ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

TIN HỌC ĐẠI CƯƠNG

Phần I - Chương 3: Thuật toán

Nội dung chính

3.1 Giải quyết bài toán

- Khái niệm về bài toán
- Quá trình giải quyết bài toán bằng máy tính
- Phương pháp giải quyết bài toán bằng MT

3.2 Thuật toán

- Khái niệm
- Biểu diễn thuật toán
- Thuật toán đệ quy
- Một số thuật toán ví dụ

3.1 Giải quyết bài toán

- Khái niệm về bài toán
- Quá trình giải quyết bài toán bằng máy tính
- Phương pháp giải quyết bài toán bằng máy tính

Khái niệm bài toán

$$A \rightarrow B$$

- A: Giả thiết, điều kiện ban đầu
- B: Kết luận, mục tiêu cần thực hiện
- $\rightarrow$: Suy luận, giải pháp cần xác định
- Giải quyết vấn đề/bài toán: Từ **A** dùng một số **hữu hạn các bước** suy luận có lý hoặc hành động thích hợp để đạt **B**

Trong khoa học máy tính

$$A \rightarrow B$$

- A là Input
- B là Output
- $\rightarrow$ là Chương trình cho phép biến đổi A thành B

Chương trình

- Chương trình
  - Cách mã hóa lại thuật toán để giải quyết vấn đề/bài toán đã cho
  - Tạo thành từ các lệnh cơ bản của máy tính
- Khó khăn
  - Tồn tại các yếu tố không xác định: A và B không đầy đủ, rõ ràng
- Giải quyết bài toán trên máy tính
  - Vấn đề tổ chức dữ liệu và thiết kế giải thuật

Cấu trúc dữ liệu + Giải thuật = Chương trình

Thiết kế thuật giải

- **Thực hiện bởi con người**
  - Là cách thức chủ yếu, dựa trên
    - Những thông tin được phản ánh rõ ràng trong A, B hoặc →
    - Các tri thức của con người
- **Tự động hóa xây dựng thuật giải**
  - Lĩnh vực mới, đang được nghiên cứu
  - Cần phải biểu diễn nội dung và các tri thức liên quan dưới dạng tương minh và đầy đủ

3.1 Giải quyết bài toán

- Khái niệm về bài toán
- Quá trình giải quyết bài toán bằng máy tính
- Phương pháp giải quyết bài toán bằng máy tính

Máy tính & Lập trình viên

- Máy tính
  - Chỉ làm được những gì được bảo.
  - **Không thông minh:** không thể tự phân tích vấn đề và đưa ra giải pháp.
  - Không thể dùng giải quyết các vấn đề liên quan đến hành động vật lý hoặc biểu thị cảm xúc

- Lập trình viên
  - Phân tích vấn đề
  - Tạo ra các chỉ dẫn để giải quyết vấn đề (xây dựng chương trình).
    - Máy tính sẽ thực hiện các chỉ dẫn này.

Các bước giải quyết bài toán

1. Xác định bài toán
2. Lựa chọn phương pháp giải
3. Xây dựng thuật toán hoặc thuật giải
4. Cài đặt chương trình
5. Hiệu chỉnh chương trình
6. Thực hiện chương trình

Bước 1: Xác định bài toán

- Mô tả bài toán cần giải quyết
  - **Dữ liệu vào:** Danh sách các dữ kiện vào
  - **Điều kiện vào:** Ràng buộc, quan hệ giữa chúng
  - **Dữ liệu ra**: Danh sách các dữ liệu ra
  - **Điều kiện ra:** Ràng buộc, quan hệ giữa chúng
- Đánh giá, nhận định tính khả thi của bài toán
  - Thời gian, kinh phí, nguồn lực,…

Ví dụ: Bài toán tìm ƯSCLN của 2 số nguyên dương

- **Nhập:** 2 số X, Y
- **Điều kiện nhập:** X, Y nguyên dương
- **Dữ liệu ra**: Z
- **Điều kiện ra:** Z là ƯSCLN(X,Y)

Bước 2: Lựa chọn phương pháp

- Tồn tại nhiều phương pháp khác nhau
  - Khác nhau về thời gian thực hiện, chi phí lưu trữ dữ liệu, độ chính xác…
- Tùy theo nhu cầu cụ thể và khả năng xử lý tự động được sử dụng để lựa chọn phương pháp thích hợp

Ví dụ: Bài toán sắp xếp dãy số

- Nổi bọt, Vun đống, Sắp xếp nhanh,…

Bước 3: Xây dựng thuật giải

- Xây dựng mô hình chặt chẽ, chính xác và chi tiết cho phương pháp đã lựa chọn
- Lặp liên tiếp các bước sau để thuật toán ngày càng hoàn chỉnh hơn (*quá trình tinh chỉnh từng bước*)
  1. Xác định và chính xác hóa các thao tac
     - Để đạt được kết quả cần làm gì?
  2. Xác định các dữ liệu cần dùng và tính chất của chúng
     - Để thực hiện, thao tác cần gì và sẽ tạo ra gì?
  3. Xác định trình tự các thao tác
     - Thao tác nào cần làm trước
     - Thao tác thực hiện 1 hay nhiều lần, thực hiện trong điều kiện nào..?

Bước 3: Xây dựng thuật giải (tiếp)

- *Quá trình tinh chỉnh từng bước* dừng lại khi
  - Yêu cầu cho biết 1 hay nhiều đại lượng
  - Tính một đại lượng theo công thức đã biết rõ
  - Thông báo một hay nhiều kết quả đã xử lý
- Sau khi tinh chỉnh cần phải diễn tả giải thuật dưới dạng chuẩn
  - Ngôn ngữ liệt kê các hành động
  - Sơ đồ khối…

Bước 4: Cài đặt chương trình

Mã hóa giải thuật bằng một ngôn ngữ lập trình

- Thay thế các thao tác bằng các lệnh tương ứng của ngôn ngữ sử dụng
  - **Thao tác:** In ra môt thông báo
  - **Câu lệnh:** printf("….")/ write(".....");
- Lựa chọn ngôn ngữ lập trình, tùy theo bài toán giải quyết
  - NNLT bậc thấp: Hợp ngữ
  - NNLT bậc cao: C, Pascal, Java,..

Bước 5: Hiệu chỉnh chương trình

Chạy thử để phát hiện và điều chỉnh các sai sót có thể có ở bước 4.

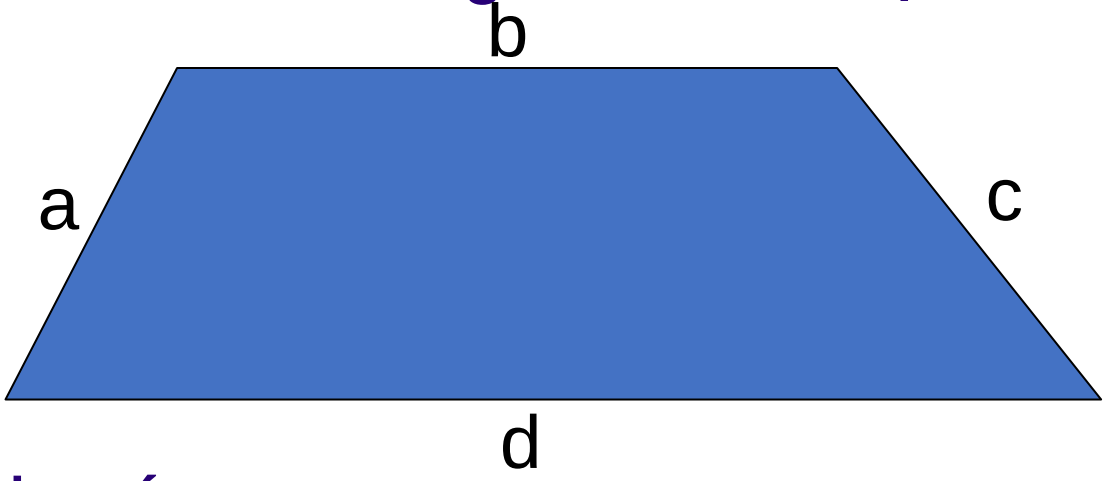
- Lỗi cú pháp:
  - Viết sai cú pháp của ngôn ngữ lập trình lựa chọn
- Lỗi ngữ nghĩa
  - Mã hóa sai giải thuật
  - Giải thuật sai

Bước 6: Thực hiện chương trình

- Cho máy tính thực hiện chương trình.
- Tiến hành phân tích kết quả thu được
  - Kết quả đó có phù hợp hay không.
    - Không phù hợp kiểm tra lại toàn bộ các bước.

Ví dụ

Tính diện tích hình thang khi biết 4 cạnh

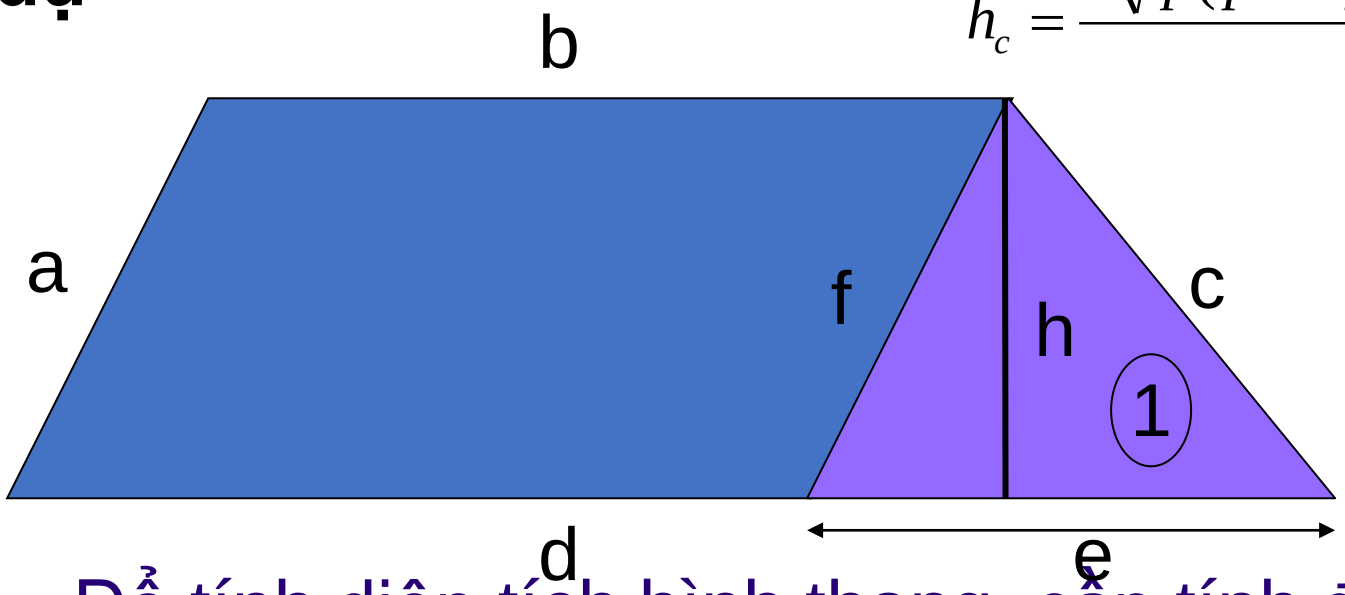


Mô tả bài toán

- Nhập: 4 cạnh a, b, c, d
- Điều kiện nhập: a, b, c, d > 0 và d > b
- Xuất: Một giá trị số tương ứng diện tích hình thang

