Thạc sĩ Trần Đức Huyên

Phương pháp giải các bài toán trong TIN HỌC

NHÀ XUẤT BẢN GIÁO DỤC-1999

Lời nói đầu

Một trong các mục tiêu khi đưa Tin học vào trường học là nhằm giúp cho học sinh có khả năng phần tích, tổng hợp, trừu tượng hóa; khái quát hóa vấn đề và đăc biệt là phát triển tư duy. Muốn vay ngoài việc day đại trà Tin Học trong nhà trường nhòm dám bảo tính phổ thòng, hướng nghiệp và day nghề ta cán phải tạo diễu kiện cho các học sinh có năng khiếu tin học được phát triển về khả năng lập trình để giải quyết các bài toán. Để góp phần vào việc nâng cao chất lượng dạy và học tin học ở nhà trường phổ thông, đặc biệt là việc bồi dưỡng các học sinh có năng khiếu về Tin Học chúng tôi đã biên soạn cuốn:

Phương pháp giải các bài toán trong Tin học

Cuốn sách nay được viết ra dựa trên kinh nghiệm giảng dạy lớp chuyên Toán và Tin học cùa trường PTTH chuyên Lê Hồng Phong TP. Hồ Chí Minh và các lớp bồi dưỡng đội tuyển học sinh giỏi Tin Học của Quốc gia và của trường Lê Hồng Phong. Chúng tôi cố gắng trình bày có hệ thống một số phương pháp cơ bản để giải các bài toán trên máy tính, đồng thời cung cấp cho học sinh một hệ thống các bài tập phong phú để rèn luyện kĩ năng lặp trình.

Sách gồm ba phần chính:

PHẦN 1 : Giới thiệu các bước tổng quát để giải một bài toán trên máy tính.

PHẦN 2 : Nêu các dạng toán cơ bản trong tin học và các giải thuật cơ bản để giải quyết các dạng toán đó.

PHẦN 3 : Các đề toán tổng hợp để học sinh rèn luyện.

Tất cả các bài tập và đề toán đều có hướng dẫn thuật toán hoặc có lời giải chi tiết bằng ngôn ngữ PASCAL nên rất thích hợp với các học sinh có tinh thần tự học muốn tự nâng cao năng lực giải quyết vấn đề và kỹ năng "Lập trình” của chính mình.

Chúng tôi rất mong nhận được các ý kiến đóng góp xây dựng quý báu của các Thầy, Cô và các em học sinh. Đặc biệt chúng tôi cũng xin chân thành cảm ơn ban Lãnh đạo và ban Biên tập Nhà xuất bản Giáo dục đã tạo điều kiện thuận lợi rất nhiều cho việc xuất bản quyển sách này và nhiều quyển khác trong tủ sách : TIN HỌC TRONG NHÀ TRƯỜNG.

Thành phố Hồ Chí Minh

8/1996

Tác giả

MỤC LỤC

CHƯƠNG I PHƯƠNG PHÁP TỔNG QUÁT ĐỂ GIẢI MỘT BÀI TOÁN BẰNG MÁY TÍNH ... 2
§1. XÁC ĐỊNH BÀI TOÁN ... 2
§2. TÌM CẤU TRÚC DỮ LIỆU ... 6
§3. TÌM THUẬT TOÁN ... 11
§4. LẬP TRÌNH ... 13
§5. CHẠY THỬ, THAY ĐỔI, KIỂM TRA CHƯƠNG TRÌNH ... 22
CHƯƠNG 2 CÁC BÀI TOÁN CƠ BẢN ... 29
§1. ĐỆ QUY ... 29
§ 2. TÌM KIẾM ... 46
§ 3. SẮP XẾP ... 91
§4. QUY HOẠCH ĐỘNG ... 115
§5. ĐỒ THỊ (GRAPH) ... 148
§ 6. TRÒ CHƠI ... 182
§7. HÌNH HỌC ... 214
CHƯƠNG 3 CÁC ĐỀ THI TIN HỌC QUỐC TẾ ... 239
§1. ĐỀ THI TIN HỌC QUỐC TẾ LẦN V ... 239
§2. ĐỀ THI TIN HỌC QUỐC TẾ LẦN VI ... 242
§3. ĐỀ THI TIN HỌC QUỐC TẾ LẦN VII ... 245
§4. ĐỀ THI TIN HỌC QUỐC TẾ LẦN VIII ... 248
§5. GỢI Ý LỜI GIẢI ... 252

CHƯƠNG I PHƯƠNG PHÁP TỔNG QUÁT ĐỂ GIẢI MỘT BÀI TOÁN BẰNG MÁY TÍNH

§1. XÁC ĐỊNH BÀI TOÁN

1. Khái niệm bài toán

Trong quá trình tồn tại và phát triển, mọi cá nhân luôn phải liên tục giải quyết các bài toán. Cuộc 9ống là một chuỗi các bài toán mà ta phải đối đầu để giải quyết.

Theo nhiều nhà nghiên cứu thi mọi bài toán đều có thể diễn đạt theo một sơ đồ chung:

A [đến] B

Trong đó:

* A là giả thiết, điều kiên ban đầu hoặc là cái đã cho, đã có khi bắt đầu giải bài toán.

* B là kết luận, mục tiêu cần đạt hoặc là cái phải tìm, phải làm ra khi kết thúc bài toán.

* [đến] là suy luận, giải pháp cần xác định hoặc là một chuỗi các thao tác cần thực hiên, cần thi hành để có được cái phải tìm B từ cái đã có A.

2. Xác định bài toán

Theo sơ đồ trên thì xác định bài toán có nghĩa là xác định A, B và nếu có thể được thì xác định luôn các thao tác được phép sử dụng để đi từ A đến B. (Điều này rất quan trọng nhưng lại thường được hiểu ngầm).

3. Bài toán trên máy tính

Một bài toán trên máy tính cũng mang đầy đủ các tính chất của một bài toán tổng quát đã nói ở trên nhưng có thể được diễn đạt theo một cách nói khác.

* A : là "input" hay còn gọi là thòng tin vào.

* B : là "Output" hay còn gọi là thông tin ra.

* [đến] : là chương trình tạo ra từ các lệnh cơ bản của máy cho phép biến đổi từ A thành B

4. Những khó khăn

Để xác định một bài toán trên máy tính ta thường gặp hai khó khăn sau:

* Thông báo về A hay B không đầy đủ và không rõ ràng.

* Thông báo về các điều kiên đặt ra cho cách giải (đến) thường không được nêu ra một cách minh bạch.

5. Một số ví dụ

Bài toán 1 (Bài toán 8 quân hậu)

Hãy tìm các cách đặt 8 quân hậu trên một bàn cờ vua sao cho không có quân hậu nào có thể ăn quân hậu khác.

Xác định bài toán

A. Xác định thông tin vào (INPUT):

— Một bàn cờ vua là một bảng hình vuông gồm 8 hàng và 8 cột:

— Quân hậu là quân có thể ăn được bất kì quân nào nằm trên cùng một hàng, cùng một cột hay cùng một đường chéo.

— Ta có tất cả là 8 quàn hậu.

B. Xác định thông tin ra (OUTPUT)

— Các bảng vuông trên đó đã có đãnh dấu vị trí của 8 quân hậu sao cho không có quân hậu nào có thể àn quân hậu khác nghĩa là trên mỗi hàng, mỗi cột, mỗi đường chéo chỉ có thể có một quân hậu. Ví dụ:

— Chỉ ra tất cả các bảng vuông khác nhau thỏa điều kiện của đề bài như bảng vuông ở trên.

Xác định các thao tác chế biến thông tin

— Lần lượt xác định vị trí của một trong 8 quân hậu trên bàn cờ (một ô trong số 64 ô) sao cho:

— Đặt đủ 8 quân hậu.

— Tất cả đều thỏa các điều kiện đã nêu.

Bài toán 2 (Bài toán mã đi tuần)

Cho một bàn cờ có kích thước n X n (n > 3).

Một con mã di chuyển theo luật cờ vua được đặt trong một ô với tọa độ đầu là (x1, y1). Hãy tìm một đường đi với n2 - 1 bước đi sao cho mọi ô trên bàn cờ đều được mã nhảy đến đúng một lần.

Xác định bài toán

A. Xác định thông tin vào (INPUT)

— Một bảng vuông kích thước n x n

— Tọa độ của vị trí thứ nhất của quân má.

— Tám nước đí có thể của quân mã được biểu diễn bởi hình sau:

B. Xác định thòng tin ra (OUTPUT)

— Một bảng vuông trên đó có đãnh dấu vị trí theo thứ tự từ 1 (vị trí đầu) đến n2 (vị trí cuối) của quân mã.

— Từ vị trí k đến vị trí k 4- 1 phải theo đúng luật đi của quân mã (nghĩa là chỉ được sử dụng 1 trong 8 nước đi mô tả ở trên).

— Ví dụ : (n = 5, X1 = 3, y1 = 3)

23	10	15	4	25
16	5	24	9	14
11	22	1	18	3
6	17	20	13	8
21	12	7	2	19

Xác định các thao tác chế biến thông tin:

- Lần lượt xác định các vị trí của quân mã (từ 2 đến n2) sao cho quân mã:

• Di chuyển đúng luật,

• Mỗi ô chỉ được đi một lần.

Bài toán 3

Cho một dãy số nguyên dương

a1, a2, a3,…, a_n.

Hãy tìm từ dãy trên một dãy con (không nhất thiết phải liên tiếp) tăng và có độ dài là dài nhất.

Xác định bài toán

A. Xác định thông tin vào (INPUT)

— Một dãy số nguyên dương

a1, a2, a3,… a_n

— Mỗi số được xác định bởi 2 yếu tố là giá trị của số và vị trí (số thứ tự) của số đó trong dãy.

B. Xác định thông tin ra (OUTPUT)

— Một dãy lấy ra từ dãy đã cho:

ai_1, ai_2, ai_3,…, ai_k

(Nghĩa là dãy con thu được sau khi gạch bỏ một số phần từ của dãy đã cho).

— Dãy con này phải có 2 tính chất:

• Tăng

$ai_1 < ai_2 < ai_3$,…, < ai_k

• Có độ dài là dài nhất trong các dãy có thể lấy được.

Xác định các thao tác chế biến thông tin:

— Lần lượt duyệt các phần tử của dãy đã cho.

— Phải quyết định xem phải loại bỏ phần tử nào để dãy còn lại là tăng và dài nhất.

Bài toán 4

Cho 2 số tự nhiên a, b. Tìm ước số chung lớn nhất của chủng.

Xác định bài toán

A. Xác đinh thông tin vào (INPUT):

— Hai số tự nhiên a, b.

B. Xác định thòng tin ra (OUTPUT):

— Số tự nhiên d thỏa

• d là ước của a và d là ước của b

• d lớn nhất trong tập các ước chung của a và b.

Xác đinh các thao tóc chế biến thờnỉi tin

— Xây dựng một dãy hữu hạn các thao tác cho phép tính được d từ a và b.

Ví dụ :

a = 16,

b = 24,

d = 8

6. Một số nhận xét quan trọng

Qua các ví dụ trên ta thấy một số nhận xét quan trọng sau đây:

— Việc xác định bài toán là rất quan trọng nó ảnh HƯỚNG rất lớn đến cách thức và chất lượng của viêc giải quyết bài toán.

— Một bài toán cho dù dược diễn đạt bằng thông báo chính xác đến đâu đi chăng nữa cũng phải giả định là phần lớn thông tin về A và B tiềm ẩn trong đầu người giải. Thông báo về A hoặc B chỉ là biểu tượng gợi nhớ đến các thông tin tiềm ẩn đó.

— Bước đầu tiên để xác định một bài toán là phải phát biểu lại bài toán một cách chính xác theo ngòn ngữ của riêng mình vì đó chính là cách ta tiếp cận bài toán, hiểu bài toán.

— Bước kế tiếp là tìm hiểu các thòng tin Input A và thông tin Output B, tìm các mối liên hệ giữa chúng.

— Nên xét một vài trường hợp cụ thể để thông qua đó hiểu được cả bài toán và cũng thông qua đó thấy rõ được các thao tác cần phải tiến hành. Thực tế cho thấy có những bài toán trong tin học chỉ có thể mô tả được thông qua các ví dụ.

§2. TÌM CẤU TRÚC DỮ LIỆU

BIỂU DIỄN BÀI TOÁN

1. Khái niệm ban đầu

Máy tính điện tử được phát minh như một thiết bị nhằm làm dễ dàng và tiến hành nhanh các tính toán phức tạp và lớn. Trong nhiều ứng dụng, khả năng lưu trữ và truy cập lượng thông tin lớn giữ vai trò quan trọng và được xem như là đặc trưng chính của nó, và khả nâng tính toán trở thành ít quan trọng trong nhiều trường hợp. Trong thực tế, thông tin cần xử lí là trừu tượng. Thông tin cung cấp cho máy tính điện tử (MTĐT) gồm một tập hợp dữ liệu về bối cảnh thực đã được mã hóa. Nói rõ hơn, tập hợp này được xem như thích đãng cho vấn đề định sẵn. Từ đó có thể đưa vào máy tính và tính ra kết quả cần tìm.

Khi giải một bài toán ta cần phải định nghĩa tập hợp dữ liệu để biểu diễn tình trạng cụ thể. Việc lựa chọn này tùy thuộc vào vấn đề phải giải quyết. Sau đó là việc lựa chọn cách biểu diễn thông tin này. Thông thường việc lựa chọn này phải tùy thuộc vào các thao tác cần thực hiện trên dữ liệu.

Ví dụ:

— Nếu chỉ cần thao tác cộng (+) thì cách tốt nhất để biểu diễn một số n là n que hoặc dùng số La Mã. Qui tắc cộng với sự biểu diễn này là khá đơn giản và tự nhiên trong khi đó phép biểu diễn dùng số Ả Rập đòi hỏi các quy tắc ít hiển nhiên hơn.

