn ít thời gian
  - Tiêu phí ít tài nguyên của máy
- Tính tổng quát: có thể áp dụng cho một lớp các bài toán có đầu vào tương tự nhau

Biểu Diễn Thuật Toán

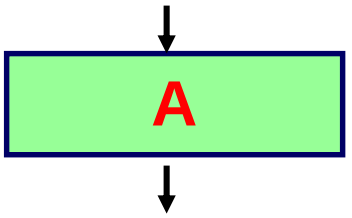
- Dạng ngôn ngữ tự nhiên
- Dạng lưu đồ (sơ đồ khối)
- Dạng mã giả
- Ngôn ngữ lập trình

Biểu Diễn Bằng Ngôn Ngữ Tự Nhiên

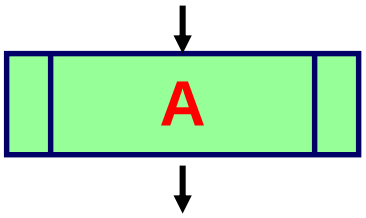
- NN tự nhiên thông qua các bước được tuần tự liệt kê để biểu diễn thuật toán.
- Ưu điểm:
  - Đơn giản, không cần kiến thức về về cách biểu diễn (mã giả, lưu đồ,...)
- Nhược điểm:
  - Dài dòng, không cấu trúc.
  - Đôi lúc khó hiểu, không diễn đạt được thuật toán.

Lưu Đồ

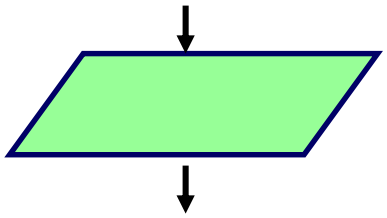
- Là hệ thống các nút, cung hình dạng khác nhau thể hiện các chức năng khác nhau.