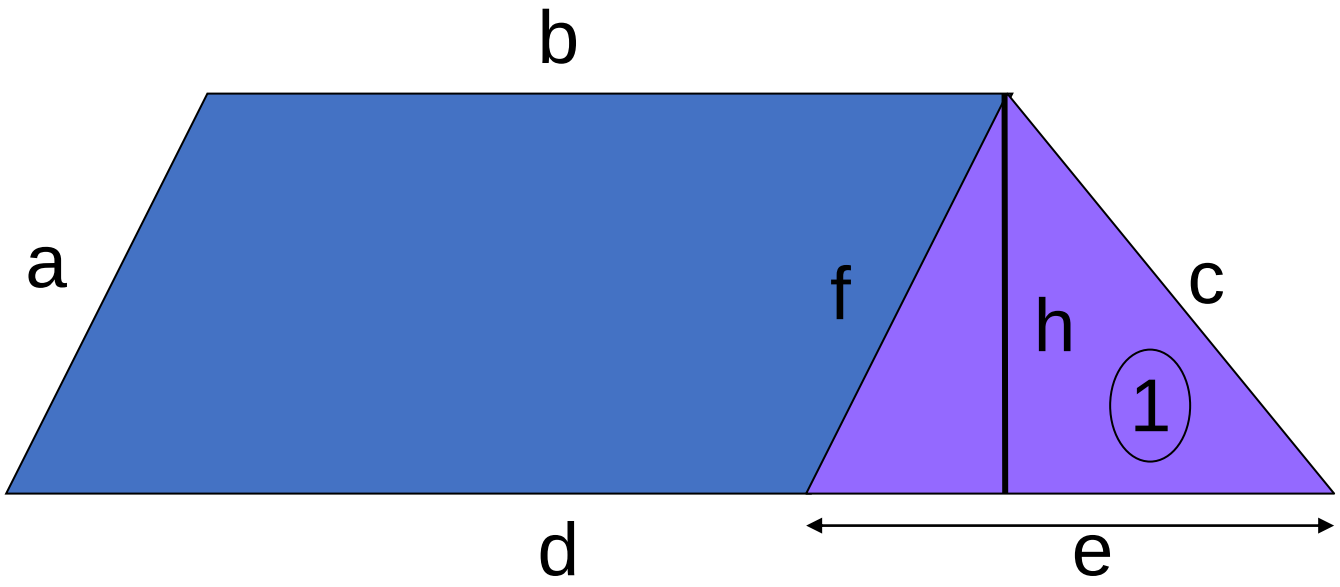
Ví dụ

$$h_c = \frac{2\sqrt{p(p-a)(p-b)(p-c)}}{c}$$



1. Để tính diện tích hình thang, cần tính đường cao (công thức S = h(b+d)/2)
2. Tính đường cao h, cần phải biết 3 cạnh của tam giác (1)
3. Cần tính cạnh tam giác (1) trước khi tính đường cao h

Ví dụ



- Để tính cạnh của tam giác (1) cần biết các cạnh của hình thang
- Các cạnh của hình thang là dữ kiện cho biết của đề bài ➔ bài toán có thể giải được

Chuẩn hóa thuật toán

1. Nhập các số a, b, c, d
2. Tính các cạnh của tam giác (1)
   - f←a
   - e←d-c
   - p ←(f+e+c)/2
3. Tính chiều cao của tam giác (1)

$$h = \frac{2\sqrt{p(p-e)(p-f)(p-c)}}{e}$$

4. Tính diện tích hình thang S= h(d+b)/2
5. In kết quả S
6. Kết thúc

3.1 Giải quyết bài toán

- Khái niệm về bài toán
- Quá trình giải quyết bài toán bằng máy tính
- Phương pháp giải quyết bài toán bằng máy tính

Phương pháp giải quyết bài toán bằng máy tính

- Xác định trực tiếp lời giải
- Tìm kiếm lời giải

Xác định trực tiếp lời giải

- Bài toán thông dụng, đơn giản, đã có lời giải từ trước.
- Xác định trực tiếp được lời giải qua
  - Các thủ tục tính toán theo công thức, hệ thức, định luật…
  - Các thủ tục bao gồm một số hữu hạn các thao tác sơ cấp, có thể chuyền thành các thuật toán và chương trình chạy trên máy tính.

Hướng xác định trực tiếp lời giải

- Trường hợp dùng các công thức lặp để tính gần đúng nghiệm của bài toán.
  - Lời giải xác định bởi các công thức lặp có thể xấp xỉ lời giải thật sự của bài toán với độ chính xác tăng theo quá trình lặp.
    - Ví dụ: giải phương trình f(x)=0 bằng PP chia đôi
  - Đây là hạn chế khi tính toán thủ công nhưng là thế mạnh của máy tính.
  - Được xem là cách xác định trực tiếp lời giải

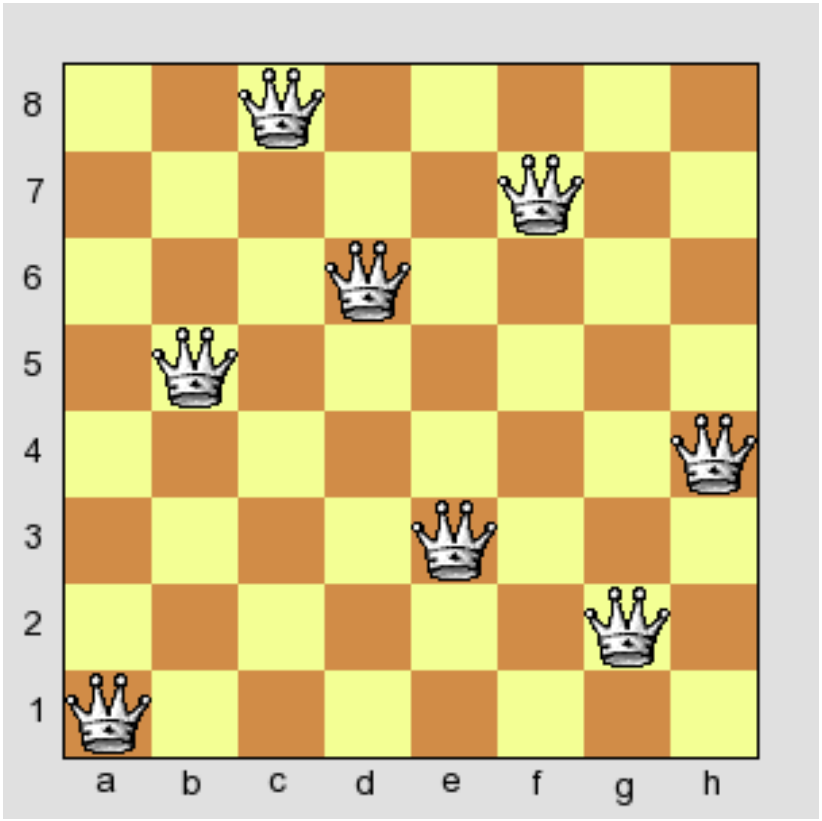
Hướng tìm kiếm lời giải

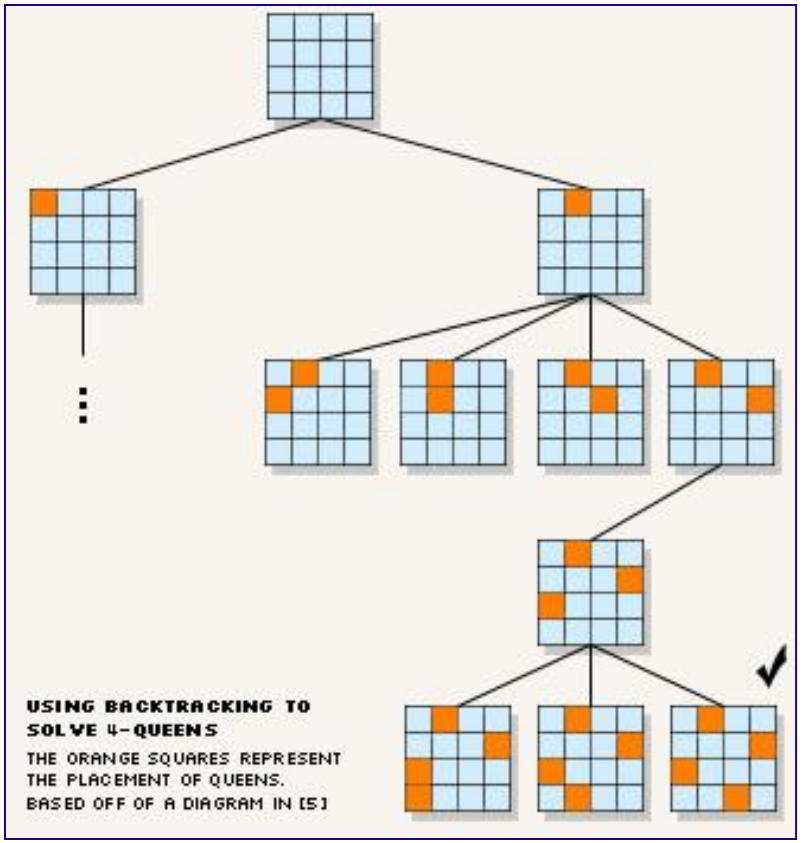
- Cách tiếp cận dựa theo nguyên lý "**thử - sai**".
- Ứng dụng hiệu quả cho một số bài toán - vấn đề phức tạp
- Có nhiều phương pháp, cách làm đã được đề xuất

Hướng tìm kiếm lời giải→Một số phương pháp

- Phương pháp liệt kê hay vét cạn:
  - Xác định tập các khả năng chứa các lời giải và cách thức liệt kê của từng khả năng để thử, không bỏ sót một khả năng nào.
- Phương pháp thử ngẫu nhiên:
  - Thử một số khả năng được chọn ngẫu nhiên trong tập (*rất lớn*) các khả năng.
  - Khả năng thành công tùy theo chiến lược chọn ngẫu nhiên và một số điều kiện cụ thể của bài toán.
- Chia bài toán thành bài toán con:
  - Chia cho tới khi bài toán ban đầu được quy thành bài toán con có lời giải
- Phương pháp quay lui:
  - Đánh dấu các thử nghiệm thất bại và thử khả năng mới (quay lui tìm đường khác).

Hướng tìm kiếm lời giải→Ví dụ bài toán 8 hậu





3.2 Thuật toán

- Khái niệm
- Biểu diễn thuật toán
- Thuật toán đệ quy
- Một số thuật toán ví dụ

Khái niệm

- Thuật toán (*algorithm*) là khái niệm cơ sở của Toán học và Tin học
- Nghiên cứu thuật toán đóng vai trò quan trọng trong khoa học máy tính
  - Máy tính chỉ có khả năng thực hiện công việc theo một thuật toán.
  - Thuật toán chỉ đạo máy tính từng bước phải làm gì.

Khái niệm

- Một tập các lệnh hay chỉ thị nhằm hướng dẫn việc thực hiện một công việc nào đó
- Bao gồm một dãy hữu hạn các chỉ thị rõ ràng và có thể thi hành được, được bố trí theo một trình tự nhất định, cần thực hiện trên những dữ liệu vào sao cho sau một số hữu hạn bước ta thu được kết quả của bài toán cho trước
- Thuật toán là sự thể hiện của một phương pháp để giải quyết một vấn đề

Ví dụ

Tìm phần tử lớn nhất trong một dãy hữu hạn các số nguyên

1. Đặt giá trị lớn nhất tạm thời (Max) bằng số nguyên đầu tiên của dãy
   Max là giá trị lớn nhất ở mỗi giai đoạn thực hiện
2. Nếu tất cả số nguyên trong dãy đã được xét, thực hiện bước 5
3. So sánh số nguyên kế tiếp trong dãy với Max
   - Nếu lớn hơn Max thì thay Max bằng số nguyên này.
4. Lặp lại bước 2
5. Thông báo: Max là giá trị lớn nhất trong dãy số.