— Tuy nhiên, trường hợp trên bị đảo ngược khi ta xét đến việc cộng các số lớn hoặc khi xét phép nhân hoặc phép chia. Việc phân tích các thao tác trên thành các thao tác đơn giản được thực hiện dễ dàng trong phép biểu diễn hằng số Ả Rập mà nguyên tắc giá trị dựa trên vị trí các chữ số.

- Mọi người đều biết là MTĐT biểu diễn mọi dữ liệu bằng các chữ số nhị phân (bit). Phép biểu diễn này ít thích hợp cho con người vì cần khá nhiều bit, song nó lại khá thích hợp cho các mạch điện tử vì hai trị 0 và 1 có thể được biể diễn một cách dễ dàng và tin cậy được với sự có hay vắng mặt cảu dòng điện, từ trường hoặc điện tích.

2. Các câu trúc dữ liệu thường dùng

- Trong ngôn ngữ PASCAL ta có sơ đồ các kiểu dữ liệu như sau:

DATA TYPE gồm có SIMPLE TYPE và STRUCTURE TYPE.

SIMPLE TYPE gồm BASIC TYPE (BOOLEAN, INTEGER, REAL, CHAR) và USER DEFINED TYPE (SUB_RANGE, ENUMERATED)

STRUCTURE TYPE gồm ARRAY, SET, RECORD, STRING, FILE

BOOLEAN : Kiểu logic chỉ có thể nhận một trong hai giá trị TRUE hoặc FALSE

INTEGER : Kiểu số nguyên (-32768 .. 32767)

REAL : Kiểu số thực ($2.9 \text{ X } 10^{-39}$..$1.7\text{x } 10^{38}$)

CHAR : Kiểu kí tự (chữ viết, chữ số,..)

SUB RANGE : Kiểu miền con cùa một kiểu sơ cấp. Ví dụ:

```
TYPE
Chu_hoa = 'A' .. 'Z' ;
Chu_thuong = 'a' ..'z' ;
Tuoi_tho = 0 .. 200 ;
```

* ENUMERATED : Kiểu liệt kê được định nghĩa bằng cách liệt kê tất cả các phần tử có thể có.

Ví dụ :

```
TYPE
Thu = (ChuNhat, ThuHai, ThuBa, ThuTu, ThuNam, ThuSau, ThuBay);
Color = (Red, Blue, Green, White, Black);
Trui cay = (Cam, Xoai, Man, Nho);
```

* * ARRAY : Kiểu mảng gồm một tập hợp các phần tử cùng thuộc một kiểu dữ liệu cơ sở được xác định bởi chỉ số.

Ví dụ:

```
TYPE
A = ARRAY [Kiểu chỉ số] OF [Kiểu cơ sở] ;
```

B = ARRAY [1.. 100] OF Integer ; C = ARRAY [1.. 10, 1.. 20] OF Boolean ;

* RECORD : Kiểu bản ghi gôm một tập hợp các phần tử thuộc các kiểu dữ liêu khác nhau.

Ví dụ:

TYPE

Hocsinh = RECORD

HoTen : String [30] ;

Lop : 1.. 12 ;

Truong : String [30] ;

DiemTB : Real ;

Ketqua : Boolean ;

END ;

* SET : Kiểu tập hợp gồm một số các đối tượng có cùng kiểu cơ sở.

Ví dụ:

TYPE

Chucai = SET OF CHAR ;

Chuso = SET OF 0.. 9 ;

* STRING : Kiểu chuối gồm một dãy các kí tự.

Ví dụ:

TYPE

Tên chuỗi = STRING [Chiều dài chuỗi].

* FILE : Kiểu Tệp gồm một tập hợp các dữ liệu có liên quan với nhau và có cùng kiểu được nhóm lại thành một đãy và được lưu trữ trên đĩa.

Ví dụ:

TYPE

Tep = FILE OF integer ;

Bai = TEXT ;

3. Các lưu ý khi chọn cấu trúc dứ liệu

Khi lựa chọn cấu trúc dử liêu để biểu diễn một bài toán ta nên dựa vào các tiêu chuẩn sau đày:

— Cấu trúc dữ liệu phải biểu diễn được đầy đủ các thông tin nhập và xuất của bài toán.

— Cấu trúc dữ liệu phải phù hợp với các thao tác của thuật toán mà ta lựa chọn để giải quyết bài toán.

— Cấu trúc dữ liệu phải phù hợp với điều kiện cho phép của ngôn ngữ lập trình và MTĐT đang sử dụng.

4. Các ví dụ

Bài toán 1 (Bài toán 8 quân hậu)

— Bàn cờ là một bảng vuông 8x8 , nên rõ ràng cấu trúc mảng (ARRAY) là thích hợp để biểu diễn.

— Theo luật cờ vua, một quân hậu ăn được mọi quân khác nằm cùng hàng, hoặc cùng cột hoặc trên cùng đường chéo trên bàn cờ. Vậy ta suy ra mỗi cột có thể chứa một và chỉ một quân hậu. Do đó để đơn giản ta ký hiệu quân hậu ở cột i là i. Như vậy tham biến i trở thành chỉ số cột và việc lựa chọn được tiến hành trên tám giá trị của chỉ số hàng j.

— Để tìm dữ liệu biểu diễn tám quân hậu trên bàn cờ, cách chọn thoạt tiên là dùng mảng vuông để biểu diễn bàn cờ, nhưng xem kỹ thấy cách biểu diễn đó dẫn tới các thao tác cồng kềnh trong việc thử quyền sử dụng các quân hậu ở các vị trí. Điều đó hết sức không hay vì thao tác trên lại phải thực hiện nhiều lần. Do đó ta sẽ chọn cách biểu diễn sao cho thao tác trên dễ bao nhiêu hay bấy nhiêu. Cách tốt nhất là biểu diễn các thông tin thực sự nổi bật và được sử dụng một cách càng trực tiếp càng tốt. Trong trường hợp của bài toán này thì đó không phải là vị trí các quân hậu mà là phải chăng đã có một quân hậu trên mỗi hàng và các đường chéo (Ta đã biết rằng đúng một quân hậu đã được đặt trên mỗi cột k, với 1 < k < i). Điều đó dẫn tới cách chọn các biến như sau:

Var x : array [1 .. 8] of Integer;

a : array [1 .. 8] of Boolean;

b : array [b1 .. b2] of Boolean;

c : array [c1 .. c2] of Boolean;

Trong đó

x[i] biểu diễn vị trí của quân hậu trong cột thứ i;

a|j] = True có nghĩa là không có quân hậu nào nầm trên hàngj;

b[k] = True có nghĩa là không có quân hậu nào nằm trên đường chéo thứ k;

c[k] = True có nghĩa là không có quân hậu nào nằm trên đường chéo thứ k;

Các chỉ số của b và C đều tính toán được, do đó việc chọn các giới hạn chỉ số b1, b2, c1, c2 cũng được quyết định theo; ta lưu ý rằng trên một đường chéo thì mọi ô có cùng tổng số các tọa độ i + j, và trên một đường chéo thì hiệu số các tọa độ i — j là không đổi. Cụ thể là:

b1 = 2, b8 = 16

c1 = -7, c8 = 7

Với cách chọn cấu trúc dữ liệu biểu diễn bài toán như trên thì :

* Câu lệnh đặt quân hậu sẽ là như sau:

X [i] := j;

A [j] := False;

B [i + j] := False;

C [i — j] := False;

* Câu lệnh cất quân hậu sẽ là:

a [j] := True;

b [i + j] := True;

c [i — j] := True;

* Điều kiện an toàn của ô (i, j) có thể đặt quản hậu được biểu diễn bởi biểu thức logic:

a [j] and b [i + j] and C [i — j] .

Bài toán 2 (Đài toán mã di tuẩn)

— Ta có thể biểu diễn bàn cờ bằng một ma trận gọi là dd. Đồng thời ta đưa ra một kiểu dữ liệu để chỉ các giá trị chỉ số.

Type chỉ số = 1.. n;

Var dd : array [chỉ số, chỉ 8ố] of integer;

— Ta quyết định chọn biểu diễn của mỗi ô trên bàn cờ bởi một số nguyên chứ không phai một giá trị Boolean để chỉ ô bị chiếm, vi rằng ta còn muốn giữ lại dấu vết của các ô bị chiếm theo tuần tự. Qui ước như sau là một cách chọn lựa hiển nhiên:

dd[x, y] := 0 : ô (x, y) chưa được thăm đến.

dd[x, y] := i : ô (x, y) đã được thăm ờ nước thứ ỉ (1 < ỉ < n2).

— Để tìm cấu trúc dữ liệu để biểu diễn 8 nước đi có thể của con mã ta phân tích như sau:

Cho một cặp tọa độ xuất phát (x, y) thì có 8 chỗ lần lượt có thể lấy làm tọa độ mã nhảy đến (u, v). Chúng được đãnh số từ 1 đến 8 như trong hình. Một phương pháp đơn giản đế thu được u, V từ X, y là cộng thêm các chênh lệch về tọa độ chừa sẵn trong một mảng các cặp chênh lệch, hoặc trong hai mảng của từng chênh lệch. Giả sử các mảng đó là a và b ta gán giá trị như sau:

a m := 2 ;b [1] := 1 ;

a [2] := 1 ; b [2] := 2 ;

a [3] := - 1 ; b [3] := 2 ;

a [4] := - 2 ; b [4] := 1 ;

a [5] := - 5 ; b [5] := - 1 ;

a [6] := 1 ; b [6] := - 2 ;

a [71 := 1 ; b [7] := - 2 ;

a [8] := 2 ; b [8] := - 1 ;

— Còn một chi tiết nữa không thể bỏ qua đó là : Một biến [u, v] chỉ tồn tại được nếu cả hai u và V đều nằm trong các giới hạn 1.. n Vậy ta nên xây dựng tập S : = [1.. n] và điều kiện đi được của con mã là

```
For j := 1 to 8 do
 Begin
u := X + a Ịj] ;
V := y + b UI ;
If (u in S) and (v in S) and dd [u, v] = 0
then
Begin
dd [u, v] := i ;
X := u ; y := V ;
Try (i + 1) ;
X := u - a Ịj] ;
y := V - b Ịj] ;
dd [u, v] := 0 ;
end ;
end ;
```

§3. TÌM THUẬT TOÁN

1. Khái niệm thuật toán

Thuật toán là một hệ thống chặt chẽ và rõ ràng các qui tắc nhằm xác định một dãy các thao tác trên những đối tượng, sao cho sau một số hữu hạn bước thực hiện các thao tác, ta đạt được mục tiêu định trước.

2. Các đặc trưng của thuật toán

a/ Tính xác định

Ở mỗi bước của thuật toán, các thao tác phải hết sức rõ ràng. Không thể gây nên sự nhập nhằng, lẫn lộn, tùy tiện. Nói cách khác là trong cùng một điều kiện, hai bộ xử lí cùng thực hiện một bước của thuật toán thì phải cho cùng một kết quả.

b/ Tính hữu hạn dừng

Một thuật toán bao giờ cũng phải dừng lại sau một số hữu hạn bước.

c/ Tính đúng đắn

Sau khi thực hiện tất cả các lệnh của thuật toán ta phải được kết quả mong muốn, kết quả đó thường được xác định theo định nghĩa có trước.

d/ Tính phổ dụng

Thuật toán có thể giải bất kì bài toán nào trong cùng một lớp các bài toán, có nghĩa là thuật toán có thể làm việc với các dữ liệu khác nhau, trong một miền xác định và luôn dẫn đến kết quả mong muốn.

d/ Tính có đại lượng vào và ra

Khi bắt đầu, một thuật toán bao giờ cũng nhận các đại lượng vào mà ta thường gọi là dữ liệu vào, các dữ liệu vào thường lấy từ một tập xác định cho trước.

Sau khi kết thúc, một thuật toán bao giờ cũng cho ta một số đại lượng ra tùy theo chức năng mà thuật toán đảm nhiệm, chúng thường được gọi là dữ liệu ra.

f/ Tính hiệu quả

— Tính hiệu quả của một thuật toán được đãnh giá dựa trên các tiêu chuẩn sau:

* Dung lượng bộ nhớ cần có.

* Số các phép tính cần thực hiện.

* Thời gian cần thiết để chạy.

* Có dễ hiểu đối với con người không.

* Có dễ cài đặt trên máy không.

3. Mở rộng khái niệm thuật toán

Để có thể giải các bài toán bàng máy tính chúng ta thường phải có một quan niệm rộng rãi hơn về thuật toán. Cụ thể là lưu ý đến các đặc điểm sau:

a/ Không cần xác định toàn bộ lời giải, các thao tác theo từng bước một cách chính xác, đơn vị và rõ ràng. Thay vào đó chỉ cần chỉ ra một cách chuyển từ một bước giải i tới bước giải kế tiếp i + 1, và tìm cách cốt nhỏ bài toán thành các bài toán con, đó chính là thuật toán ĐỆ QUY rất quan trọng để giải các bài toán tổng quát.

b/ Có nhiều bài toán không có cách giải đúng, hoặc cách giải đúng không thể chấp nhận được do hạn chế về thời gian chạy và kích thước bộ nhớ. Nhưng nếu ta chấp nhận kết quả gần đúng thì có thể tồn tại nhiều cách giải đỡ phức tạp và có hiệu quả hơn, đó chính là các thuật toán Heuristic để giải các bài toán gần đúng.

Ví dụ 1 (Bài toán Euclide)

Tìm ước số chung lớn nhất của hai số nguyên dương m và n. Ta cổ thuật toán Euclide dược diễn tả bằng lưu đồ như sau:

Ví dụ 2 (Bài toán sàng Eratosthenes)

Tim tất cả các số nguyên tố trong các số nguyên từ 1 đến N.