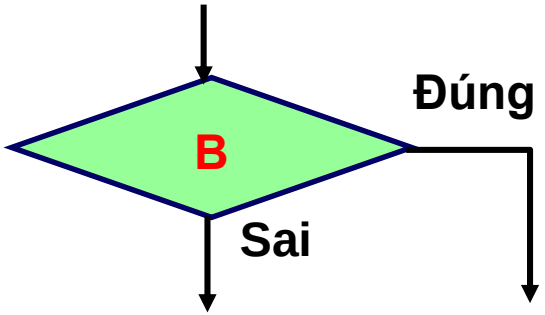
Thực hiện A



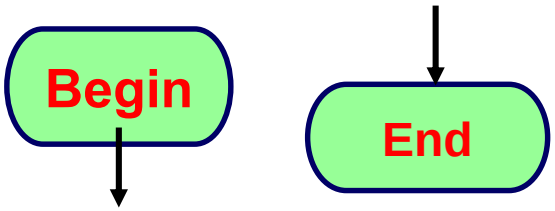
Gọi hàm A



Vào / Ra dữ liệu

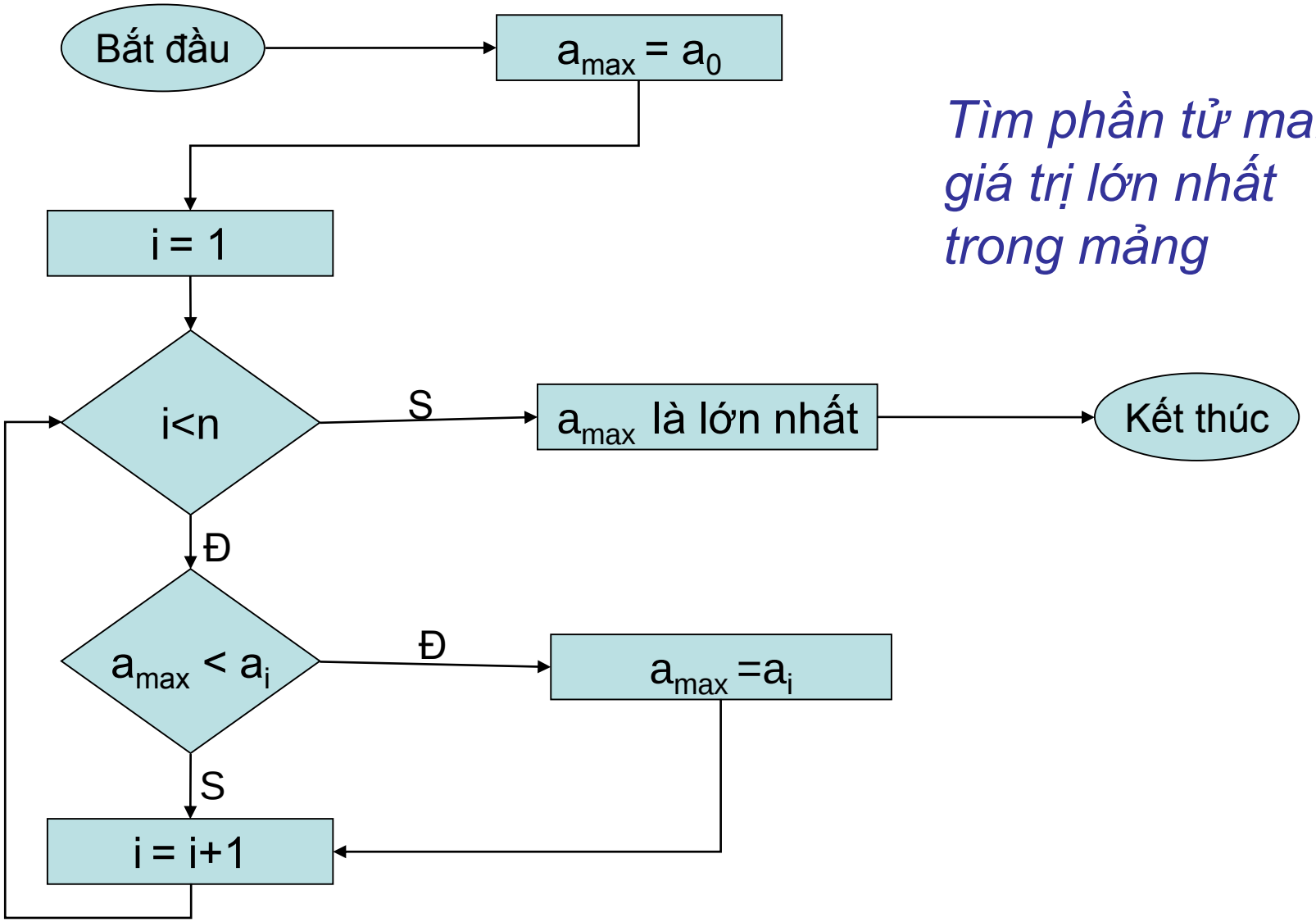


Điều kiện rẻ nhánh B



Nút giới hạn bắt đầu / kết thúc chương trình

Biểu Diễn Bằng Lưu Đồ



Tìm phần tử mang giá trị lớn nhất trong mảng

Biểu Diễn Bằng Mã Giả

- Ngôn ngữ tựa ngôn ngữ lập trình:
  - Dùng cấu trúc chuẩn hóa, chẳng 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←B)
  3. So sánh: “==”, “!=”
  4. Khai báo hàm (thuật toán)

Thuật toán *<tên TT> (<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 hoá 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 Nghiệm

- 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 tích dự đoán)

Phương Pháp Xấp Xỉ

- N: kích thước dữ liệu đâu vào
- O(c): hằng số, thời gian chạy không phụ thuộc N
  - Phép gán, nhập, xuất
- O(logN): logarit, chia đôi miền giá trị
  - Tìm kiếm nhị phân trên mảng 1 chiều
- O(N): tuyến tính
  - Vòng lặp for, tìm kiếm phần tử trong mảng 1 chiều
- O(NlogN): tách làm 2 bài toán nhỏ, lặp lại N lần
  - Quicksort
- $O(N^2)$
  - Hai vòng lặp for lồng nhau