Định nghĩa trong Khoa học máy tính

Thuật toán để giải một bài toán là một dãy hữu hạn các thao tác và trình tự thực hiện các thao tác đó sao cho sau khi thực hiện dãy thao tác này theo trình tự đã chỉ ra, với đầu vào (input) ta thu được kết quả đầu ra (output) mong muốn

Thao tác/lệnh

- Là hành động cần được thực hiện bởi cơ chế của thuật toán
- Các thao tác (*lệnh*) sẽ biến đổi bài toán từ trạng thái trước tới trạng thái sau
- Dãy các thao tác cần thiết sẽ biến đổi bài toán từ trạng thái ban đầu đến kết quả
- Các thao tác có thể phân tích thành thao tác khác nhỏ hơn
- Thứ tự thao tác là quan trọng
  - Cùng tập thao tác, thứ tự khác nhau dẫn đến kết quả khác nhau
- Cơ cấu thể hiện trình tự thực hiện các thao tác gọi là **Cấu trúc điều khiển**
  - Có 3 loại cơ bản: Tuần tự, Lặp, Rẽ nhánh

Các đặc trưng của thuật toán

Khi mô tả thuật toán, cần chú ý các đặc trưng

- Nhập
- Xuất
- Tính xác định
- Tính hữu hạn
- Tính hiệu quả
- Tính tổng quát

Nhập/Xuất

- **Nhập (input):**
  - Các giá trị "*đầu vào*" (input values) từ một tập hợp nhất định nào đó.
- **Xuất (output):**
  - Những giá trị trả về (output values) thuộc một tập hợp nhất định nào đó thể hiện lời giải cho bài toán/vấn đề
  - Tương ứng với tập hợp các giá trị nhập

Tính xác định

- Các bước trong thuật toán phải chính xác rõ ràng, không gây sự nhập nhằng nhầm lẫn
- Cùng một điều kiện nhập, cùng một giải thuật thì 2 bộ VXL (người, máy) phải cho ra cùng một kết quả

Tính hữu hạn

- Trong mọi trường hợp của dữ liệu vào, thuật toán phải cho ra kết quả sau một thời gian hữu hạn
  - Thời gian có thể phụ thuộc vào từng bài toán cụ thể, dữ liệu cụ thể các thuật toán khác nhau cho một bài toán
  - Ví dụ bài toán sắp xếp dãy số
    - Buble sort, Quick sort,..

Tính hiệu quả

- Thực hiện thuật toán cần
  - Thời gian thực thi
  - Các công cụ hỗ trợ (giấy, bộ nhớ,..) để lưu kết quả trung gian
- Độ phức tạp thuật toán: Thời gian và các công cụ hỗ trợ
  - Thuật toán càng hiệu quả độ phức tạp càng nhỏ
  - Trong máy tính, thường quan tâm tới
    - Thời gian thực hiện: số thao tác cơ bản cần thực hiện
    - Nhu cầu bộ nhớ mà thuật toán sử dụng

Tính tổng quát

Thuật toán có tính tổng quát cao nếu có thể giải bất kỳ bài toán nào trong một lớp lớn các bài toán

Ví dụ

Thuật toán giải phương trình $ax^2+bx+c=0$ tổng quát hơn thuật toán giải phương trình $x^2+5X+6=0$

3.2 Thuật toán

- Khái niệm
- Biểu diễn thuật toán
- Thuật toán đệ quy
- Một số thuật toán ví dụ

Biểu diễn thuật toán

- Mô tả thuật toán một cách tường minh
  - Truyền đạt thuật toán cho người khác
  - "*Truyền đạt*" thuật toán cho máy tính
- Phương pháp:
  1. Ngôn ngữ tự nhiên: người - người
  2. Lưu đồ thuật toán: sử dụng ký pháp
  3. Mã giả: mô phỏng ngôn ngữ lập trình
  4. Ngôn ngữ lập trình: chương trình máy tính

Ngôn ngữ tự nhiên

- Nguyên tắc:
  - Sử dụng ngôn ngữ tự nhiên để liệt kê các bước của thuật toán
- Đặc điểm
  - Không yêu cầu phải có một số kiến thức đặc biệt
  - Dài dòng,
  - Không làm nổi bật cấu trúc của thuật toán.

Biểu diễn bằng ngôn ngữ tự nhiên

- **Bài toán**: giải phương trình bậc nhất

$$ax + b = 0$$

- **B1**: Nhập a và b.
- **B2**: Nếu a<>0 thì hiển thị "Phương trình có 1 nghiệm duy nhất x = -b/a". Kết thúc.
- **B3**: (a=0) Nếu b <> 0 thì hiển thị "Phương trình vô nghiệm". Kết thúc.
- **B4**: (a=0)(b=0) Hiển thị "Phương trình vô số nghiệm". Kết thúc.

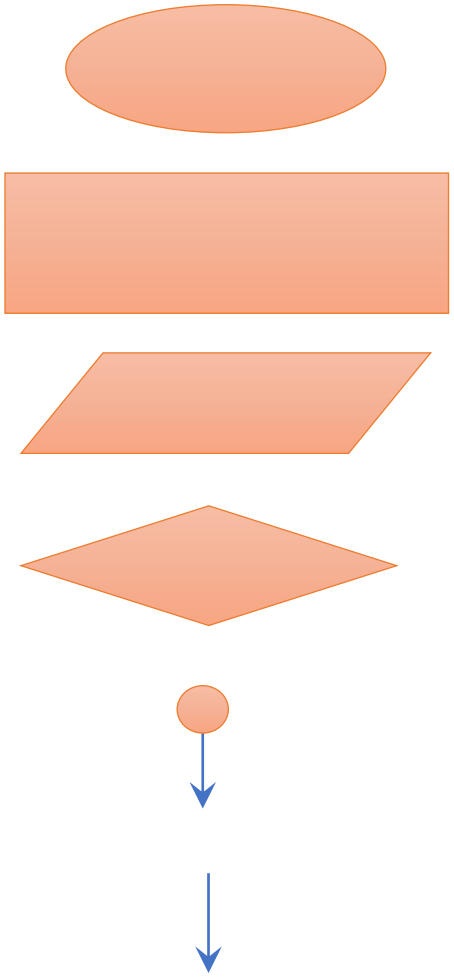
Ví dụ

Tìm giá trị lớn nhất của một dãy N số nguyên

- B1: Nhập N
- B2: Nhập dãy số a_i gồm N số.
- B3: Gán giá trị a_1 cho Max, 2 cho biến i ($i \leftarrow 2$)
- B4: Nếu i > N, thực hiện bước 8
- B5: Nếu a_i > Max, gán giá trị a_i cho Max.
- B6: Tăng i lên 1 đơn vị.
- B7: Quay lên B4.
- B8: Thông báo: Max là giá trị lớn nhất dãy
- B9: Kết thúc.

Lưu đồ thuật toán

- Ký pháp



Bắt đầu hoặc ***kết thúc***

Thao tác tính toán

Lệnh vào, lệnh ra

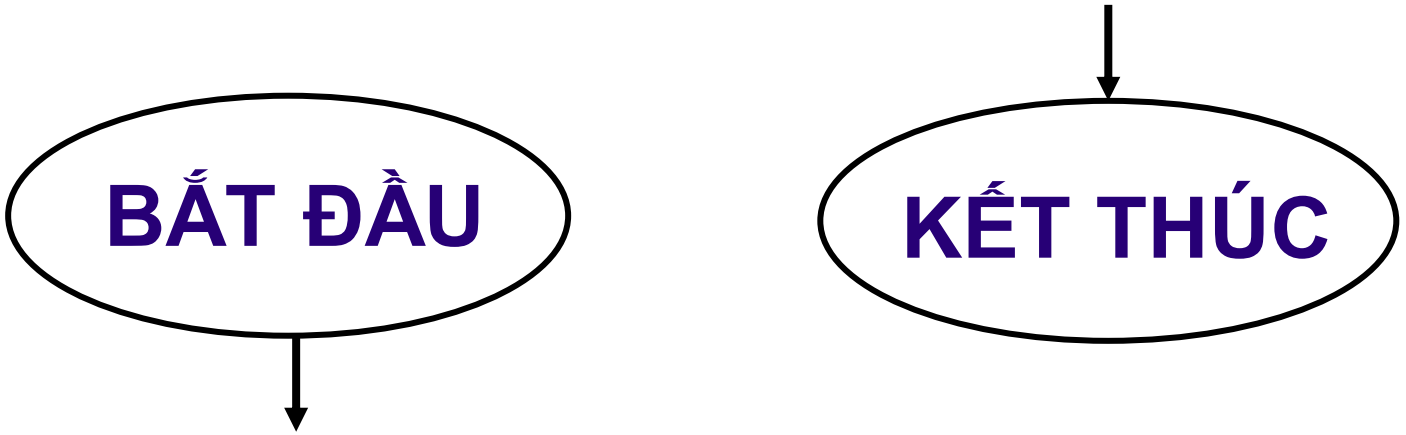
Kiểm tra điều kiện, rẽ nhánh

Nối tiếp đoạn lệnh

Luồng thực hiện

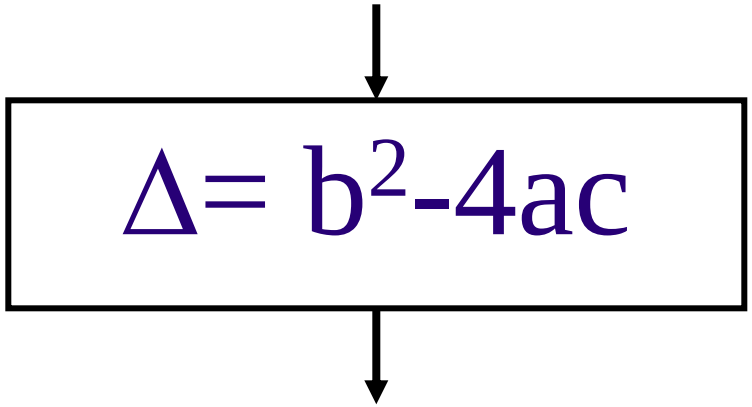
Khối giới hạn

- 2 loại khối giới hạn: đầu và cuối
- Ghi rõ điểm bắt đầu và kết thúc (dừng) của thuật toán