(N lớn hơn hoặc bằng 3)

Ta có thuật toán Eratosthenes như sau:

Ví dụ 3 (Bài toán 8 quân hậu)

Hãu tìm cách đặt 8 quân hậu trên một bàn cờ vua sao cho không có quân hậy nào có thể ăn quân hậu khác.

Ta dùng thuật toán Đệ qui có quay lui dựa trên phương pháp thử và sai như sau:

Ví dụ 4 (Bài toán mã đi tuần)

Cho một bàn cờ có kích thước n x n. Một con mã di chuyển theo luật cờ vua được đặt trong một ô với tọa độ đầu là (x1y1). Hãy tìm một dường đi với n^2-1 bước đi sao cho mọi ô trên bàn cờ đề được mã nhảy đến đúng một lần.

Ta dùng thuật toán Đệ qui có quay lui dựa trên phương pháp thử và sai như sau:

Ví dụ 5

Cho 2 dãy số nguyên dương a1, a2,…, an

b1, b2,…, bm

sao cho a1 + a2 + ... + an = b1 + b2 + ... + bm

Hãy lập một bảng số a [1.. n, 1.. m] gồm các số nguyên dương thỏa mãn:

— Tổng các số ở hàng i bhng ai

— Tổng các số ở cột j bàng aj

Ta sử dụng thuật toán Heuristic "GREEDY" (Tham ăn) như sau:

§4. LẬP TRÌNH

1. Khái niệm

Lập trình là dùng một ngôn ngứ máy tính cụ thể nồo đó (ví dụ I 'itHcal) để diễn tả thuật toán, cấu trúc dữ liệu thành các câu lệnh để mriy tính có thể thực hiện được và giải quyết đúng bài toán mà người l(q» trình mong muốn.

2. Kĩ năng lập trình

Kĩ năng lập tringh là kĩ năng cài đặt thành công các thuật toán bằng một ngôn ngữ lập trình. Đã gọi là kĩ năng thì chỉ có thể có được thông qua rèn luyện tích cực.

Kinh nghiệm cho thấy một thuật toán hay nhưng do cài đặt vụng về, lộn xộn khi chạy trên máy tính có thể cho kết quả rất tồi tệ.

3. Phát triển chương trình bằng cách tinh chế từng bước

Tinh chế từng bước là một phương pháp khoa học, có hệ thống giúp ta phân tích các thuật toán và cấu trúc dữ liệu từ đó viết thành chương trình.

Muốn lập trình giỏi không phải chỉ cần nắm vững ngôn ngữ lập trình là đủ.

Vấn đề cốt yếu là biết phương pháp phát triển dần dần để chuyển các ý tưởng ra thành chương trình hoàn chỉnh.

4. Phương pháp tinh chế từng bước

Ban đầu chương trình được viết bằng những lời tự nhiên (Tiếng Việt chẳng hạn) thể hiện sự phân tích tổng thể của người lập trình.

Ở từng bước sau, mỗi câu lời được phân tích ra chi tiết hơn bằng nhiều câu lời khác tương ứng với sự phân tích một công việc thành những công việc nhỏ hơn. Mỗi câu lời đó là một sự đặc tả công việc.

— Ta nói ở mỗi bước ta đã tinh chế những câu lời đó. Sự tinh chế được hướng về phía ngôn ngữ lập trình mà ta sẽ dùng. Nghĩa là, càng ở những bước sau, các câu lời tự nhiên (tiếng Việt) càng được thay nhiều bằng các câu lời của ngôn ngữ lập trình (ví dụ Pascal)

— Một câu lời tự nhiên nếu đơn giản thì có thể thay bằng một vài phát biểu, còn nếu nó tỏ ra phức tạp thì có thể coi nó là một thủ tục, và ta tiếp tục tinh chế nó.

— Trong quá trình tinh chế đó, ta cần phải đưa ra những biểu diễn dữ liệu, vì dữ liệu là nguyên liệu của máy tính. Như vậy cùng với sự tinh chế các công việc, dữ liệu cũng được tinh chế, phân rả, hay cấu trúc hóa. Điều này có nghĩa là sự tinh chế các đặc tả chương trình và dữ liệu là song song.

— Phương pháp tinh chế từng bước là một thể hiện của tư duy giải quyết vấn đế từ trên xuống trong đó sự phát triển của các bước là hướng về phía ngôn ngữ lập trình được dùng. Đãy của sự đi xuống trong hoạt động phân tích là các phát triển và các mô tả dữ liệu viết bằng ngôn ngữ lập trình.

— Hiểu được phương pháp tinh chế từng bước sẽ giúp người lập trình có được một định hướng thể hiện sự ngăn nắp trên tờ giấy nháp của mình, tránh khỏi việc phải mò mẫm, thử đi thử lại nhiều lần các chương trình mang tính trực giác.

5. Các ví dụ

Ví dụ 1 (Bài toán Euclide)

Tìm ước số chung lớn nhất của hai số nguyên dương m và n.

Theo thuật toán Euclide đã trình bày trong §3 ta tinh chế chương trình như sau:

Bắt đầu

Nhập m, n, lưu m vào a, lưu n vào b

While m khác n

If M > n then thay m bởi m — n

else thay n bởi n — m

Xuất UCLN của a, b là m

Kết thúc

Chương trình có thể được viết hoàn chỉnh như sau

Program UCLN ;

```
Var m, n : integer ; a, b : integer ;
Begin
Writeln (’Xin cho biet m, n’) ;
Readln (m, n) ; a := m ; b := n ;
While m < > n do
if M > n then m := m — n
else n := n — m;
Writeln (’UCLN cua’ , a, b, ’la’ , m) ;
Readln ;
End.
```

Ví dụ 2 (Bài toán Sàng Eratosthenes)

Tìm tất cả các số nguyên tố trong các số nguyên từ 1 đến N (N lớn hơn hoặc bằng 2).

Theo thuật toán Eratosthenes trong §3 ta có thể tinh chế chương trình như sau.

TINH CHẾ LẦN I

- Lấy 2 tập

NT = [] (Để chứa các số nguyên tố tìm được)

S = [2,.. N] (Tập các số cần xét)

- Tìm số đầu tiên trong s đưa vào NT.

- Loại bỏ khỏi s các bội số của số nguyên tố vừa tìm được (sàng lọc)

- Số đầu tiên còn lại của s là số nguyhn tft Tiếp tục quá trình cho đến khi s = [ ]

- Xuất NT

TINH CHẾ LẦN 2

```
Bắt đầu
NT := [ ] ;
S := [2... N] ;
Repeat
Tìm số đầu tiên trong S
NT := NT + [S0]
Loại khỏi S các bội của S0
Until S = [ ]
Xuất NT ;
Kết thúc.
```

TINH CHẾ LẦN 3 (Chương trình hoàn chỉnh)

```
Program ERATOSTHENES ;
Const
N = 100 ;
Type
Nguyen = 1.. N ;
Var
NT, S:Set of Nguyen ;
S0 , I : Integer ;
Begin
NT := [ ] ;
S := [2.. N] ;
S0 := 2 ;
Repeat
While n0t (S0 in S) do
S0 :=S0+1 ;
NT := NT + [S0] ;
I := SO ;
While I =< N do
Begin
S := S – [I]
I := I + S0 ;
End ;
Until S = [];
For I := 1 to N do
If I in NT then writeln (I) ;
Readln ;
End
```

Rõ ràng là cấu trúc dữ liệu tập hợp Set of Nguyen tuy dễ hiểu nhưng rất cồng kềnh và làm máy chạy rất chậm chạp, ta có thể dùng mảng Boolean linh hoạt hơn như sau:

TINH CHẾ LẦN 4 (Dùng mảng Boolean)

```
Program ERATOSTHENES Const N = 1000 ;
Var a : array [1.. n] of Boolean ;
i, j : integer ;
Begin
a [1] := False ;
```

```
For i := 2 to N do a [i] := true ;
For i := 2 to (N div 2) do For j : = 2 to (N div i) do
a [i * j] := False ;
For i := 1 to N do
If a [i] then write (i : 4) ;
Writeln ;
End.
```

Trong ngôn ngữ PASCAL, nếu dừng mảng Bôlean thì ta bị giới hạn N nhỏ hơn hoặc bằng 10000. Để có thể chạy với số lớn hơn ta không dùng mảng lưu như sau.

TINH CHẾ IẢN 5 (Không dùng Mảng, tập hợp)

```
Program Eratosthenes ;
Var i, j, k, n : integer ;
Begin
Repeat
Write ('N =  ') ; Readln (N) ;
Until n >= 2 ;
Write (*2*) ;
For i := 3 to N do Begin
K := 0 ;
For j := 2 to Trunc [Sqrt (i)] do If (i mod j) = 0 then K := 1 ;
If K = 0 then write (i) ;
End ;
End.
```

Ví dụ 3 (Bài toán 8 quân hậu)

Hãy tìm cách đặt 8 quân hậu trên một bàn cờ vua sao cho không có quân hậu nào có thể ăn quân hậu khác.

Theo thuật toán đệ quy có quay lui dựa trên phương pháp thử và sai đã trình bày ở §3 ta có chương trình Pascal như sau.

TINH CHẾ LẦN 1

```
Program EIGHT_QUEEN;
Var a, x : array [1.. 8] of integer;
b:array [2.. 16] of integer;
c:array [-7, 7] of integer;
i:integer;
Procedure print;
Var j : integer;
```

```
Begin
For j := 1 to 8 do write (x [j] : 4);
Writeln;
Readln;
End;
Procedure Try (i : integer);
Var j : integer ;
Begin
if i > 8 then print else
For j:= 1 to 8 do
if (a [j] = 0) and (b [i + j] = 0) and
(c [i - j] = 0) then
Begin
x [i] := j;
a U] := 1;
b [i + j]:= 1;
c [i - j]:= 1;
Try (i + 1);
c [i - j]:= 0;
b [i + j]:= 0;
a [j] := 0;
end;
end;
Procedure Init;

Var j : integer;
Begin
Fiilchar (a, sizeof (a), 0);
Fillchar (b, sizeof (b), 0);
Fiilchar (c, sizeof (c), 0);
End;
Begin [Chương trình chính]
Init;
Try (1) ;
End.
```

Sau khi chạy chương trình trên, một trong các kết quả có thể thấy trên màn hình là:

1

5

8

6

3

7

2

4

Ứng với một lời giải sau trên bàn cờ:

Muốn tránh dùng đệ quy ta có thể dùng vòng lạp Repeat như sau :

Xét cột đầu:

j := 1 ; i = 0;

Thử cột

Repeat

i := i + 1 (xét một hàng)

Antoan := a [i] and b [i + j] and C [i + j];

Until antoan or (i = 8)

Đặt quân hậu vào ô (i, j);

X [i] := i;

a [i] := False; b [i + j] := False;

c [i - j] := False;

Xét cột kế

j := j + 1; i := 0 Đã xong với cột cuối j > 8 Đã quay lại qua cột đầu j < 1 Quay lại

j : = j — 1; [xét lại cột trước]

if j > = 1 then

Begin

Loại quân hậu ở ô (i, j) {i = X [j]}

if i = 8 then Begin

j = j — 1

if j > = 1 then

Bỏ quân hậu ỏ ô (i, j) {i = X [j]}

end;

end;

end;

Bỏ quân hậu ở ô (i j);

i := x[j];

a [i] := True; b [i + j] := True; C [i — j] := True

Và chương trình PASCAL hoàn chỉnh là:

TINH CHẾ LẦN 2 (Không dùng Đệ Qui)

```
Program EIGHT QUEEN;
Const
HAU = 'Q';
OTRONG = '#';
Var
x : array [1.. 8] of integer; antoan : boolean;
a : array [1.. 8] of Boolean; i, j : integer
b : array [2.. 16] of Boolean;
c : array [-7.. 7] of Boolean;
Begin {Khởi tạo tình trạng ban đầu}
For i := 1 to 8 do a [i] := True;
For i := 2 to 16 do b [i] := True;
For i := -7 to 7 do C [i] := True;
j := 1
i := 0
Repeat
Repeat
i := i + 1
antoan := a [i] and b [i 4- j] and C [i - j];
Until antoan or (i = 8);
If antoan then
Begin {Đặt quân hậu vào}
X [j] := i;
a [i] := False; b [i + j] := False;
c [i — j] := False;
j := j + 1; (xét cột kế)
1 := 0;
End;
Else
Begin {Quay lại}
```

```
j := j — 1; {cột trước}
if j > = 1 then
Begin {Bỏ quân hậu đi}
i := x[j];
a [i] := True;
b [i + j] := True;
c [i - j] := True;
if i = 8 then
Begin
j := i — 1; {Quay thèm một cột} if j > = 1 then Begin
i := x [j];
a [i] := True; b [i + j] := True;
c [i - j] := True;
End;
End;
End;
End;
Until (j > 8) or (j < 1);
If j < 1 then Writeln ('VO NGHIEM');
Else
For i := 1 to 8 do
Begin
For j := 1 to 8 do
if x [j] := i then write (HAU)
else write (OTRONG);
Wrriteln;
End;
End.
```

Ví dụ 4 (Bài toán mã đi tuần)

Cho một bàn cờ có kích thước n x n. Một con mã di chuyển theo luật cờ vua được đặt trong một ô với tọa độ ban đầu là (xx, yx). Hãy tìm một đường đi với n^2-1 bước đi sao cho mọi ô trên bàn cờ đều được mã nhảy đến đúng một lần

Theo luật toán Đệ quy có quay lui dựa trên phương pháp thử và sai đã trình bày ở §8 ta có chương trình PASCAL như sau:

```
Program Knightour;
Const
```

```
a : array [a.. 8] of integer = (2, 1, - 1, - 2, - 2, - 1, 1, 2);
b : army [1.. 8] of integer = (1, 2, 2, 1, - 1, - 2, - 2, - 1);
n := 5; nsq := 25;
Type
Index = 1.. n;
Var
q : Boolean;
dd : Array [index, index] of integer;
S: Set of Index;
x, y : Integer;
Procedure Init;
Begin
Fillchar (dd, sizeof (dd), 0);
```

§5. CHẠY THỬ, THAY ĐỔI, KIỂM TRA CHƯƠNG TRÌNH

1. Chạy thử

Một chương trình đã viết xong chưa chắc đã chạy được trên máy tính để cho ra kết quả mong muốn.