Phương Pháp Xấp Xỉ

- Quy tắc cộng
  - O(max(f(n),g(n)))
    - f(n): thời gian thực hiện chương trình đầu
    - g(n): thời gian thực hiện chương trình sau

Ví dụ:

  - x = x*4; //O(c)
  - for (int i=0; i<n; i++) //O(N)

        cout << i;

- Độ phức tạp O(max(O(c),O(n))) = O(n)

Phương Pháp Xấp Xỉ

- Quy tắc nhân
  - Áp dụng cho các chương trình lồng vào nhau

Ví dụ:

```
for (int i=0; i<n; i++) //O(N)
        for (int j=0; j<n; j++) //O(N)
                cout << i + j;
```

- Độ phức tạp $O(N^2)$

Sự Phân Lớp Theo Độ Phức Tạp Của Thuật Toán

- **Sử dụng ký hiệu BigO**
  - Hằng số : O(c)
  - logN : O(logN)
  - N : O(N)
  - NlogN : O(NlogN)
  - 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.

Kiểu Dữ Liệu - Kiểu Dữ Liệu Trừu Tượng

- Kiểu dữ liệu là một tập hợp các giá trị và một tập hợp các phép toán trên các giá trị đó.
  - Boolean là một tập hợp có 2 giá trị TRUE, FALSE và các phép toán trên nó như OR, AND, NOT ….
  - Integer là tập hợp các số nguyên có giá trị từ -32768 đến 32767 cùng các phép toán cộng, trừ, nhân, chia, Div, Mod, …

Kiểu Dữ Liệu - Kiểu Dữ Liệu Trừu Tượng

- Kiểu dữ liệu có hai loại:
  - Kiểu dữ liệu sơ cấp là kiểu dữ liệu mà giá trị dữ liệu của nó là đơn nhất.
    - Ví dụ: kiểu Boolean, Integer….
  - Kiểu dữ liệu có cấu trúc (hay còn gọi là cấu trúc dữ liệu) là kiểu dữ liệu mà giá trị dữ liệu của nó là sự kết hợp của các giá trị khác.
    - Ví dụ: ARRAY là một cấu trúc dữ liệu.

Kiểu Dữ Liệu - Kiểu Dữ Liệu Trừu Tượng

- Kiểu dữ liệu trừu tượng: là một mô hình toán học cùng với một tập hợp các phép toán trừu tượng được định nghĩa trên mô hình đó.
  - Ví dụ: tập hợp số nguyên cùng với các phép toán hợp, giao, hiệu.
- Kiểu dữ liệu trừu tượng là một kiểu dữ liệu do chúng ta định nghĩa ở mức khái niệm (conceptual), nó chưa được cài đặt cụ thể bằng một ngôn ngữ lập trình.
- Khi cài đặt một kiểu dữ liệu trừu tượng trên một ngôn ngữ lập trình cụ thể, chúng ta phải thực hiện hai nhiệm vụ:
  - Biểu diễn kiểu dữ liệu trừu tượng bằng một cấu trúc dữ liệu hoặc một kiểu dữ liệu trừu tượng khác đã được cài đặt.
  - Viết các chương trình con thực hiện các phép toán trên kiểu dữ liệu trừu tượng mà ta thường gọi là cài đặt các phép 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 của thuật toán mà ta chọn để giải quyết bài toán.
  - 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
  - Đáp ứng quy trình làm phần mềm.
- Tính trong sáng
  - Dễ hiểu và dễ chỉnh sửa
- Tính hữu hiệu.
  - Tài nguyên + giải thuật

Quy Trình Làm Phần Mềm

- *Bước 0:* Ý tưởng (concept).
- *Bước 1:* Xác định yêu cầu (Requirements Specification).
- *Bước 2:* Phân tích (Analysis).
- *Bước 3:* Thiết kế (Design).
- *Bước 4:* Cài đặt (Implementation).
- *Bước 5:* Thử nghiệm (Testing).
- *Bước 6:* Vận hành, theo dõi và bảo dưỡng (Operation, follow-up and Maintenance).