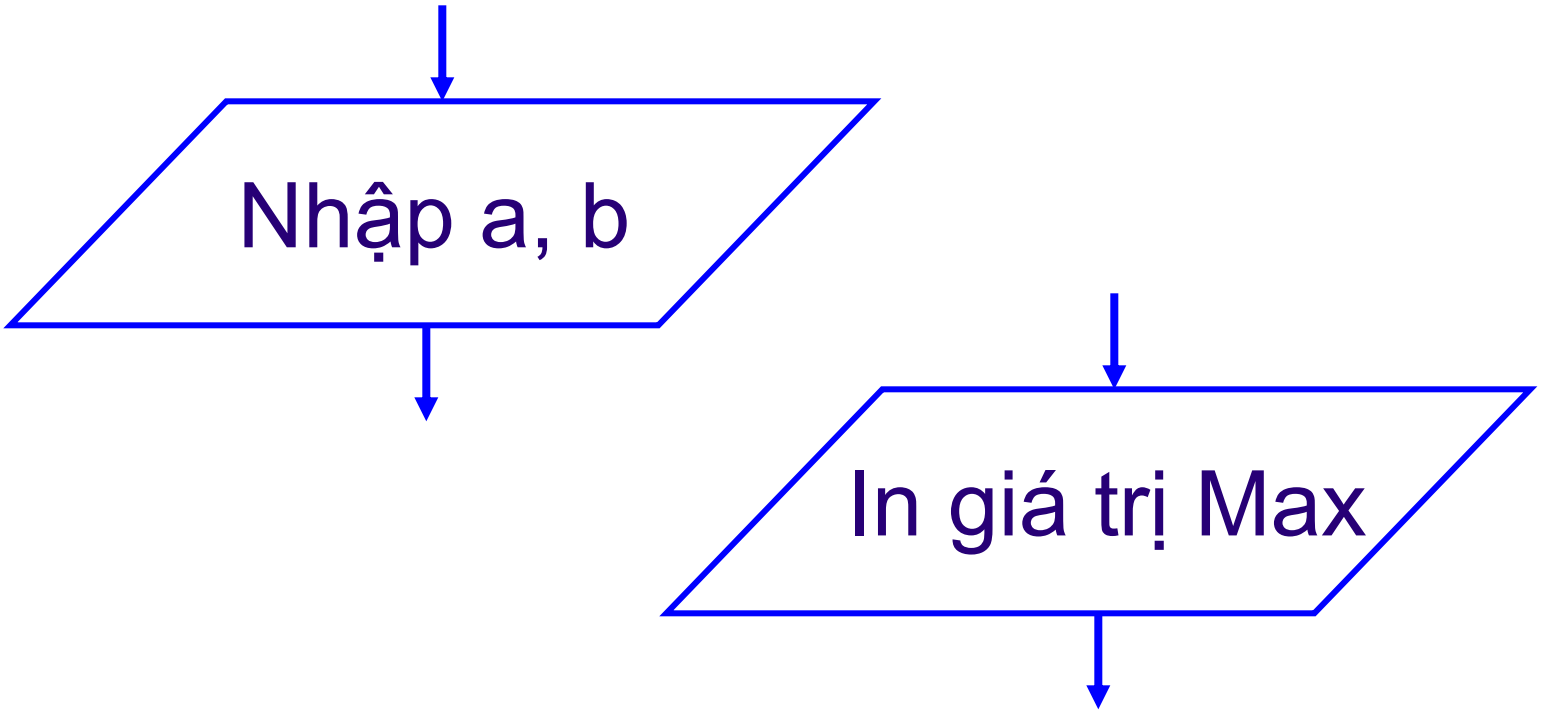
Khối thao tác

Hình chữ nhật chứa dãy các lệnh xử lý dữ liệu



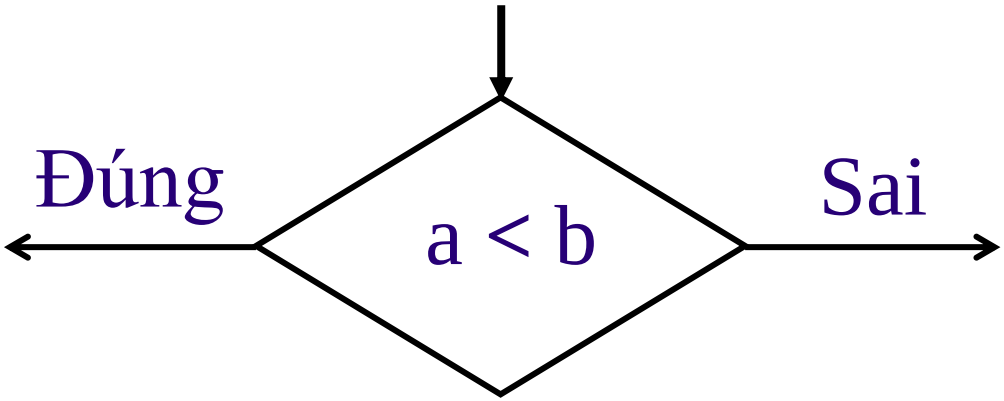
Khối vào/ra dữ liệu

Hình bình hành chứa thao tác nhập/ xuất dữ liệu



Khối điều kiện

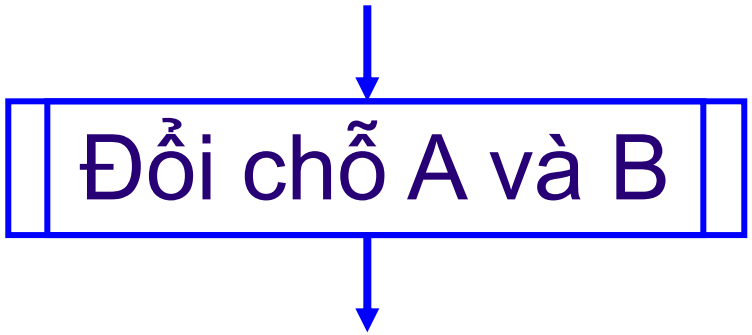
Là hình thoi chứa một điều kiện/biểu thức logic cần kiểm tra, một luồng vào, và hai luồng ra.



Khối gọi chương trình con

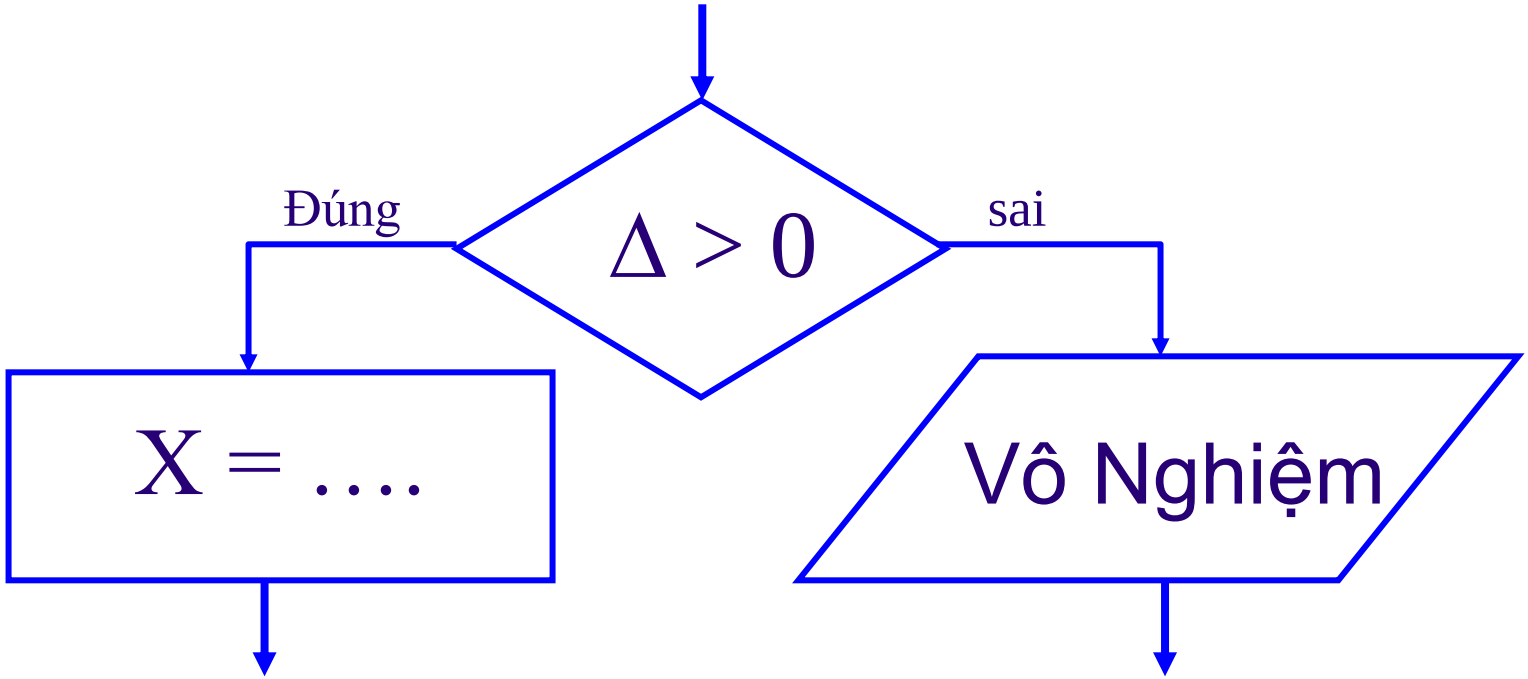
Là hình chữ nhật, cạnh kép chứa tên một chương trình con cần thực hiện

- Chương trình con: Thuật toán đã biết
- Nhằm giảm độ phức tạp, tăng tính cấu trúc của sơ đồ



Cung

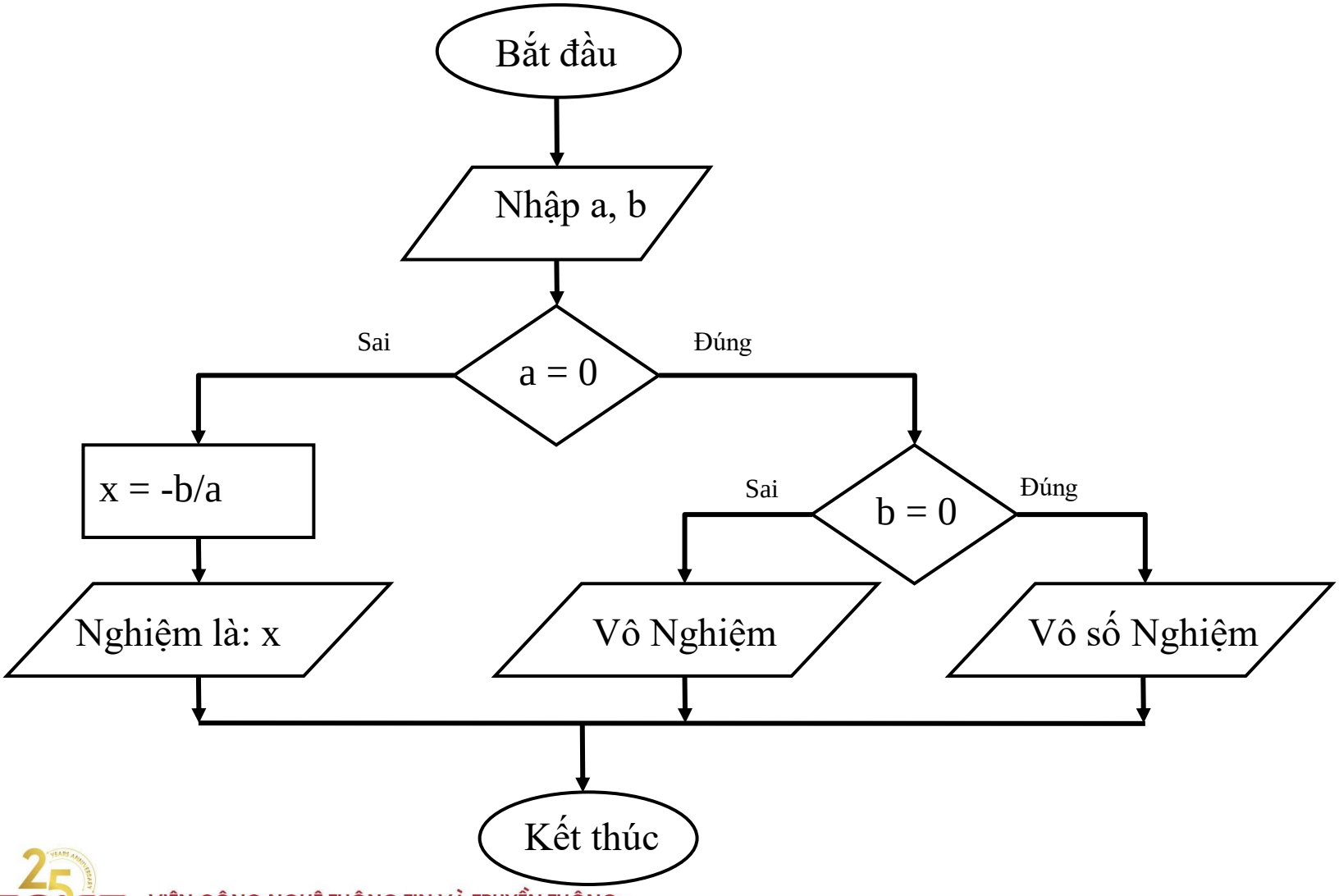
Là các đường nối từ nút này đến nút khác của lưu đồ