Kĩ năng tìm lỗi, sửa lỗi, điềuu chỉnh, viết lại chương trình cũng là một kĩ năng quan trọng của người lập trình.

Quá trình rèn luyện kĩ năng này phải bắt đầu trong việc tìm lỗi và sửa chữa lỗi của chính mình.

2. Phân loại lỗi và cách sửa chữa

Lỗi về thuật toán: điều chỉnh lại thuật toán, thậm chí có thể loại bỏ thuật toán sai, tìm thuật toán khác nghĩa là làm lại từ đầu.

Lỗi về trình tự: phải xem lại thuật toán, phân tích lại từ trên xuống để đặt lại cho đúng với thuật toán.

Lỗi về cú pháp: viết lại cho đúng với cú pháp của ngôn ngữ lập trình mà mình đang sử dụng. (Các phần mềm ngôn ngữ hiện nay đều có phần tiện ích giúp đỡ tìm kiếm lỗi cú pháp và hướng dẫn sửa chữa).

3. Xây dựng các bộ Test

Có nhiều chương trình rất khó kiểm tra tính đúng đắn. Nhất là các chương trình tìm kiếm lời giải tối ưu, vì chúng ta chưa biết kết quả đúng là thế nào ? Vì vậy việc tìm lỗi rất khó khăn.

Để tháo gỡ các khó khăn trên, ta nên viết các chương trình cho nhập dữ liệu bằng Files thay vì bằng bàn phím (Vì mỗi lần sửa chữa trong khi chạy thử lại phải gõ bàn phím lại từ đầu).

Thông thường ta dùng các thủ tục có dạng sau để đọc các Files chứa các bộ Test.

```
Procedure Read data;
Var S : String;
f : Text;
X : Biến chứa dữ liệu;
Begin
Writen ('Xin cho biet ten File');
Readln (S);
Assign (f, S);
Reset (f);
Read (f,x);
Clowe (f);
End;
```

Khi tự mình phải làm các bộ Test để thử chương trình của chính mình ta nên lưu ý các điều sau:

• Nên khởi đầu bằng các bộ Test nhỏ nhưng chứa các giá trị đặc biệt (Vì kinh nghiệm cho thấy đây là chỗ dễ bị lỗi nhất).

• Làm nhiều các bộ Test nhưng phải đa dạng tránh lặp đi lặp lại các bộ Test tương tự.

• Nên kết thúc bằng các bộ Test có kích thước lớn để kiểm tra tính chịu dựng của chương trình.

Thông qua các bộ Test để điều chỉnh chương trình sửa chữa các lỗi khó thấy khi lập trình.

Phải ghi nhớ rằng một chương trình chạy qua được một số bộ test không có nghĩa là chương trình đó đã đúng. Bởi vì có thể là ta chưa xây dựng được bộ Test để chỉ ra cái sai của nó.

Nếu được, nên tìm cách chứng minh tính đúng đắn của chương trình (Việc này đôi lúc rất khó).

4. Thay đổi chương trình

Một chương trình đã viết xong, đã chạy tốt, đã giải được bài toán mà ta mong muốn điều đó chưa có nghĩa là quá trình lập trình đã kết thúc. Ta thường phải sửa đổi nó theo một hướng nào đó để đáp ứng một yêu cầu mới (Ví dụ tổng quát hóa vấn đề lên...). Chính ở đây phương pháp tinh chế từng bước tỏ ra có hiệu quả. Những câu lời tự nhiên ở những bước tinh chế đầu sẽ giúp ta dễ thấy được những

chỗ cần sửa trong chương trình cũ, giúp ta thừa HƯỚNG những lao động phân tích, thiết kế và chứng minh đã làm lúc trước.

Vậy phương pháp tinh chế từng bước có ý nghía đối với khả năng duy trì của chương trình.

Cũng vậy, phương pháp tinh chế từng bước còn giúp tăng cường tính phổ dụng của chương trình, nghĩa là sự chuyển đổi chương trình sang một môi trường khác ví dụ như sang một ngôn ngữ, lập trình khác hay một hệ thống máy tính khác.

Ví dụ (Bài toán 8 quân hậu tổng quát)

Tìm tất cả các cách đặt 8 quản hậu lên bàn cờ sao cho không có quân nào ăn nhau.

Trở lại với bước tinh chế thứ 2 của bài toán 8 quân hậu ở 4, ta thấy cần có 2 sửa đổi sau:

* Khi đã đặt được quân hậu cuối cùng vào bàn cờ, thì ta sẻ in lời giải đó ra, nhưng khống kết thúc chương trình. Mà ta sẽ thực hiện một sự quay lại để tìm một lời giải khác.

* Chương trình sẽ ngừng khi sự quay lại đã quá cột đầu.

Chương trình có dạng phác họa như sau:

```
Xét cột đầu
Repeat
Thử cột;
if an toàn then Begin
Đặt quân hậu vào;
Xét cột kế;
if cột kế vượt quá cột cuối cùng then
Begin
In ra lời giải;
Quay lại;
end;
end;
Else quay lại
Until đã quay lại quá cột đầu.
```

Chỉ cần đến đây là ta đã có thể viết lại thành chương trình hoàn chỉnh vì những câu trả lời tự nhiên đề đã được phân giải thành Pascal ở chương trình trước. Chương trình 8 quân hậu tổng quát như sau

```
Program EIGHTQUEEN;
Const
```

```
HAU = ’Q’;
OTRONG = ‘#’
Var
x : array [1.. 8] of integer;
a : array [1.. 8] of Boolean;
b : array [2.. 16] of Boolean;
c : array [-7.. 7] of Boolean;
i, j, Stt, ii, jj : integer; antoan, quaylui : Boolean;
Begin
Stt := 0; {Số thứ tự của một tình thế}
For i := 1 to 8 do a [i] := true;
For i := 2 to 16 do b [i] := true;
For i := —7 to 7 do C [i] := true;
{Tình trạng ban đầu} j := 1; {Xét cột đầu}
i := 0;
Repeat
Quay lui := False ; {Chưa cần quay lui}
Repeat
i := i+1 ; {thử một cột}
antoan := a[i] and b[i+j] and c[i — j] ;
Until antoan or (i = 8) ;
if antoan then
Begin {Đạt quân hậu vào}
x[j] := 1;
a[i] := False; b[i + j] := False; c[i - j] := False;
j := j + 1; {xét cột kế}
i := 0;
if j > 8 then {Đã quá cột cuối cùng}
Begin
Stt := Stt + 1;
Writeln (’Tinh the thu’, Stt : 3,);
For ii := 1 to 8 do
Begin
Write (ii : 3, ’ ’);
For jj := 1 to 8 do
if x [jj] = ii then
```

```
Write (HAU) else write (OTRONG);
Writeln ;
end ; {In ra một lời giải}
Readln ;
Quay lui := true;
{Quay lại tìm một lời giải khác}
End;
End;
if (not antoan) or quaylui then
Begin
j := j - 1
if j >= 1 then
Begin {Gỡ quân hậu ra}
o := x [j];
a[i] := True; b[i + j] := True; c[i - j] := True;
if i = 8 then
Begin {Quay lại thêm một cột nữa} j := j - 1; if j > = 1 then
Begin {Lại gỡ quân hậu ra}
i := x[j];
a[i] := True; b[i+j] := True; c[i—j] := True;
end;
end;
end;
end;
Until (j < 1) ; {Đã quay lại quá cột đầu}
End.
```

5. Tiêu chuẩn của chương trình

Thế nào là một chương trình tốt ?

Một chương trình tốt cần phải thỏa mãn bốn tính chất sau :

a) Tính tin cậy:

Một chương trình tin cậy là một chương trình chạy đúng như dự định.

Muốn một chương trình tin cậy, ta phải:

- Có một giải thuật đúng.
- Thể hiện một cách đúng đắn giải thuật thành chương trình (Cài đặt đúng).
- Sau khi đã dịch chương trình, phải tiến hành thử chương trình: Cho những dữ liệu mẫu vào, chạy thử chương trình và kiểm tra lại kết quả.

Tuy nhiên việc chạy thử chương trình chỉ giúp phát hiện một số lỗi, chứ không phải tất cả. Vì vậy khi viết chương trình xong phải kiểm tra tính đúng đắn của nó. Điều này dẫn đến sự ra đời của lí thuyết chứng minh chương trình.

- Một quan niệm quan trọng trong việc lập trình là : Thiết kế chương trình song song với chứng minh chương trình. Nghĩa là trong quá trình xây dựng chương trình ta luôn phải kiểm tra và chứng minh đúng đắn của các bước:

b) Tính uyển chuyển:

- Chương trình cần phải để sửa đổi.

- Trong quá trình sử dụng một chương trình, những nhược điểm của nó dần dần bộc lộ. Đồng thời những yêu cầu đối với chương trình cũng có thể có những thay đổi so với thiết kế ban đầu.

- Nếu ta viết đi viết lại chương trình từ đầu thì sẽ tốn rất nhiều công tác. Tốt hơn, hiển nhiên, là sửa chữa lại chương trình đã có. Một chương trình tốt phải có tính chất để sửa đổi, nhằm giúp tiết kiệm công sức của người lập trình trong quá trình phát triển chương trình.

- Trong lĩnh vực xây dựng phần mềm, người ta thường nói : Chương trình có chu trình sống gồm các giai đoạn:

• Thiết kế.

• Thi công.

• Sử dụng.

• Sửa đổi.

c) Tính trong sáng:

- Chương trình phải viết cho dễ đọc, dễ hiểu.

- Chương trình phải tạo thuận lợi cho việc bào trì bao gồm:

- Sửa sai.

- Cải tiến.

- Biến đổi.

- Để có một chương trình trong sáng, ta cần chú ý về : các tên của biến, kiểu, chương trinh con, ... phải có tính gợi nhớ, viết dưới dạng cấu trúc, sử dụng nhiều chú thích.

Một chương trình không trong sáng thường mắc các sai tóm:

• Các danh hiệu quá vắn tắt,

• Các phát biểu viết không chọn lựa,

• Không tạo dạng cấu trục.

Một chương trình như vậy không những sẽ khó đọc, khó hiểu khi kiểm tra mà còn có thể dẫn đến nhiều lỗi sai khó phát hiện.

d) Tính hữu hiệu:

Hữu hiệu là chạy nhanh và ít tốn bộ nhớ, nghĩa là ít tốn không gian và thời gian.

Để có một chương trình hữu hiệu, quan trọng nhất là có một giải thuật hữu hiệu. Sau đó, cán có những khéo léo nhất định, khi lập trình

Tuy nhiên, yêu cầu hữu hiệu nhiều khi làm cho chương trình trở nên rối rắm, khó hiểu, khi sửa đổi. Những nhà lập trình cấu trúc khẳng định, tiêu chuẩn hữu hiệu không quan trọng bằng 3 tiêu chuẩn trên.

Ngày xưa, khi ngành lập trình còn thô sơ, người ta thường chú trọng các mẹo vặt để nâng tính hữu hiệu (chạy nhanh, ít tốn bộ nhớ). Do đó, chương trình khó đọc, tối tăm. Ngày nay, với sự phát triển của phần cứng (máy chạy rất nhanh, có bộ nhớ lớn). Yêu cầu hữu hiệu không còn đặt ra quá nặng như trước nữa.

Một chương trình không phải chỉ là một tập các mệnh lệnh người giao cho máy mà còn là một tập các thông tin giao tiếp giữa người với người.

6. Kết luận.

Quá trình xây dựng chương trình là một chuỗi các bước tinh chế.

Ở mỗi bước, một công việc được phản rn lhni nhiều công viêc con. Mỗi sự tinh chế cho một mô tả công việc có thể được đi kèm bởi một tinh chế cho mô tả của dữ liệu – là cái tạo nên phương tiện liên lạc giữa các công việc. Sự tinh chế mô tả của chương trình và của các cấu trúc dữ liệu phải tiến hành đồng thời.

Mức độ của tính module đạt được theo cách này sẽ quyết định tính hÁt. dế dàng hay khó khốn trong việc làm thích ứng một chương trinh . An nửa đổi hay mở rộng về mục đích hay về môi trường thực thi (ngôn Itgiì, máy tính).

Trong quá trình tinh chế từng bước, mối cách biểu diễn tự nhiên của bài toán đang xét cần được sử dụng càng lâu càng tốt. Hướng phát triển của sự biểu diễn trong quá trình tinh chế được quyết định bởi ngôn ngữ lập trình mà ta dùng. Ngôn ngữ lập trình phải cho phép, chúng ta diễn tả một cách tự nhiên nhất và rõ ràng nhất các cấu trúc , tin chương trình và dữ liệu. Đồng thời nó cũng phải cho phép hướng dẫn quá trình tinh chế bằng cách trình bày những đặc tính căn bản và những nguyên lí cấu trúc tự nhiên đối với máy tính mà chương trình chạy trên đó.

Mỗi sự tinh chế dẫn đến một số các quyết định thiết kế, dựa trên một tập các tiêu chuẩn thiết kế.

Ví dụ:

- Tính hữu hiệu.
- Sự tiết kiệm bộ nhớ.
- Tính trong sáng.