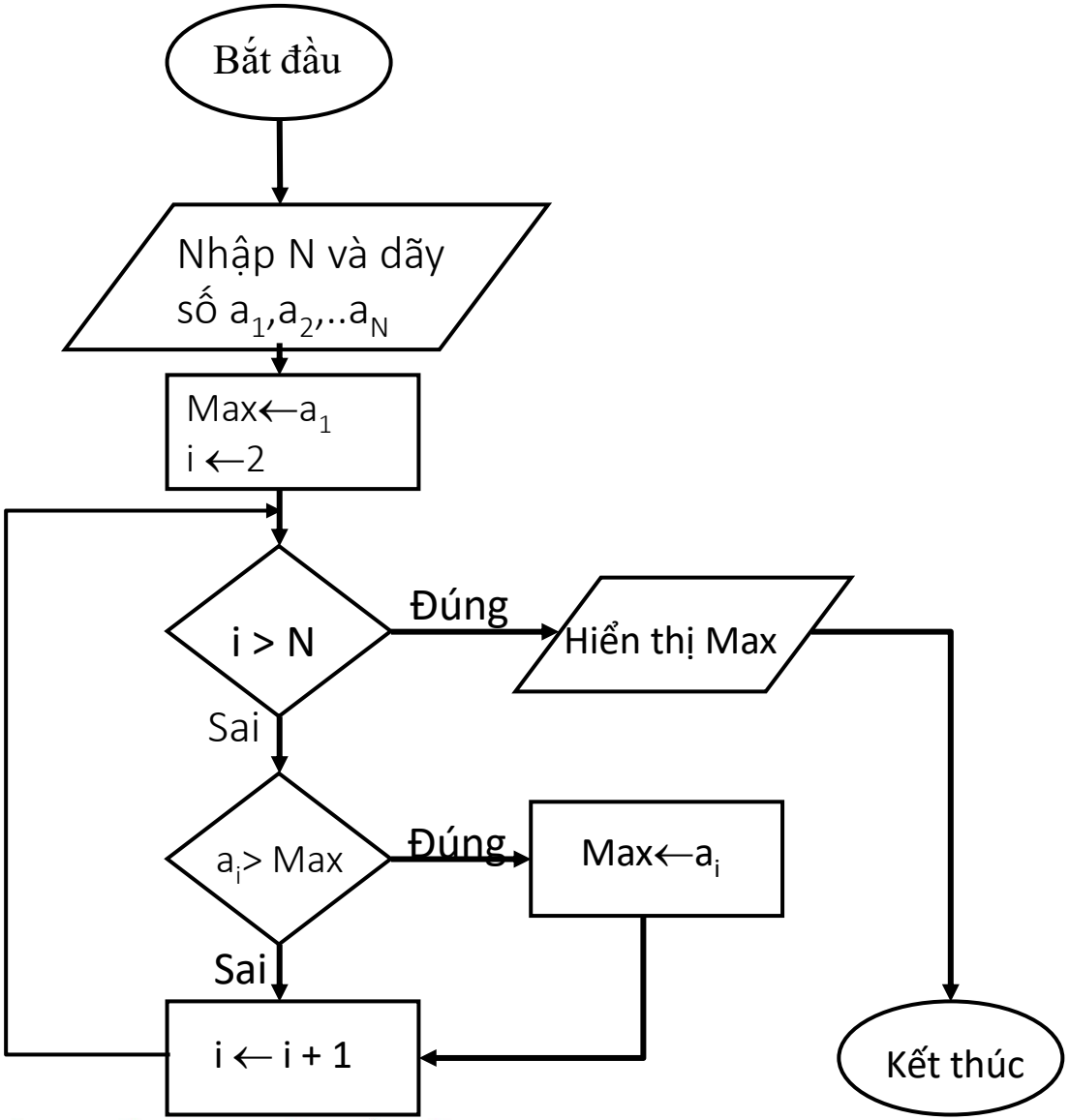
Hoạt động

- Bắt đầu từ khối giới hạn đầu tiên (bắt đầu).
- Đi theo luồng thực thi (mũi tên).
- Thực hiện các thao tác được ghi trong nút
- Nếu là nút điều kiện: kiểm tra điều kiện rồi đi theo luồng thực thi tương ứng.
- Thuật toán sẽ dừng khi gặp nút kết thúc

VD: Biểu diễn sơ đồ khối



VD: Biểu diễn sơ đồ khối



Mã giả

- Mô tả thuật toán theo ngôn ngữ phỏng theo ngôn ngữ lập trình
  - Sử dụng các mệnh đề có cấu trúc chuẩn hóa
  - Vẫn dùng ngôn ngữ tự nhiên.
    - Có thể sử dụng các ký hiệu toán học
    - Có thể sử dụng cấu trúc kiểu thủ tục để trình bày thuật toán đệ quy/thuật toán phức tạp..
- Đặc điểm:
  - Tiện lợi, đơn giản, và dễ hiểu.
- Các cấu trúc thường gặp:
  - Gán, lựa chọn, lặp, nhảy, gọi hàm

Thao tác gán giá trị

- Mục đích:
  - Đặt giá trị cho một biến
- Biểu diễn:

$Max := a_1$

$Max \leftarrow a_1$

$n \leftarrow n + 1$

Cấu trúc lựa chọn/rẽ nhánh

if <điều kiện> **then** <hành động>

endif

Hoặc là

if <điều kiện> **then** <hành động>

else <hành động>

endif

Cấu trúc lặp

```
while <điều kiện> do
      <hành động>
end while
```

```
repeat
   <hành động>
until <điều kiện>
```

```
for biến←giá trị đầu to giá trị cuối do
   <hành động>
end for
```

```
for biến ←giá trị đầu downto giá trị cuối
   do
   <h/động>
end for
```

Lệnh nhảy không điều kiện

Nhảy đến vị trí có nhãn là L

goto L

Chú ý: không nên dùng!

Hàm/Thủ tục

- **Khai báo hàm**

  Function <Tên hàm>(<Các tham số>)

  Hành động với các tham số

  return <Giá trị>

  End Function

- **Gọi hàm**

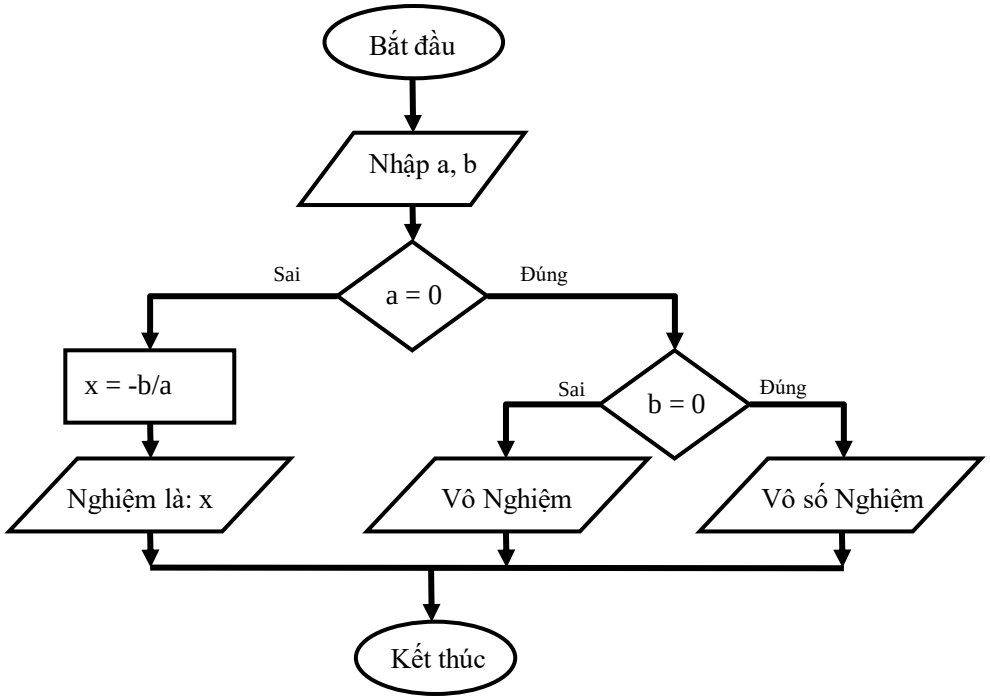
  [**Call**] <Tên hàm>**(**<Các tham số>**)**

Ví dụ: tìm số lớn nhất của dãy

1. **Begin**
2. **Input** N
3. **Input** a_1, a_2,.. a_N
4. Max $\leftarrow a_1$
5. i $\leftarrow$ 2
6. **While** i $\leq$ N **do**
7. **If** a_i > Max **Then**
8. Max $\leftarrow a_i$
9. **End if**
10. i $\leftarrow$i+1
11. **End while**
12. **Output** Max
13. **End**

Ngôn ngữ lập trình

Giải phương trình ax+b=0



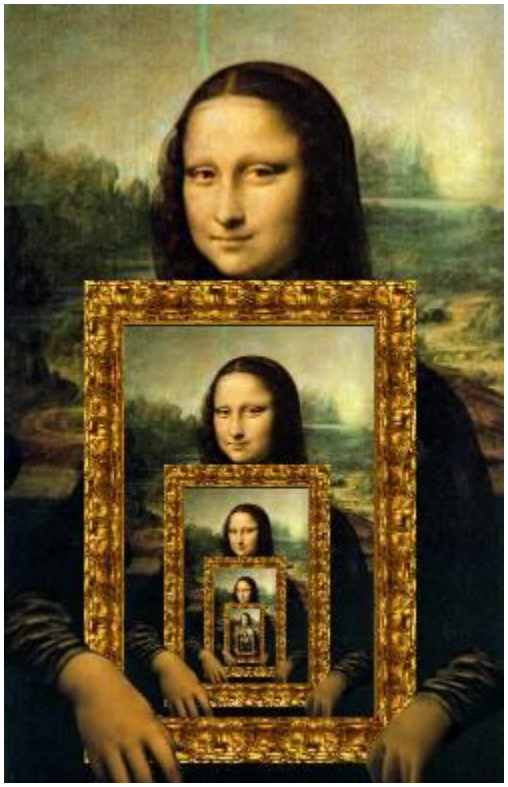

```
#include <stdio.h>
int main(){
 float a,b;
 scanf("%f %f",&a,&b);
 if(a==0)
   if (b==0) printf("Vo so
nghiem");
   else printf("Vo nghiem");
 else printf("Nghiem %f",-b/a);
 return 0;
}
```

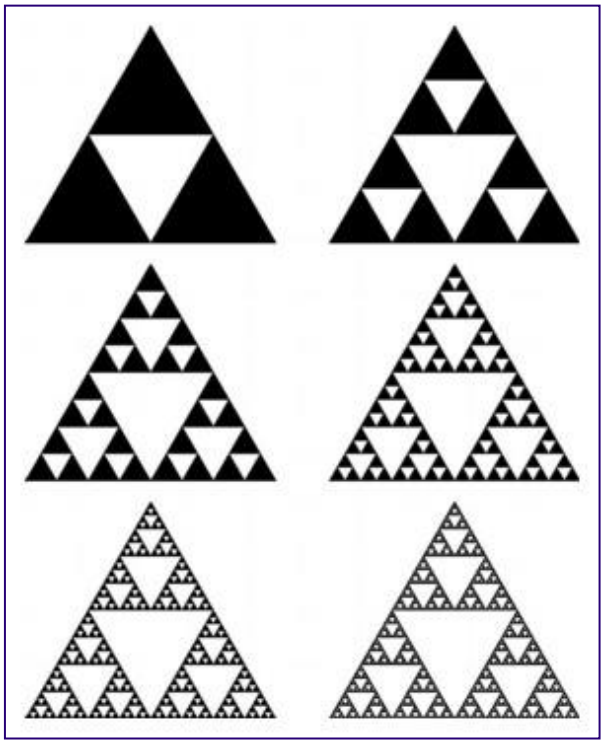
➔ Sẽ học trong nội dung tiếp theo của học phần

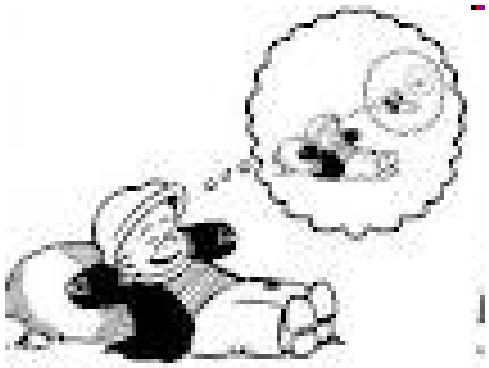
3.2 Thuật toán

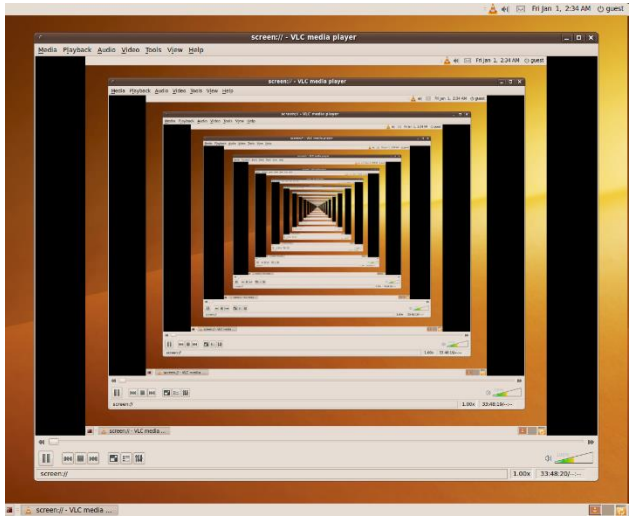
- Khái niệm
- Biểu diễn thuật toán
- Thuật toán đệ quy
- Một số thuật toán ví dụ

Ví dụ đệ quy








screen:// - VLC media player
Media Playback Audio Video Tools View Help

Khái niệm thuật toán đệ quy

- Bài toán có thể được phân tích và đưa tới việc giải một bài toán cùng loại nhưng cấp độ thấp hơn,
  - Độ lớn dữ liệu vào nhỏ hơn
  - Giá trị cần tính toán nhỏ hơn
- **Ví dụ:** Giai thừa của một số tự nhiên n, ký hiệu là n!, được định nghĩa bằng cách quy nạp như sau:
  - 0!=1,
  - n!=(n-1)!*n, với mọi n>0
- **Thuật toán đệ qui:** mỗi bước của thuật toán tái hiện lại chính thuật toán đó với đầu vào nhỏ hơn

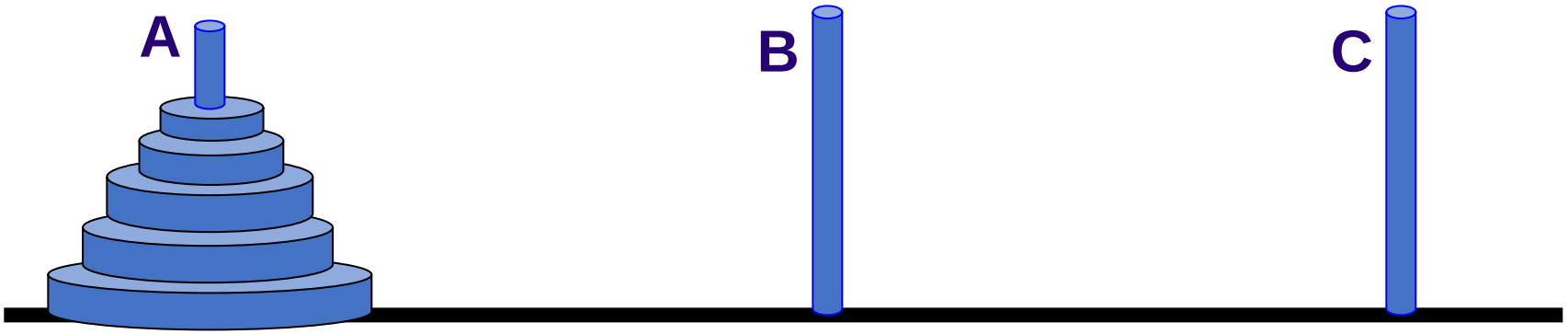
VD: Tính giai thừa của một số tự nhiên

- Input: số tự nhiên n
- Output: F(n)=n!

- Thuật toán:

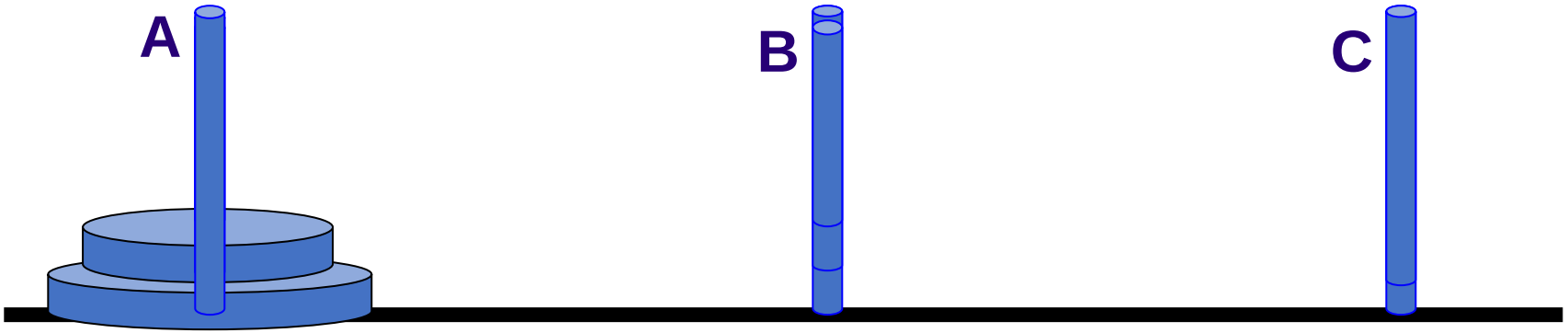
```
Function F(n);
        If n=0 then
                Return 1
        Else
                Return n * F(n-1)
        End If
End Function
```

Ví dụ: Bài toán tháp Hà nội

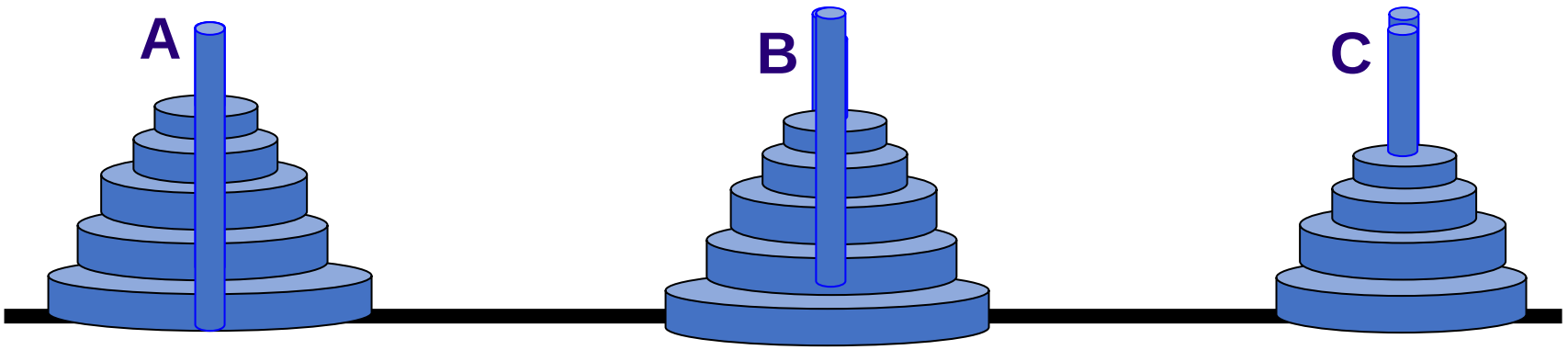


Yêu cầu: Dịch chuyển N đĩa từ cọc A sang B với trung gian là cọc C

N= 2:



Trường hợp tổng quát với N > 2:




```
PROCEDURE ThapHanoi (N,A,B,C)
   IF N= 2 Then Print(A→C, A→B, C→B)
   ELSE
      ThapHanoi(N-1,A,C,B)
      Print(A →B)
      ThapHaNoi(N-1,C,B,A)
   ENDIF
END
```

Lưu ý

- Thuật toán đệ qui gồm 2 phần
  - **Phần cơ sở:** không cần thực hiện lại thuật toán
    - Không có yêu cầu gọi đệ qui.
    - Là điều kiện dừng thuật toán
  - **Phần đệ qui:** có yêu cầu gọi đệ qui
    - Yêu cầu thực hiện lại thuật toán
    - Đặt trong một điều kiện kiểm tra việc gọi đệ qui.
- Đệ qui gọi rất nhiều chương trình con

  → dễ gây tràn bộ nhớ (stack)
- Nếu có thể, nên khử đệ quy (vd: dùng vòng lặp)

3.2 Thuật toán

- Khái niệm
- Biểu diễn thuật toán
- Thuật toán đệ quy
- Một số thuật toán ví dụ

Các bài toán

1. Thuật toán số học
   - Hoán đổ giá trị
   - Số nguyên tố, phân tích ra thừa số nguyên tố…
   - Tìm ước số chung, phân số tối giản
   - Số hoàn hảo
2. Thuật toán trên dãy
   - Vào/ra dãy
   - Tìm Max, Min
   - Sắp xếp
   - Tìm phần tử; Đếm phần tử
   - Tính toán trên các phần tử..
     - Trung bình cộng, tính tổng,…
   - Chèn phần tử/Xóa phần tử (*liên quan tới kiểu mảng*)