• Tính cấu trúc.

Người lập trình phải được rèn luyện để có ý thức về các quyết dịnh liên quan và biết khảo sát nghiêm túc cũng như từ bỏ các lời giải nguy cả khi chúng là đúng.

Người lập trình phải học cách cân nhắc mọi phương tiện của từng lời giải theo tiêu chuẩn đó. Đặc biệt là phải biết từ bỏ những quyết định ta có, và đi ngược lên, nếu cần thiết, trở lại đỉnh.

Cần phải từ bỏ niềm tin tưởng ngây thơ ràng lập trình là dễ, các ngôn ngữ lập trình là đủ mạnh và các máy tính là đủ nhanh.

CHƯƠNG 2 CÁC BÀI TOÁN CƠ BẢN

§1. ĐỆ QUY

1. Khái niệm đệ quy.

— Một khái niệm X gọi là được định nghĩa theo đệ quy nếu trong định nghĩa X có sử dụng ngay chính khái niệm X.

Ví dụ 1 : Số tự nhiên.

• 0 là một số tự nhiên.

• n là số tự nhiên nếu n — 1 là số tự nhiên.

Ví dụ 2: Hàm giai thừa n !

• 0! = 1

• Nếu n > 0 thì n! = n (n — 1)!

Ví dụ 3 : Người giàu là

• Người có nhiều tài sản.

• Hoặc có cha mẹ giàu.

2. Chương trình con đệ quy.

Một chương trình con là một định nghĩa về một công việc nào đó. Vậy sự định nghĩa đó cũng có thể là đệ quy.

Ngôn ngữ PASCAL cho phép các chương trình con đệ quy.

Một chương trình con (hàm hoặc thủ tục) được gọi là đệ quy nếu trong quá trình thực hiện nó có phần phải gọi đến chính nó.

Ví dụ 1 : Hàm tính Giai Thừa của n (tính n!)

```
Functi0n GT (n : word) : Longint;
Begin
if n := 1 then GT :== 1 else
GT := n *GT (n - 1);
End;
```

Khi có lệnh gọi hàm, chẳng hạn X := GT (4);

Thì máy sẽ ghi nhớ là:

GT (4) := 4* GT (3) và đi tính GT(3)

Kế tiếp máy lại phải ghi nhớ:

GT (3) := 3* GT (2) và đi tính GT(2)

Kế tiếp máy lại phải ghi nhớ:

GT(2) := 2* GT(1) và đi tính GT(1)

Theo định nghĩa của hàm GT:

GT(1) := 1

Máy sẽ quay ngược lại : GT(2) := 2*1 = 2;

Rồi tiếp tục quay ngược : GT(3) := 3* 2 — 6;

GT(4) := 4*6 = 24.

Cuối cùng kết quả trả về là x = 24

Ví dụ 2 : Hàm tính lũy thừa nguyên của một số thực x (tính xn).

```
Functi0n LT (x : Real, n : integer) : Real ;
Begin
If n = 0 then LT := 1 else LT := x*LT (x, n - 1);
End;
```

Ví dụ 3 : Thủ tục tìm phần tử thứ i của dãy hoán vị của n sô tự nhiên từ 1 đến n.

```
Procedure Try (i : byte) ;
Var j : byte ;
Begin
If i > n then xuat else For j := 1 to n do
If b [j] then {số j chưa chọn}
Begin
a[i] := j ; b[j] := False ;
Try (i + 1) ; b[j] := True ; a[i] := 0 ;
End ;
End ;
```

3. Cấu trúc chính của một chương trình con đệ quy

Một chương trình con đệ quy về căn bản luôn gồm 2 phần.

* Phần cố định:

Trong đó chứa các tác động của hàm hoặc thủ tục với một số giá trị cụ thể ban đầu của tham số.

Ví dụ 1:

```
If n = 1 then GT 1 ;
```

Ví dụ 2

If n = 0 then LT := 1 ;

Ví dụ 3:

If i > ri. then Xuat ;

*Phần hạ bậc

Trong đó tác động cần được thực hiện cho giá trị hiện thời của các tham số được định nghĩa bằng các tác động đã được định nghĩa trước đây với kích thước nhỏ hơn của tham số:

Ví dụ 1:

If n > 1 then GT := n*GT(n-l) ;

Ví dụ 2:

If n > 0 then LT : = x*LT(x, n — 1) ;

Ví dụ 3:

If i <= n then Try (i +1) ;

4. Đệ quy hỗ tương

Nếu có 2 chương trình con B_1 và B_2 gọi lẫn nhau ta sẽ có một sự đệ quy hỗ tương.

Có những trường hợp cần phải tiến hành khai báo đầu một chương trình con trước khi khai báo đầy đủ chương trình con đó đặc biệt là trong đệ quy hỗ tương, khi B_1 gọi B_2 và B_2 lại gọi B_1.

Ví dụ:

```
Program A;
Procedure B2 (x : integer) : Forward ;
Procedure B1 (y : integer) ;
Begin
If y > 0 then
Begin
y := y + 1 ;
B2 (y)
End;
End;
Procedure B2; {Chỉ cần ghi tèn của chương trình con, không cần ghi lại danh sách tham số}
Begin
If x > 0 then
Begin
```

X := X — 2 ;

B1 (x) ; end ;

End ;

Begin {Chương trình chính}

B1 (3);

End.

5. Ưu điểm của đệ quy

Đè quy mạnh ở chỗ có thể định nghĩa một tập rất ỉớn các tác động chỉ bởi một số rất ít mệnh đề.

Đệ quy rất thích hợp để giải các bài toán có bản chất đệ quy.

Một chương trình viết theo giải thuật có tính đệ quy sẽ mang tính "Người” hơn do đó sẽ:

* Sáng sủa.

* Dễ hiểu.

* Nêu bật được bản chất của vấn đề.

Bản chất của đệ quy là ý tưởng quy nạp hay hạ bậc trong toán học nghĩa là đưa một số vấn đề phức tạp về một vấn đề đơn giản hơn.

Có rất nhiều thuật toán lí thú hoàn toàn được giải quyết đơn giản bằng các chương trình đệ quy, và rất nhiều nhà thiết kế thuật toán thích diễn đạt vấn đề theo kiểu đệ quy.

Có nhiều thuật toán cho đến nay vẫn chưa có cách giải không đệ quy.

Đệ quy là quả tim của các nghiên cứu lí thuyết rất tự nhiên của tính toán, các hàm và chương trình đệ quy đóng vai trò trọng tâm trong các nghiên cứu toán học nhằm cố gắng tách rời các bài toán có thể giải quyết bằng máy tính khỏi các bài toán không thể giải quyết bàng máy tính.

Lập trình đệ quy là một phương tiện hiệu quả để tổ chức một thao tác tìm kiếm phức tạp trong một tập hợp các khả năng có thể có.

6. Các bài toán đệ quy cơ bản

Bài 1 (Bài toán Hoán vị)

Tìm tất cả các hoán vị của một mảng gồm có n phần tử.

THUẬT TOÁN 1:

Giả sử đã có được tất cả các hoán vị của n — 1 phần tử ra tìm cách sinh ra tất cả các hoán vị của n phần tử của dãy a : HV (n)

For j := n downto 1 do

Begin

đổi chỗ (a[n], a[j]) ;

```
HV (n - 1);
đổi chỗ (a[n], a[j]) ;
End ;
```

Chương trình PASCAL như sau

```
Program HOAN_VI1 ;
Const n = 10 ;
Type Day = array [1..n] of integer ;
Var a : Day ;
m, n, i : integer ;
Procedure Init ;
Var i : integer ;
Begin
For i := 1 to n do a[i] := I;
End ;
Procedure DOI_CHO (var x, y : integer);
Var z : integer ;
Begin
z := x;
x := y;
y := z;
End ;
Procedure Xuat ;
Var i : integer;
Begin
Wrireln ;
For i := 1 to n do write (a[i]);
End ;
Procedure HV (I : integer) ;
Var j : integer;
Begin
If i=1 then Xuat
Else
For j := i downto 1 do
Begin
DOI_CHO (a[i,j, a[jl) ;
HV (i-1);
```

```
DOI_CHO (a[i], a[j]) ; End;
End ;
Begin
Init ;
HV (n) ;
Readln ;
End.
```

THUẬT TOÁN 2

— Ta đặt mảng A[1...N] để chứa hoán vị vừa tìm được.

— Mảng B[1..N] of 0..1 để làm cờ với B[i] cho biết số i đã được chọn vào hoán vị hay chưa.

— Thuật toán được lập theo kiểu đệ quy với 2 thủ tục là : PRINT và TRY (i : byte);

— Thủ tục PRINT sẽ in ra hoán vj vừa Um được.

— Thủ tục FIND (i : byte) có nhiệm vụ tìm phần tử thứ i trong hoán vị và được gọi một cách đê quy. Cơ chế hoạt động của nó như sau:

```
Procedure FIND (i : byte) ;
Var j : byte ;
Begin
If i > N then PRINT Else
Begin
For j := 1 to N do If B[j] then Begin a[i] := j;
b[j] := 0 ;
FIND (i + 1) ;
B[j] := 1 ;
End ;
End ;
End ;
```

Ví dụ: N = 3

Chương trình PASCAL như sau :

```
Program HOAN_VI2 ;
Const N = 10 ;
Var
A : Array [1..N] of byte ;
B : Array [1..N] of 0..1 ;
i, dem : integer ;
```

```
Procedure PRINT ;
Var i : integer ;
Begin
dem := dem + 1 ;
Write (‘Hoan vi thu’, dem, ”) ;
For i := 1 to N do Write (A[i]) ;
Writeln ;
Readln ;
End ;
Procedure FIND (i : byte) ;
Var j : byte ;
Begin
If i > N then PRINT ;
Else
Begin
For j := 1 to N do
If (B[j] = 1) then
Begin
A[i] := j ;
B[j] := 0 ;
FIND (i + 1) ;
B[j] := 1 ;
End ;
End ;
End ;
Begin
Dem := 0 ;
For i := 1 to N do B[i] := 1;
FIND (1) ;
End.
```

Bàl 2 (Bài toán tháp Hà Nội)

Có 2 cột A, B, C và n dĩa kích thước khác nhau ở cột A, đĩa lớn ở dưới, đĩa nhỏ ở trên. Hãy chuyển các đĩa từ A sang C sao cho:

- Mỗi lần chỉ được chuyển một đĩa.
- Ở các cột, tình trạng đĩa luôn luôn là đĩa lớn ở dưới, đĩa nhỏ ở trên.

* Ta suy nghĩ theo kiểu đệ quy như sau:

- Ta muốn chuyển n đĩa từ cột A sang cột c, lấy cột B làm trung gian.

- Bởi vì đĩa lớn nhất phải nhm dưới bên phải làm cách nào đó chuyển n—1 đĩa trên cùng từ cột A sang cột B lấy C làm trung gian rồi chuyển cái đĩa lớn nhất ở dưới cùng từ cột A sang cột C. Và rồi ta phải làm cách nào đó chuyển n — 1 đĩa từ cột B sang cột C lấy A làm trung gian.

* Thuật toán tóm tắt như sau:

Để chuyển n đĩa từ A sang C, lấy B làm trung gian.

Ta làm:

— Chuyển n—1 đĩa từ A sang R, lÁy C làm trung gian.

— Đem 1 đĩa từ A sang c.

— Chuyển n-1 đĩa từ B sang (!, l\y A làm trung gian.

* Viết lại thành dạng thù tục

```
Procedure Doicot (n, A, B, C) ;
Begin
If n > 0 then
Begin
Doicot (n—1, A, c, B) ;
Chuyển một đĩa (A, C) ;
Doicot (n—1, B, A, C) ;
End ;       .       .
End ;
```

* Chương trình PASCAL đầy đủ như sau:

```
Program Thap Ha_Noi ;
Type
Cot = Char; (Có 3 cột là 'A', 'B', 'C')
Var
n : integer ;
Procedure ChuyenDia (X, Y : cột) ;
Begin
Writeln ('Chuyển một đĩa từ', X' sang', Y) ; ) ;
End ;
Procedure DoiCot (n : integer ; A, B, C : Cot) ;
Begin
If n > 0 then
Begin
```

```
DoiCot (n—1, A, c, B) ;
ChuyenDia (A, C) ;
DoiCot (n— 1, B A, C) ;
end ;
Begin {Chương trình chính}
Write (’Xin cho biết số đĩa’) ;
Readln (n) ;
DoiCot (n, A, B, C) ;
End.
```

Chương trinh sẽ thực hiện như sau với n = 3

Các hước chuyển đĩa sê là:

A - C

A - B

C - B

A - C

B - A

B - C

A - C

Bài toán 3 (Dãy Fibonacci)

Tính số hạng thứ n của dãy Fibonacci được định nghĩa như sau :

$F_o=1$

$F_1=1$

$F_n = F_n\text{-}1 + F_n\text{-}2$

Chương trình PASCAL như sau

```
Program Fibonacci :
Var
n : integer ;
Functi0n Fib (n : integer) : integer ;
Begin
If n < = 1 then Fib := 1 Else
Fib := Fib(n-1) + Fib(n-2) ;
End ;
Begin
Writeln (’xin cho biet n’) ;
Readln (n) ;
Writeln (’So hang thu’, n, ’cua day Fibonacci la’, Fib (n)) ;
```

```
Readln ;
End.
```

Bài 4 (Tìm ước số chung lớn nhất)

Tìm ước chung lớn nhất của hai số.