Hoán đổi giá trị 2 biến X, Y

Nguyên tắc: Dùng một biến trung gian T

```
Function swap(X,Y)
        T←X
        X ←Y
        Y←T
End Function
```

Kiểm tra số nguyên tố

Begin

1. **Input** P

2. Flag ← FALSE

3. **If** P=1 **Then** **Goto** Bước 6

4. flag ← TRUE (*gán cho cờ hiệu "flag" giá trị true*)

5. **For** k:=2 **to** p-1 **do**

If (k là ước số của P) **Then**

flag ← FALSE;

Goto B6

Endif

End for

6. **If** flag=TRUE **Then Output:** P là số nguyên tố

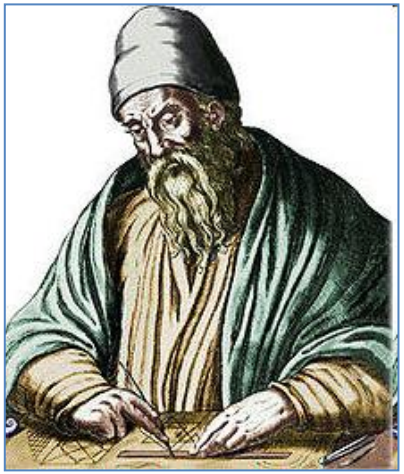
Else Output: P không là số nguyên tố

Endif

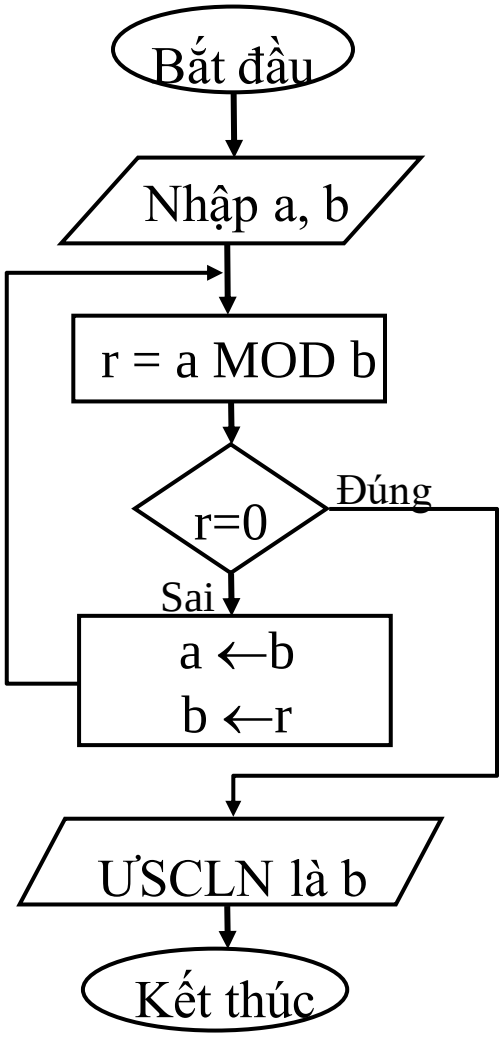
End

Tìm ƯSCLN của 2 số nguyên dương (1/3)

1. Nhập số a
2. Nhập số b
3. Chia a cho b thu được số dư r
4. Nếu r = 0 chuyển sang bước 6
5. Nếu r ≠ 0 gán giá trị b cho a, giá trị r cho b và quay lại bước 3
6. Thông báo kết quả ƯSCLN là b
7. Kết thúc

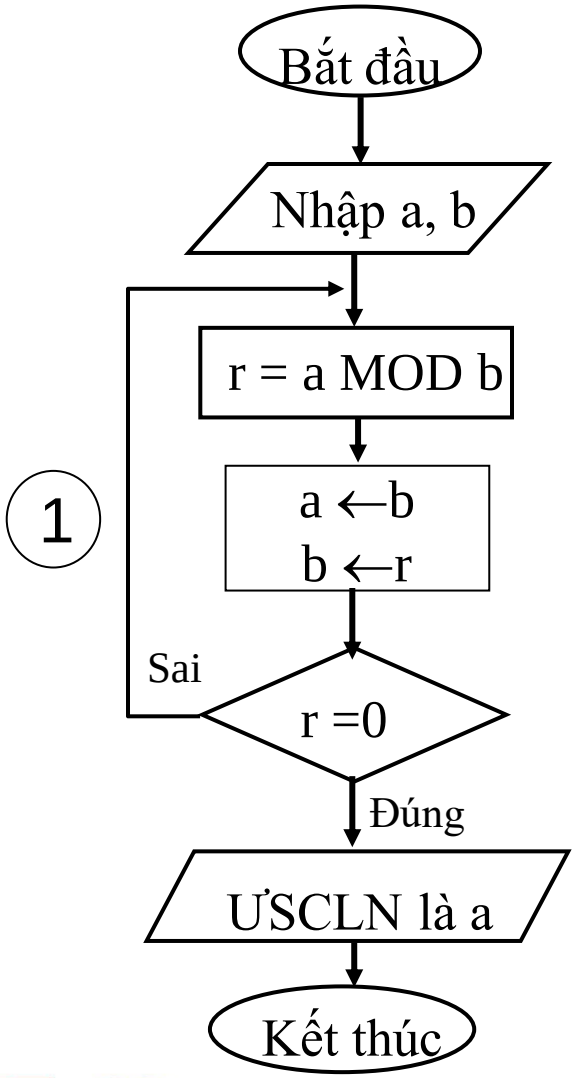


Tìm ƯSCLN của 2 số nguyên dương (2/3)




```
Begin
1.  Input a, b
2.  Repeat
3.      r ← a MOD b
4.      if r=0 then goto 8
5.      a ← b
6.      b ← r
7.  Until r=0
8.  Output USCLN là: b
End
```

Tìm ƯSCLN của 2 số nguyên dương (3/3)

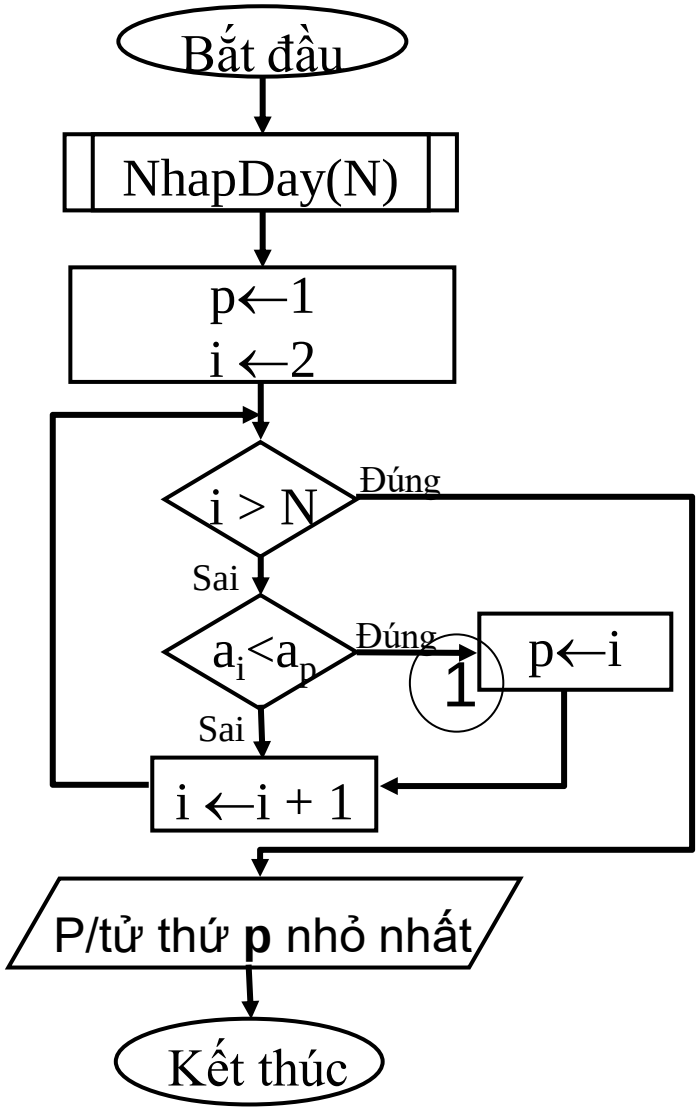


Begin

1. **Input** a, b
2. **Repeat**
3. r ← a MOD b
4. a ← b
5. b ← r
6. **Until** r=0
7. **Output** *USCLN là: a*

End

Tìm phần tử nhỏ nhất trong dãy gồm N số




```
Begin
1.  Nhapday(N)
2.  p←1
3.  i ←2
4.  While i ≤ N do
5.      If a_i < a_p then
6.          p←i
7.      Endif
8.      i ←i+1
9.  Endwhile
10. Output P/tử nhỏ nhất: p
End
```

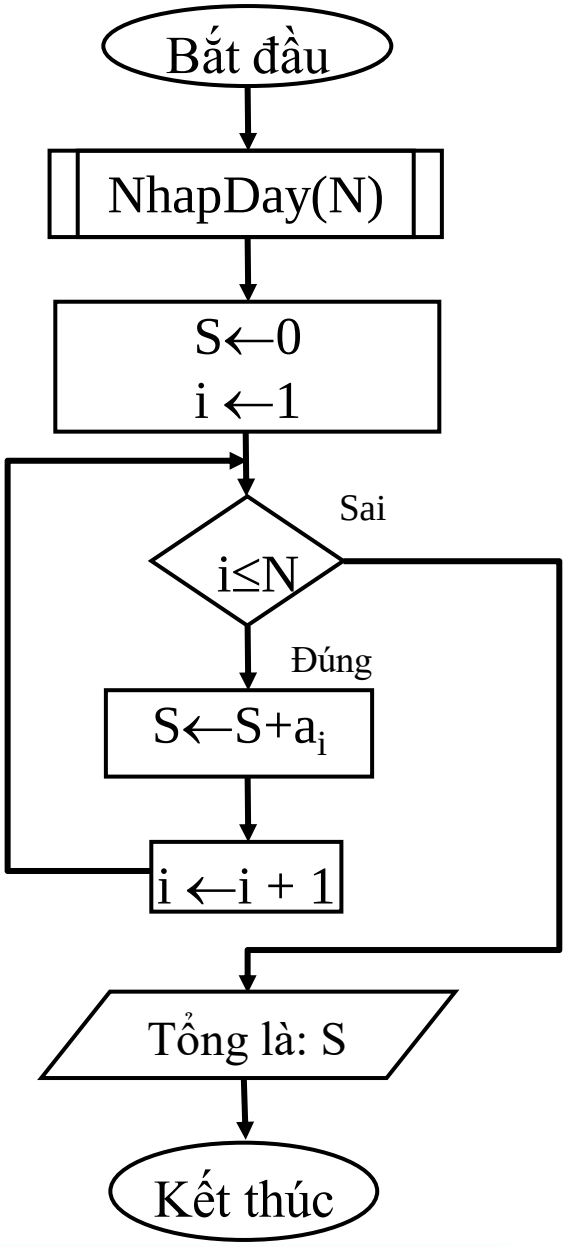
Tìm tổng của dãy gồm N số

Begin
1. NhapDay(N)
2. S← 0
3. i ← 1
4. **While** i ≤ N **do**
   S ← S + a_i
   i ←i+1
5. **End while**
6. **Output:** Tổng là S

End

Begin
1. NhapDay(N)
2. S← 0
3. **For** i ←1 **to** N **do**
   S ← S+ a_i
4. **End For**
5. **Output:** Tổng là **S**

End

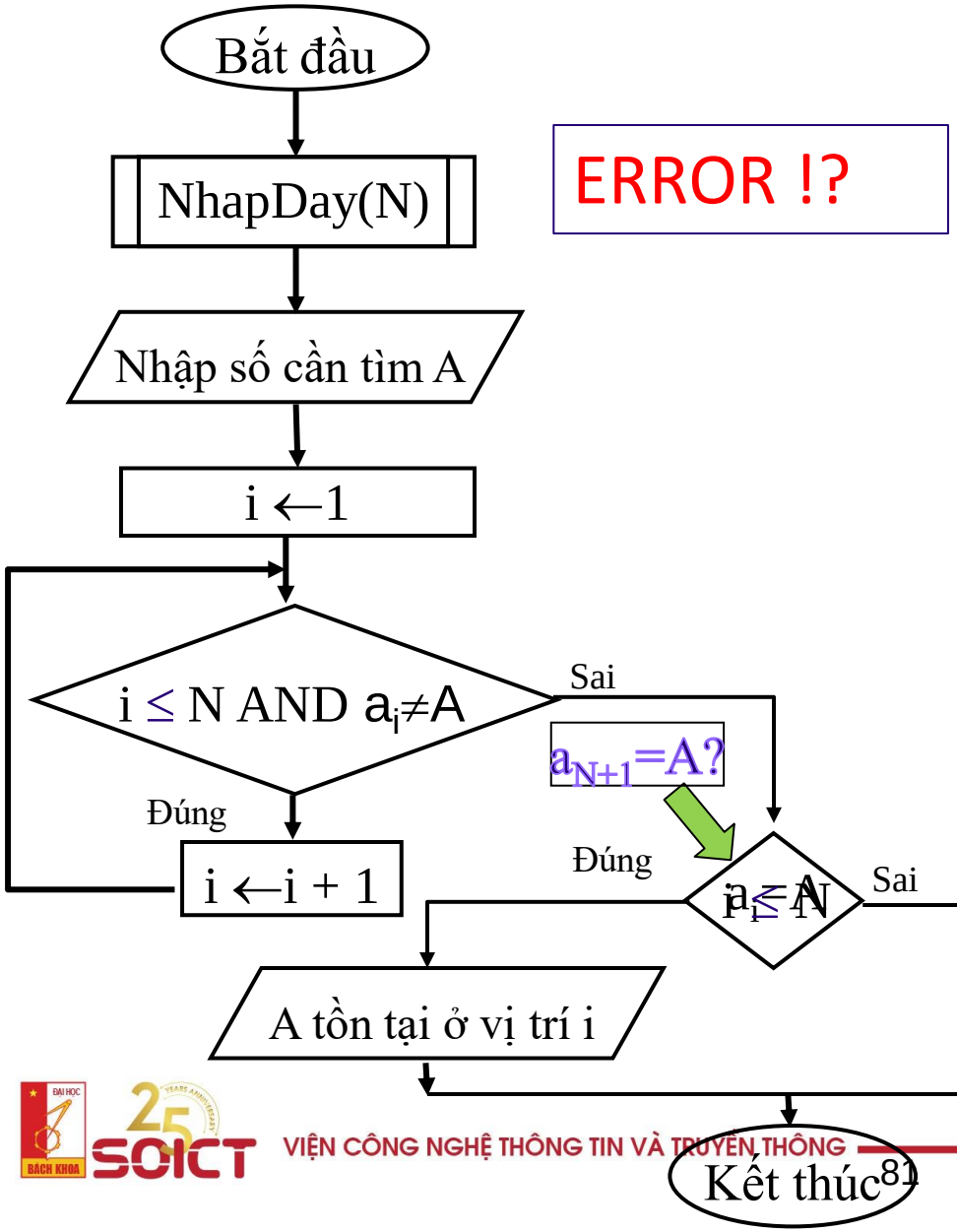


Kiểm tra một phần tử có thuộc một dãy hay không

Begin

1. Nhập N, dãy số a_1, a_2,…a_N và số cần tìm A
2. i←1
3. Nếu i > N sang bước 7
4. Nếu A = a_i sang bước 8
5. i←i+1
6. Quay về bước 3
7. Thông báo: A không thuộc dãy và kết thúc
8. Thông báo A thuộc dãy và kết thúc

Kiểm tra một phần tử có thuộc một dãy hay không



ERROR !?

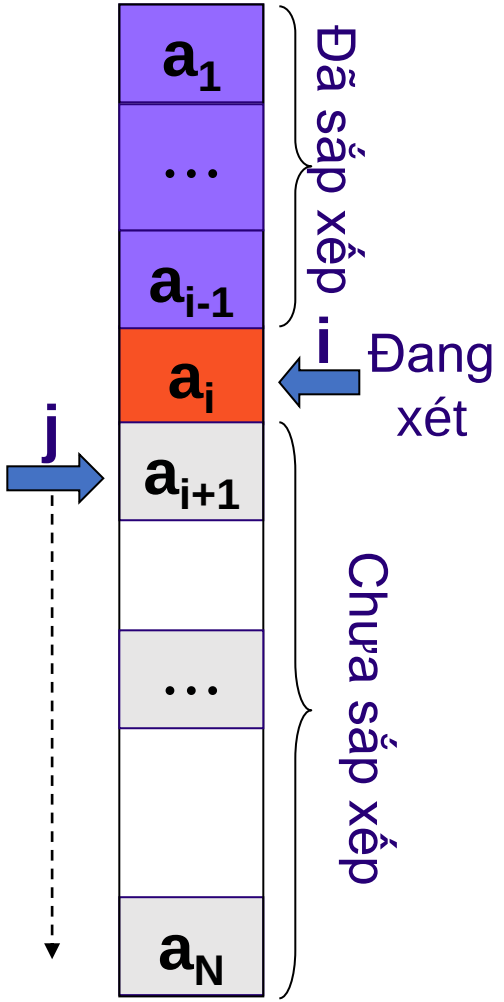
```
Begin
1. NhapDay(N)
2. Input A
3. i ←1
4. While (i≤N AND a_i≠A) Do
       i ← i + 1
5. End While
6. If i ≤ N Then Output A ở vị trí i
7. ELSE Output A không tồn tại
8. End If
End
```

Sắp xếp theo thứ tự tăng của dãy gồm N số

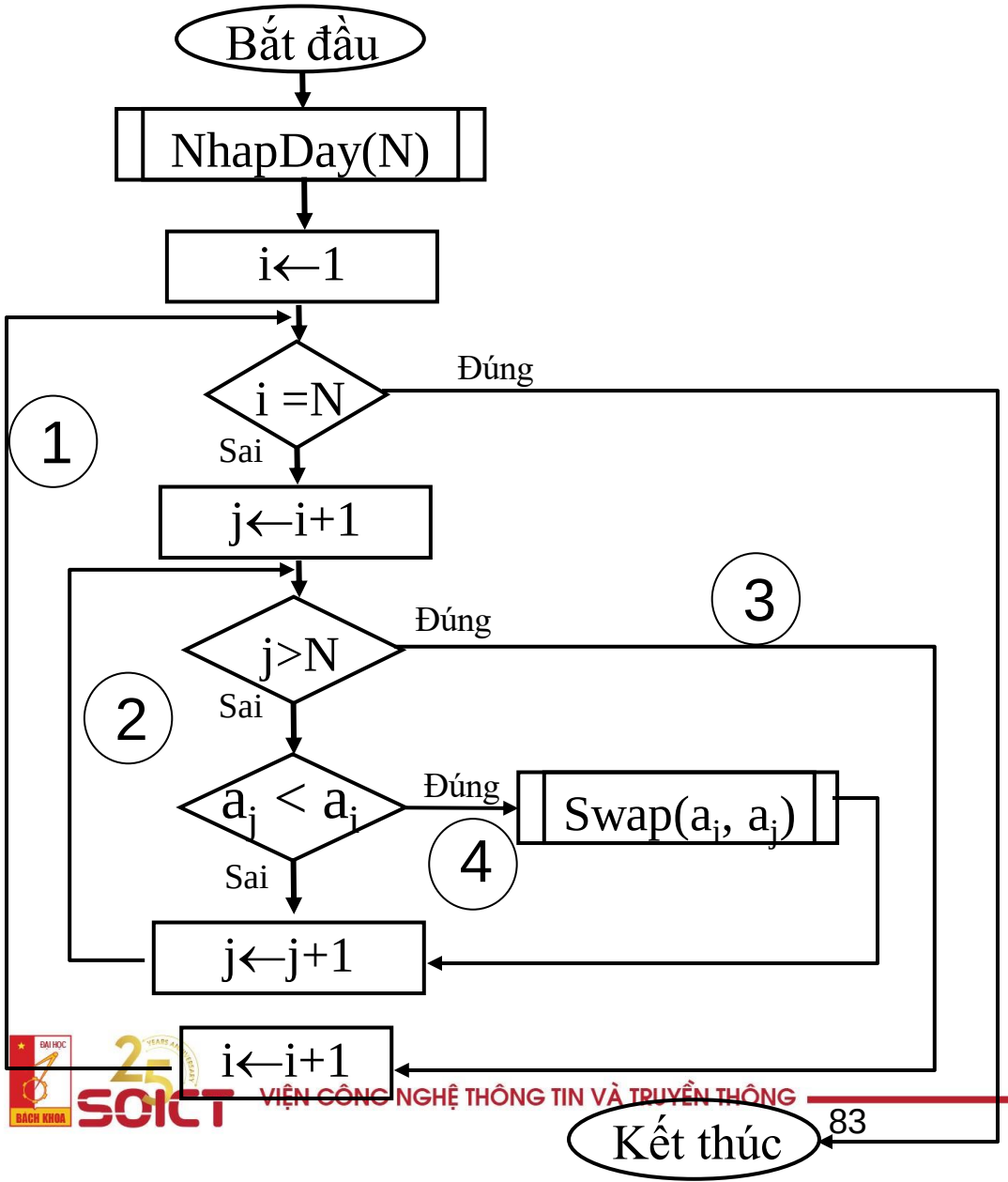
Begin

1. Nhập N và dãy $a_1, a_2, \ldots a_N$
2. $i \leftarrow 1$
3. Nếu i = N, thuật toán kết thúc
4. $j \leftarrow i + 1$
5. Nếu j > N thực hiện bước 9
6. Nếu $a_j < a_i$ thì đổi chỗ a_i và a_j
7. $j \leftarrow j + 1$
8. Quay lại thực hiện bước 5
9. $i \leftarrow i + 1$
10. Quay lại thực hiện bước 3

End



Sắp xếp theo thứ tự tăng của dãy gồm N số




```
Begin
1. NhapDay(N)
2. i ←1
3. While i < N Do
4.   j ←i+1
5.   While j  ≤ N Do
6.       If a_j < a_i Then
             Swap(a_i,a_j)
7.       End If
8.       j ← j+1
9.    End while
10.   i ← i +  1
11.End while
End
```

Bài tập về nhà

Mô tả các thuật toán (sơ đồ khối/mã giả) sau

- Đếm số chẵn của một dãy N số nguyên
- Đưa ra trung bình cộng các số dương của một dãy gồm N số
- Sắp xếp lại dãy N số thực theo nguyên tắc:
  - Các số 0 ở đầu dãy, sau đó là các số âm rồi đến các số dương
- Tìm số lớn nhất trong 1 dãy, và đếm số số có giá trị tìm được