* Ta dựa vào tính chất đệ quy sau đây của ước chung:

USC(m, n) = USC(n, m Mod n)

* Chương trình PASCAL như sau:

```
Program USCLN ;
Var
So1, So2 : integer ;
Functi0n USC (m, n : integer) : integer ;
Var
r : integer ;
Begin
r := m Mod n ;
If r = 0 then
USC:= n
Else
USC := USC(n, r) ;
End ;
Begin
Writeln ('Cho số thứ 1'); Readln (So1) ;
Writeln ('Cho số thứ 2') : Readln (So2) ;
Writeln ('Ước số chung lớn nhất là', USC (So1, So2)) ;
Readln ;
End.
```

Bài 5 (Đảo ngược chữ số)

Lập trình để nhập vào một số nguyên. Rồi in ra chữ đó theo thứ tự đảo ngược:

Ví dụ : Nhập vào 1987.

In ra : 7891

* Chương trình PASCAL như sau:

```
Program DAONGUOCCHUSO ;
Var
So : integer ;
Procedure DAOCHUSO (conso : integer) ;
```

Begin

Write (conso Mod 10 : 1) ;

If (Conso DIV 10 <> 0 then

DAOCHUSO (conso DIV 10) ;

End ;

Begin

Write (‘Cho vao mot so nguyen duong’);

Readln (So) ;

DAOCHUSO (So) ;

Writeln ;

Readln ;

End.

Bài 6 (Tính giá trị của số LA MÃ)

Lập trình để tính giá trị của một số viết dưới dọng LA MÃ

Ví dụ : MDCLXVT = 1666

GIẢI THUẬT :

— Ta đạt tên hàm đệ quy như sau

LAMA (r : string; i : integer) : integer

Hàm này có nhiệm vụ tính giá trị của số La Mã được viết dưới dạng String r từ vị trí thứ i cùa String.

— Đặt X := Giatri (r [i]);

— Điều kiện dừng là:

If i = Length (r) then LAMA := x;

— Phần đệ qui là:

If i < Lenght (r) then

If x < Giatri (r [i + 1]) then

LAMA := X + LAMA (r, i + 1)

Else LAMA := - X + LAMA (r, í + 1);

* Chương trình PASCAL như sau

Program TINH_SO_LA_MA;

Var r : String;

Procedure NHAP;

Begin

Writeln (’Xin cho biet so Lama’);

Readln (r); end;

Functi0n LAMA (r : string; i : integer) : integer;

```
Vnr X : integer;
Functi0n GIATRI (c : char) : integer,
Begin
Case C of
’M’, ’m’ : GIATRI := 1000;
‘D’, ’d’ : GIATRI := 500;
‘C’, ‘c’ : GIATRI := 100;
‘L’, ‘l’ : GIATRI := 50;
‘X’, ‘x’ : GIATRI := 10;
’V’, ‘v’ : GIATRI := 5;
‘I’, ‘i’ : GIATRI := 1;
Else
GIATRI := 0; .
End;
End;
Begin
x := GIATRI (r [i]);
If i = Length (r) then LAMA := x
Else
If x < GIATRI (r[i + 1]) then
LAMA := - x + LAMA (r, i + 1)
Else
LAMA := x + LAMA (r, i + 1);
End;
Begin {Main Program}
NHAP;
Writeln (‘Gia tri cua so La ma’, r, ’la', :AMA (r, 1));
Readln;
End.
```

Bài 7 (Bài toán sắp xếp)

Cho dãy số nguyên A [1..n] hãy sắp xếp A theo thư tự tăng

Thuật toán (QUICKSORT)

Ta dùng phương pháp phân vùng để xử lí nội dung chính là chọn phần tử x ở giữa của dãy làm chuẩn để so sánh, từ đó phân hoạch dãy này thành 3 dãy con liên tiếp như sau:

* Dãy con thứ nhất gồm các phần tử nhò hơn x

* Dãy con thứ hai gôm các phần tử bàng x.

* Dãy con thứ ba gồm các phần tử lớn hơn x.

Sau đó lại áp dụng giải thuật phàn hoạch này cho dãy con thứ nhấc và dãy con thứ ba, nếu các dãy con này có nhiều hơn một phần tử (Đệ quy).

Cụ thể là xét một đoạn của dáy từ thành phần thứ L đến thành phấn thứ R ta thực hiện các bước sau:

* Lấy giá trị của thành phân thứ (L + R) div 2 gán vào biến x

* Cho i ban đầu là L

* Cho j ban đầu là R

* Lặp lại

• Chừng nào còn A[i] < X thì tăng i

• Chừng nào còn A[j] > X thì giảm J.

• Nếu i < = j thì

- Hoán vị A [i] và A [j]

- Tăng i

- Giảm j

Cho đến khi i > j

• Sắp xếp đoạn từ A [L] đến [j]

• Sắp xếp đoạn từ A [i] đến A [R]

Ví dụ :

Sắp xếp dãy số:

39

50

7

37

89

13

1

62

- Đầu tiên ta gán X = 37

- Sau 2 lần đối chỗ ta được dãy

1

13

7

37

89

50

39

62

— Xử lý dãy con: 1, 13, 7

— Ta được: 1, 7, 13

— Xử lý dãy con: 89, 50, 39, 62

— Ta được: 39, 50, 89, 62

39, 50, 62, 89

Cài đặt bằng PASCAL như sau:

```
Procedure
QUICKSORT (Var A : Array [l..n] of integer);
procedure SORT (L, R : integer);
Var i, j, TG, X : integer;
Begin
X := A [(L + R) div 2];
i := L;
j := R;
Repeat
While (A [i] < X) do inc (i);
While (A [j] > X) do dec (j);
If i <= j then
Begin
TG := A [i];
A [i] := A [j];
A [j] := TG;
inc (i);
Dec (j);
End;
Until i > j
IF L < j then SORT (L, J);
IF R > i then SORT (I, R);
Begin
SORT (1, N);
End;
```

Thuật toán 2 (SELECTI0NSORT)

Ta dùng thủ tục đệ quy FIND (i : integer).

```
Begin
IF i < n then Begin
For j := i + 1 to n do
IF A [i] > A [j] then
Hoán vị (A [i]; A [j]);
FIND (i + 1);
End;
End;
```

- Cài đặt bằng PASCAL như sau:

```
Procedure SELECTI0NSORT (Var A : Array [1... n] of integer);
Procedure FIND (i : integer);
Var TG, j : integer;
Begin
IF i < N then
Begin
For j := i + 1 to n do
IF A [i] > A [j] then
Begin
TG := A [i];
A [i] := A [j];
A [j] := TG;
end;
FIND (i + 1);
End;
End;
Begin
FIND (1);
End ;
```

7. Khử đệ quy

- Có một số giai thuật đệ quy thuộc loại tính toán đơn giản có thể được thay thế bởi một giải thuật khác không tự gọi nó, sự thay thế dó được gọi là khử đệ quy.

Tuy nhiên, như vậy không có nghĩa là phải khử đệ quy bằng mọi giá và không nên e ngại cũng như ác cảm với việc dùng đệ quy.

Ví dụ 1

* Hàm đệ quy sau:

```
Functi0n GT (n : word) : longint;
```

```
Begin
IF n = 0 then GT := 1
Else GT := n * GT (n - 1);
End;
```

* Có thể được khử đệ quy như sau:

```
Functi0n GT (n : word) : longint;
Var
i : word;
T : longint;
Begin
T := 1;
IF i > 0 then
For i := 1 to n do T := T * i;
GT := T;
End;
```

Ví dụ 2

* Hàm đệ quy sau:

```
Functi0n Fib (n : integer) : integer;
Begin
If n := 0 then Fib := 0
Else IF n = 1 then Fib := 1
Else Fib := Fib (n - 1) + Fib (n - 2);
End;
```

* Có thể được khử đệ quy như sau:

```
Functi0n Fib (n : integer) : integer;
Var
i, x, y : integer;
Begin
If n := 0 then Fib := 0
Else If n := 1 then Fib := 1
Else Begin
i := 1 ; y := 0; x := 1;
While i < n do
Begin
i := i + 1;
x := x + y;
```

```
y := x - y;
End;
Fib := x;
End;
End;
```

Ví dụ 3

Thủ tục đệ quy SELECTI0N SORT sau đây

```
Procedure SELECTI0NSORT (Var A : Array [1.. n] of integer);
Procedure FIND (i : integer);
Var TG, j : integer;
Begin
If i < N then Begin
For j := i + 1 to n do
If A [i] > A [j] then
Begin
TG := A [i];
A [i] := A [j];
A [j] := TG;
end;
FIND (i + 1);
end;
end;
Begin
FIND (1);
End;
```

Có thể khử đệ quy như sau:

```
Procedure SELECTI0NSORT (Var A : Array [1.. n] of integer);
Var TG, i, j : integer;
Begin
For i := 1 to n - 1 do
For j := 1 + 1 to n do
If A [i] > A [j] then
Begin
TG := A [i];
A [i] := A 01;
A [j] := TG;
```

end;

End;

8. Kết luận về đệ quy

— Đệ quy quả là một vùng đất đầy quyến rũ đối với người lập trình, tại đó những vấn đề khó nhất được giải quyết bằng một vài dòng lệnh.

— Không nên e ngại đệ quy cũng như không nên lạm dụng đệ quy.

— Nghệ thuật giải quyết bài toán trong tin học là nghệ thuật (quyết định khi nào dùng đệ quy và khi nào không dùng đệ quy.

— Cần lưu ý rhng đệ quy không những là một phương pháp lập trình quan trọng mà còn là một phương pháp suy nghĩ để giải quyết vấn đề một cách tổng quát dựa trên ý tưởng:

* Đơn giản hóa công việc

* Phân vùng để xử lí

Trong các phần sau như:

* Tìm kiếm

* Vét cạn

* Đồ thị

Chúng ta sẽ thường xuyên gộp lại đệ quy.

§ 2. TÌM KIẾM

1. Khái niệm

Một trong cái nhiệm vụ cơ bản của máy tính là tìm kiếm.

Có thể nóo nhưng hầu hết các bài toán trong tin học đều sử dụng bài toán tìm kiếm.

Ví dụ:

Tìm một đối tượng thỏa mãn một số các điều kiện nào đó trong một tập hợp các đối tượng đã xác định.

Tìm một lời giải tối ưu trong một tập các lời giải của một bài tóan đã xác định.

2. Phương pháp tổng quát

Để giải một bài toán tìm kiếm ta thường thực hiện các bước sau.

Bước 1:

- Xác định tập hợp tìm kiếm sao cho không thừa, không thiếu, không sai.

- Tổ chức dữ liệu để có thể mô tả tập hợp tìm kiếm một cách đơn giản và chính xác.

Bước 2:

- Xác định đối tượng cần tìm kiếm.

- Phân tích các đặc trưng của đối tượng cần tìm kiếm.

- Biểu diễn các đặc trưng đó thành các biểu thức BOOLEAN.

Bước 3:

Xác định thuật toán (chiến lược) tìm kiếm ví dụ như sau:

- Tìm tuần tự

- Tìm nhị phân

- Thử sai bằng vét cạn

- Thử sai và quy lui

- Tìm bằng một thuật toán đặc biệt

3. Tìm kiếm tuần tự

Để đơn giản, chúng ta chỉ trình bày qua ví dụ tìm một giá trị X nguyên xem có nằm trong một dãy các số nguyên hay không?

Nếu có, thì cho biết vị trí của nó.

THUẬT TOÁN

- Đi từ phần tử đầu đến phần tử cuối của dảy A [1.. n].

- So sánh từng phần tử của dãy với giá trị X.

- Nếu tìm thấy giá trị X thì thoát ra và thông báo thứ tự của phần tử thứ i thỏa A [i] := X.

- Nếu sau khi đi hết mà không tìm thấy thì việc tìm kiếm xem như đã kết thúc với kết quả là 0.

Chương trình PASCAL như sau:

```
Functi0n
Tim (x : integer; A : Array [1.. n] of integer) : Word;
Var i : word;
Begin i := 1
While (i <= n) and n0t (x = A [i]) do inc (i);
If i > n then Tim := 0;
Else Tim := i;
End; s
```

4. Tìm kiếm nhị phân trên một mảng đã được sắp

THUẬT TOÁN

- Giả sử mảng A [1.. n] đã được sắp thứ tự tăng dần.

- Ta chia đôi mảng A xem thành phần ở giữa có giá trị lớn hơn hay nhỏ hơn giá trị X để từ đó xác định nên tiếp tục tìm kiếm ở nửa trẽn hay nửa dưới.

- Giả sử chọn được dãy trên, ta tiếp tục chia đôi dãy trên để tìm kiếm cho đến khi gặp giá trị X (tìm thấy) hoặc không thể chia đôi được nửa (không tìm thấy).

Cụ thể là:

- Ta dùng 3 biến Low, High, Mid để lưu giữ chỉ số của đầu, cuối, giữa của dãy đang tìm kiếm.

- Ban đầu cho Low := 1; High := n;

- Chừng nào High > Low thì thực hiện:

• Mid := (Low + High) div 2;

• Nếu x > A [Mid] thì

Low := Mid + 1;

• Ngược lại {x <= A [Mid]}

• Nếu x < A [Mid] thì

High := Mid - 1;

• Ngược lại {x = A [Mid]}

High := -1;

• Nếu High = - 1 thì tim thấy x tại thành phần thứ Mid

• Ngược lại thì không tìm thây.

Chương trình PASCAL như sau:

```
Functi0n
Tim (x : integer; A : Array [1.. n] of integer) : word;
Var
Low, High, Mid : integer;
Begin
Low := 1; High := n;
While High >= Low do
Begin
Mid := (High + Low) div 2;
If (A [Mid] < x) then
Low := Mid + 1
Else
If (x < A [Mid]) then
High := Mid - 1
Else
High := -1
End;
If high = - 1 then Tim := Mid;
Else Tim := 0
End;
```

5. Tìm bằng vét cạn — thử sai

THUẬT TOÁN TỔNG QUÁT

— Duyệt hết không gian tìm kiếm (vét cạn).

— Kiểm tra xem từng đối tượng một có thỏa tính chất của đối tượng cần tìm hay không nếu đúng thì lưu lại, nếu sai thì bỏ qua (thử sai).

Ví dụ 1

Lập trình tìm tất cả các cách thay thế các dấu ? bởi các dấu phép tính +, =, *, /, trong biểu thức dưới đây sao cho biểu thức có trị bằng 35:

((((1 ? 2) ? 3) ? 4) ? 5) ? 6

THUẬT TOÁN

— Viết hàm F để tính toán như sau

```
Functi0n
F(X, Y, d : integer) : Real;
Begin
Case d of
1 : F := (X + Y);
2 : F := (X *- Y);
3 : F := (X * Y);
4 : F := (X / Y);
End;
```

— Vét cạn tất cả cốc cách điền dấu bằng 5 vòng lặp:

```
For i := 1 to 4 do
For j := 1 to 4 do
For k := 1 to 4 do
For 1 := 1 to 4 do
For m := 1 to 4 do
F (F (F (F (1,2,i), 3, j), 4, k), 5, l), 6, m) = 35
Then Xuat,
```

Chương trình Pascal như sau

```
Program DAU;
Const
A : Array [1.. 4] of char =
('+','-','*','/');
Var
i, j, k, l, m : integer;
Procedure Xuat;
```

```
Begin
Writeln (‘((((l’, A [i], '2)’, A [j], ‘3)’ A [k], '4)’ A [1], ‘5)’, A [m], ‘6)=35’);
Readln;
End;
Functi0n F(X, Y, d : integer):Real;
Begin
Case d of
1 : F := (X + Y);
2 : F := (X - Y);
3 : F := (X * Y);
4 : F := (X / Y);
End;
Begin {Main program}
For i := 1 to 4 do
For j := 1 to 4 do
For k := 1 to 4 do
For 1 := 1 to 4 do
For m := 1 to 4 do
If F (F (F (F (1,2i),3j),4,k),5,l),6,m) = 35 then Xuat;
End.
```

Ví dụ 2

Tìm cách điền chín chữ số khác nhau {1,2,3,4,5,6,7,8,9} vào bảng vuông 3x3 như sau:

a	b	c
a*	b’	c*
a”	b”	c”

Sao cho

a’b’c’ = 2abc

a”b”c” = 3abc

THUẬT TOÁN

—Vét cạn tất cả các cách chọn abc bằng một vòng lặp:

```
For i : = 123 to 333 do
Begin
j := 2 * i;
k := 3 * i;
```

```
KT(i, j, k);
End;
```
— Thủ tục kiểm tra được viết như sau :
```
Procedure KT(i, j, k : integer);
Var x : Array [1.. 9] of integer;
Begin
x[1] = i Mod 10;
x[2] = (i div 10) mod 10;
x[3] = i div 100;
x[4] = j Mod 10;
x[5] = (j div 10) mod 10;
x[6] = j div 100;
x[7] = k Mod 10;
x[8] = (k div 10) Mod 10;
x[9] = k div 100;
If ([x[1], x[2], x[3], x[4], x[5], x[6], x[7], x[8], x[9]]
= [1, 2, 3, 4, 5; 6, 7, 8, 9])
then
Begin
Writeln (x[3], x[2], x[1]);
Writeln (x[6], x[5], x[4]);
Writeln (x[9], x[8], x[7]);
End;
End;
```
Chương trình PASCAL như sau :
```
Program BANGSO;
Var i, j, k : integer;
Procedure KT(i, j, k : integer); Var X : array [1.. 9] of integer; Begin
x[1] = i Mod 10;
xt2] = (i div 10) Mod 10;
x[3] = i div 100;
x[4] = j Mod 10;
x[5] = (j div 10) Mod 10;
x[6] = j div 100;
x[7] = k Mod 10;
x[8] = (k div 10) Mod 10;
```

```
x[9] = k div 100;
If ([x[l], x[2], x[3], x[4], x[5], x[6], xt7], x[8], x[9]]
= [1, 2, 3, 4, 5, 6, 7, 8, 9])
then
Begin
Writeln (x[3], x[2], x[1]);
Writeln (x[6], x[5], x[4]);
Writeln (x[9], x[8], x[7]);
Readln;
End;
End;
Begin {Main Program};
For i := 123 to 333 do
Begin
j := 2 * i;
k := 3 * i;
KT(i, j, k);
End;
End.
```

6. Tìm bằng đệ quy — thử sai — Quay lui

THUẬT TOÁN TỔNG QUÁT

— Chia không gian tìm kiếm ra thành nhiều cấp (cây tìm kiếm).

— Dừng thủ tục đệ quy đi từ cấp thấp đến cấp cao.

— Tại mỗi cấp xét tất cả các cách có thể chọn được.

— Chọn thử một cách, nếu còn hi vọng chọn đúng (thử sai) thì tiếp tục lên cấp cao hơn.

— Nếu đã sai thì quay lui, gỡ bỏ những gì vừa chọn ngay phía trước để chọn lại cách khác.

- Ví dụ 1

Tìm tất cả các hoán vị của dãy số nguyên (1,2,3,..., n).

THUẬT TOÁN

Ta chia không gian tim kiếm thành n cấp ứng với n số hạng của 1 hoán vị.

Đặt cờ B[1.. n] of Boolean trong đó B[j] cho biết số j đã được chọn cho hoán vị hay chưa.

Dùng thủ tục đệ quy sau đây để thử sai và quay lui.

```
Procedure Try (i : integer);
```

```
Var j : integer;
Begin
If i > n then Xuat
Else For j := 1 to n do
If B [j] then
Begin
A [i] := j; {Chọn}
B [j] := False;
Try (i + 1); {Đệ quy}
B [j] := True; {Quay lui}
A [1] := 0;
end;
End;
```

Chương trình Pascal đầy đủ như sau :

```
Program HOAN_VI;
Const n = 5;
Var A : Array [1.. n] of integer;
B : Array [1.. n] of Boolean;
Procedure XUAT;
Var j : integer;
Begin
For j := 1 to n do write (A [j] : 4);
Writeln;
End;
Procedure Init;
Var j : integer;
Begin
For j := 1 to n do B [j] : = True;
End;
Procedure TRY (i : integer);
Var j : integer;
Begin
If i > n then Xuat
Else
For j : = 1 to n do
If B [j] then
```

```
Begin
A [i] := j;
B [j] := False;
Try (i + 1);
B [j] := True;
A [i] := 0;
End;
End;
Begin {Main Program}
Init;
Try (1);
End.
```

Ví dụ 2

Cho 3 kí tự A, B, c. Từ 3 kí tự trên hãy viết chương trình tạo ra chuối kí tự thỏa mãn các tính chất sau:

1) Có độ dài 100;
2) Không có hai chuỗi con liên tiếp nào giống nhau;
3) Số kí tự B là ít nhất.

THUẬT TOÁN

- Ta chia không gian tìm kiếm thành 100 cấp ứng với 100 số hạng của chuỗi.
- Dùng hàm OK sau đây đố kiổm tn» tính đúng đắn của chuỗi (x1, x2,...xi) với giả thiết rằng chuỗi (x1, x2,…, xi-1) đã đúng.

Chú ý các vị trí sau của 2 chuỗi con có độ dài K:

```
Functi0n OK (i : integer) : Boolean;
{* Kiểm tra điều kiện 2 của đắu bài *}
Var
K : integer;
Begin
OK := True;
For K := 1 to Trunc (i/2) Do
If Copy (Y, i - 2 * K + 1, K) = Copy (Y, i-K+1, K) then
Begin
OK := False;
Exit;
End;
End;
```

— Dùng thủ tục đệ quy sau đây để thử sai và quay lui.

```
Procedure TRY (i : integer);
Var j : integer,
Begin
If i > n then.CHECKMIN Else
For j := 1 to 3 do Begin
Y := X + M Cl;
If OK (i) then Begin
X := X + M 01;
IF M [j] = 'B' then
Count := Count + 1;
TRY (i + 1);
If M [j] := 'B' then count := count — 1
Delet (X, i, 1);
end;
end;
End;
```

- Đề tìm chuối có 9Ố kí tự B ít nhất ta dùng thủ tục CHECKMIN sau:

```
Procedure CHECKMIN;
Begin
If Min > Count then
Begin
Min := Count;
Z := X;
End;
```

Chương trình PASCAL đầy đủ như sau Program FINDSTRING;

```
Const N = 100;
Type Chuoi = String [100];
Var X, Y, X : chuoi;
M  : String [3];
Count, Min : integer,
Procedure Init;
Begin
Count := 0;
Min := 100;
X :=
```

```
M := ’ACB’;
end;
Functi0n OK (i : integer) : Boolean;
Var
K : integer;
Begin
OK : = True;
For K := 1 to Trunc (i/2) do
If copy (Y,i - 2 * K + 1, K) = Copy (Y, i - K +1, K)
Begin
OK := False;
Exit;
end;
End;
Procedure CHECKMIN;
Begin
If Min > Count then Begin
Min := Count;
z := X;
end;
End;
Procedure TRY (i : integer);
Var j : integer;
Begin
If i > N then CHECKMIN Else
For j := 1 to 3 do
Begin
Y := X + M[j];
If OK[i] then
Begin
X := X + M [j] ;
If M [j] = ‘B’ then Count := Count + 1 ;
Try (i + 1) ;
If M UI = ’B’ then Count := Count – 1 ;
Delet (x , i , 1) ;
end ;
```

```
end ;
End;
Begin {Main Program}
Init ;
Try (1) ;
Writeln (‘Chuỗi căn tìm là : ‘ , Z) ;
Writeln (‘ Số phần tử B có trong chuỗi là : ‘ , Min);
End.
```

Ví dụ 3

Trong hệ tọa độ vuông góc, cho tọa độ của n hòn đảo là M1(X1, Y1) ,M2(X2 , Y2),..., Mn(Xn , Yn) Giả thiết rằng tất cả các thùng chứa của Canô chỉ đủ chứa một số xăng để đi quãng đường dài không quá L km cho trước. Trên mỗi đảo đều có sẵn sàng dự trữ để Canô có thể nạp đầy các thùng chứa. Hãy tìm mọi đường đi có thể của Canô xuất phát từ đảo Nj(Xj , Yj) đến đảo Nj(Xj,Yj) và chỉ ra một đường đi tối ưu (có số lần ghé vào đảo để lấy xăng là ít nhất).

THUẬT TOÁN

— Ta tìm tất cả các đường đi .và đường đi tối ưu bàng thủ tục đệ qui sau đây để thử sai và quay lui.

```
Begin {Main Program}
Init ;
Try (1) ;
Writeln (’Chuối cân tim là : ’ , Z) ;
Writeln (’Số phần tử B có trong chuỗi là : ‘, Min) ;
End.
Procedure Try (i : integer) ;
Var j : integer ;
Begin
If j = Target then
Begin Print ;
Findmin ;
end
Else
For j := 1 to n do
If n0t (tt [j]) and (m >= Kc (i, j)) then
Begin
t := t + 1 ;
```

```
Dao [t] := j ;
tt [j] := True ;
Sl := Sl + 1 ;
Try (j) ;
Sl := Sl - 1 ;
tt [j] := False ;
t := t - 1 ;
end ;
End ;
```

- Ta tìm đường đi. tối ưu tòng thủ tục sau

```
Procedure Findmin ;
Begin
If Min > Sl then begin
Min := Sl ;
t1 := t ;
Dao1 := Dao ;
End ;
End ;
```

Ta xuất các đường đi tìm được tòng thủ tục sau :

```
Procedure Print ;
Var j : integer ;
Begin
Tot := True ;
Writeln (‘Tồn tại đường đi: ’) ;
For j := 1 to t do
Write (‘M’ , Dao [j] , ’ = >’) ;
Writeln ;
End ;
```

- Ta xuất đường đi tối ưu bằng thủ tục sau

```
Procedure Printmin ;
Var j : integer ;
Begin
Writeln ;
Writeln (‘Đường đi tối ưu: ’) ;
For j := 1 to t1 do
Write (’M’, Dao1 (j] , ‘= >’) ;
```

```
Writeln ;
Writnln (’Số lần ghé qua:’ , Min - 1)
End ;
```

Toàn bộ chương trình PASCAL như sau

```
Program TIMĐAO; Uses crt ;
var
Dao, Dao1, X, y : Array [1.. 100] of integer ;
tt : Array [1.. 100] of Boolean ;
Min, Sl, n, t, t1, i, Source, Target : integer ;
s, m : real ;
Tot : Boolean ;
Procedure Init ;
Begin
Clrscr ;
Write (’Chon:’) ;
Readln (n);
For i := 1 to n do
Begin
Write (’Cho x’ , i, * , y*, i, V) ; Readln (x [i], y [i]) ; tt [i] := False ; end ;
Write (’Diem xuat phat:’) ;
Readln (Source) ;
Write (’Diem ket thuc:’) ;
Readln (Target) ;
Write (’Cho m:’) ;
Readln (m) ;
t := 1 ; Dao [1] := Source ; tt [1] := True ;
Sl := 0 ; min := maxint ; Tot := False ;
End ;
Functi0n dd (i, j : integer) : Real ;
(Tính khoảng cách giữa 2 đảo i, j)
Begin
dd := sqrt (sqr (x [i] — x [j]) + sqr) y [i] ;
End ;
Procedure Print ;
Var j : interger ;
Begin
```

```
Tot := True ;
Writeln (‘Ton tai duong di ;’) ;
For j := 1 to t do
Write (‘M’, Dao [1], ’ =>’) ;
Writeln ;
End ;
Procedure Printmin ;
Var j : integer ;
Begin
Writeln ;
Writeln (’Duong di toi uu:’) ;
For j := 1 to t1 do
Write (‘M’, Dao1 [j], ’=>’) ;
Writeln ;
Writeln (’So Ian ghe qua:’ , Min - 1) ;
End ;
Procedure Findmin ;
Begin
If min > Sl then
Begin
Min := Sl ;
t1 := t ;
Dao1 : = Dao ;
End ;
End ;
Procedure Try (i : integer) ;
Var j : integer ;
Begin
If j := Target then
Begin
Print ;
Findmin ;
End ;
Else
For j := 1 to n do
If n0t (tt UP and (m > KC (i, j))
```

```
then
Begin
t := t + 1 ;
Dao [t] := j ;
tt [j] := True ;
Sl := Sl + 1 ;
Try (j) ;
Sl := Sl — 1 ;
tt [j] := False ;
t := t - 1 ;
End ;
End ;
{Chương trình chính}
Begin
Init ;
Try (Source) ;
If tot then Printmin
Else
Writeln (’Khong co duong di’) ;
End.
```

7. Các bài toán tìm kiếm cơ bản

Bài 1 : Sơ đồ đường đi 2 chiều giữa N thành phố được cho bởi ma trận A[i, j] trong đó A [i,j] = 1 nếu có đường đi giữa thành phố i tới j và A [i, j] = 0 nếu không có đường đi.Hãy lập trình tìm mọi đường đi khác nhau giữa hai thành phố P và Q.

Dữ liệu: cho trong file INP.BL1

+ Dòng đầu ghi số n là số đỉnh của đồ thị (0 < n < 100)

+ Dòng i+1(1 =< i =< n) chứa n số A [i, 1] A [i, 2] ... A [i, n] }

+ Dòng cuối ghi 2 số p và Q

Kết quả: xuất ra màn hình.

Ví Dụ:

Kết quả :

Lời giải thứ 1: 1- 2- 3- 4- 5

Lời giải thứ 2 : 1- 2- 5

Lời giải thứ 3 : 1- 4- 3- 2- 5

Lời giải thứ 4 : 1- 4 - 5

Lời giải thứ 5 : 1 - 5

INF , BL1

6

0 1 0 1 1 0

1 0 1 0 1 0

0 1 0 1 0 0

1 0 1 0 1 0

1 1 0 1 0 0

0 0 0 0 0 0

1 6

Kết quả:

Không có đường đi từ 1 đến 6

HƯỚNG DẪN GIẢI THUẬT:

Tìm kiếm theo chiều sâu:

+ Đỉnh đầu tiên là đỉnh p

+ Từ một đỉnh i mới tới ta lần lượt thử mọi đường đi từ i đến các đỉnh j còn lại (A [i, j] > 0 và đỉnh j chưa đến trước đó). Mỗi lần thử đường đi từ i đến j ta xóa bỏ cạnh A [i, j] và A [j, i] (sau đó cần phục hồi 2 cạnh này lại)

+ Nếu đến được đỉnh Q thì xuất kết quả.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_ 1:
Const Max = 100 ;
Var n, i, j, p, Q  : Byte ;
A : Array [1.. Max, 1 .. Max] of Byte
L : Array [1.. Max] of Byte ;
K : Array [1.. Max] of Boolean ;
Jem : Long Int ;
Procedure Nhap ;
Var  f : Text ;
Begin
dem := 0 ;
Assign (f,’ INP.BL1’) ;
Reset (f) ;
Readln (f, n) ;
FillChar (A, SizeOf (A) , 0) ;
For i := 1 to n do
Begin
```

```
For j := 1 to n do
Read (f, A [i, j]) ;
Readln (f) ;
End ;
Readln (f, p, Q) ;
Close (f) ;
FillChar (L, SizeOf (L), 0) ;
L [1] := p ;
Fillchar (K, SizeOf (K), 1) ;
K [P] := False ;
End ;
Procedure Xuat (VT : Byte) ;
Var i : Byte ;
Begin
Inc (dem) ;
Write (‘Loi giai thu’ , dem, ‘:’ , p, ”) ;
For i := 2 to VT do Write (‘ ->’, L [i], ”) ;
Writeln ;
End ;
Procedure Tim (VT : byte) ;
Var  i : Byte ;
Begin
If L [VT - 1] = Q then Xuat (VT - 1)
Else
For i := 1 to N do
If (A [L [VT - 1], i] > 0) and (K [i]) then
Begin
K [i] := False ;
L [VT] := i ;
A [L [VT - 1], i] := 0 ;
A [i, L [VT - 1]] := 0 ;
Tim (VT + 1) ;
L [VT] := 0 ;
A [L [VT - 1], i] := 1 ;
A [i, L [VT - 1]] := 1 ;
K [i] := True ;
```

End ;

End ;

BEGIN

Nhap ;

Tim (2) ;

If dem = 0 then Writeln (’Khong co duong di tu ’,P,’ den ‘, Q);

END.

BÀI 2:

Hình 1 là mội bảng tam giác các số nguyên không âm.

Hãy viết chương trình để tinh tổng lớn nhất các số trên đường đi từ itinh tam giác và kết thúc tại một điểm nào đó ở đáy tam giác.

+ Mỗi nước đi ta được quyền đi thẳng xuống bên trái hay bên phải của số đó.

+ Số hàng trong tam giác lờn hơn 1 và =< 100

+ Các số trong tam giác đều là số nguyên không âm và nhỏ hơn 100

- Dữ liệu : cho trong file INP.BL2

+ Dòng đầu ghi số lượng các dòng trong tam giác (N)

+ Dòng i+1 (1 =< i =< N) ghi i số

Vi dụ :

5

7

3 8

7 1 0

2 7 4 4

4 5 2 6 5

Kết quả : Xuất ra màn hình tổng lớn nhát tìm được

VÍ DỤ:

INP.BP2

5

7

3 8

7 1 0

2 7 4 4

4 5 2 6 5

Kết quả 30

10

7

4

6

3 8

8 1 0

2 7 4 4

4 5 2 6 5

3 4 8 6 3 5

3 5 1 5 3 7 8

3 5 7 1 7 8 3 7

8 6 5 3 1 4 5 7 8

3 5 1 6 7 4 3 2 1 3

Kết quả 58

HƯỚNG DẪN GIẢI THUẬT:

Tìm kiếm theo chiều rộng:

— Mảng B [i, j] là độ dài đường đi dài nhất từ (1, 1) đến (i, j)

— Nhận xét : từ 6 (i, j) tá có thể đến 6 (i+1, j) hay (i+1, j+l)

— For i := 2 to N do

For j := l to i — 1 do

For k := 0 to 1 do

Nếu (B [i, j + k] = $ FFFF)

hay (B [i — 1, j] + A [i, j + k] > B [i, j + kl) thì

B [i, j + k] := B [i — 1, j] + A [i, j + k] ;

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_2 ;
Var  N, i, j, k : Byte ;
A : Array [1.. 100, 1.. 100] of Byte ;
B : Array [1.. 100, 1.. 100] of Word ;
Procedure Nhap ;
Var  f : Text ;
Begin
Assign (f,' INP.BL2') ;
Reset (f) ;
Readln (f, N) ;
FillChar (A, SizeOf (A), 0) ;
For i := 1 to N do
Begin
```

```
For j := 1 to i do
Read (f, A [i, j]) ;
Readln (f) ;
End ;
FillChar (B, SizeOf (B), $ FF) ;
Close (f) ;
End ;
Procedure Xuly ;
Begin
B [1, 11 := A [1, 1] ;
For i := 2 to N do
Begin
For j := 1 to i — 1 do
Begin
If (B [i, j + k] = $ FFFF)
or (B [i — 1, j] + A [i, j + k] > B [i, j + k]) then
B [i, j + k] := B [i - 1, j] + A [i, j + k] ;
End ;
j := 0 ;
For i := 1 to N do
If j < B [N, i] then j := B [N, i] ;
Writeln (j) ;
End ;
BEGIN
Nhap ;
Xu ly ;
END.
```

Bài 3. Cho ma trận A kích thước 3x3 .

Hãy điền các số từ 1 đến 9 vào ma trận sao cho trên mỗi hàng đều là một số nguyên tố. (Tìm mọi trường hợp). Kết quả xuất ra màn hình.

(Không dùng đệ quy)

HƯỚNG DẪN GIẢI THUẬT:

— Sử dụng 9 vòng For lồng nhau, ta lần lượt có được tất cả các bộ phận hoán vị của 9 phần tử từ 1 đến 9. Với mỗi bộ kết quả này, ta kiểm tra xem các số tạo thành có là số nguyên tố hay không và xuất kết quả.

Có tất cả 816 lời giải khác nhau.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_3 ;
Uses CRT ;
Var
k : Byte ;  ,
A : Array [1.. 8] of Byte ;
B : Array [1.. 9] of Boolean ;
i : Array [1.. 9] of Byte ;
x, y : Integer ;
C : Array [100.. 999] of Boolean ;
dem : Integer ;
Functi0n KT (i : Word) : Boolean ;
Var  j : Word;
Begin
KT:= False ;
For j := 2 to Trunc (Sqrt (i)) do
If i mod j = 0 then Exit ;
KT True ;
End ;
Procedure NguyenTo ;
Var  i : Integer ;
Begin
For i := 100 to 999 do
C [i] := KT (i) ;
End ;
BEGIN
dem := 0 ;
Clrscr ;
FillChar (A, SizeOf (A), 0) ;
FillChar (B, SizeOf (B), 1) ;
NguyenTo ;
For i [1] := 1 to 9 do
If B [i [1]] then
Begin
B [i[1]] := False ;
For i[2] := 1 to 9 do
```

```
If B [i[2]] then
Begin
B [i[2]] := False ;
For i[3] := 1 to 9 do
If B [i[3]] then
Begin
B [i[3]] := False ;
If C [i[l] * 100 + i [2] * 10 + i [3]] then
For i[4] := 1 to 9 do
If B fi[4]] then
Begin
B [i[4]] := False ;
For i[5] := 1 to 9 do
If B [i[5]] then
Begin
B [i[5]] := False ;
For i[6] := 1 to 9 do
If B [i[6]] then
Begin
B [i[6]] := False ;
If C [i [4] * 100 + i [5] * 10 + i [6]] then
For i[7] := 1 to 9 do
If B [i[7]] then
Begin
B [i[7]J := False ;
For i[8] := 1 to 9 do
If B [i[8]] then
Begin
B [i[8]] := False ;
For i[9] := 1 to 9 do
If B [i[9]] then Begin
B [i[9]] := False ;
If C [i[7] * 100 + i [8] * 10 + i [9]] then
Begin
Inc (dem) ;
Writeln (’Loi giai thu’, dem, :’) ;
```

```
Writeln (i[I],’ ‘, i [2], ’ ‘, i [3] ) ;
Writeln (i[4],’ ’, i [5], ’ ‘, i [6]) ;
Writeln (i[7],’ ‘, i [8], ’ ‘, i [9]) ;
End ;
B [i[9]] := True ;
End ;
B [i[8]] := True ;
End ;
B [i[7]] := True ;
End ;
B [i[6]] := True ;
End ;
B [i[5]] := True ;
End ;
B [i[4]] := True ;
End ;
B [i[3]] := True ;
End ;
B [i[2]] := True ;
End ;
B [i[1]] := True ;
End ;
END.
```

Bàl 4 : Hai người chơi trò chơi như sau : Người thứ nhất sẽ nghĩ ra một số nguyên dương trong khoảng từ 1 đến N (N được cho biết trước). Người thứ hai sẽ lần lượt đưa ra các số dự đoán. Vời mỗi số dự đoán này, người thứ hai sẽ nhận được câu trả lời cho biết số mình vừa nêu ra lớn hơn, nhỏ hơn hay bằng với số mà người thứ nhất đã nghĩ. Bạn hãy giúp người thứ hai chọn đúng số cần tìm với số lần đoán càng it càng tốt.

VÍ DỤ:

N = 10

Số cần tìm là 4

Lần đoán thứ 1 : 5

1 — Số bạn nghĩ nhỏ hơn

2 — Số bạn nghĩ lớn hơn

3 — Chính xác

Trả lời : 1

Lần đoán thứ 2 : 3

1 — Số bạn nghĩ nhỏ hơn

2 — Số bạn nghĩ lớn hơn

3 — Chính xác Trả lời : 2

Lần đoán thứ 3 : 4

1 — Số bạn nghĩ nhỏ hơn

2 — Số bạn nghĩ lớn hơn

3 — Chính xác Trả lời : 3

Số bạn nghĩ là 4

N = 8

Số cần tìm là 7

Lần đoán thứ 1 : 4

1 — Số bạn nghĩ nhỏ hơn

2 — Số bạn nghĩ lớn hơn

3 — Chính xác Trả lời : 2

Lần đoán thứ 2 : 6

1 — Số bạn nghĩ nhỏ hơn

2 — Số bạn nghĩ lớn hơn

3 — Chính xác Trả lời : 2

Lần đoán thứ 3 : 7

1 — Số bạn nghĩ nhỏ hơn

2 — Số bạn nghĩ lớn hơn

3 — Chính xác

Trả lời : 3

Số bạn nghĩ là 7

HUỚNG DẪN GIẢI THUẬT

Tìm kiếm nhị phân

1. x := 1; y := N ; a := 0

2. a := (y + x) div 2

3. Lần đoán thứ i : a

4. Nếu a lớn hơn số cần tìm thì gán y := a

Nếu a nhỏ hơn số cần tìm thì gán x := a

5. Nếu số lần đoán vượt quá $\log_2 N$ thì chấm dứt

Ngược lại thì trở lại bước 2

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_4;
UsesCRT;
Const MaxN = 10000;
Var  N : Integer;
x, y, a : Integer;
dem : Integer;
Traloi : Byte;
BEGIN
Clrcsr;
Write ('Xin cho biet gia tri N = ’);
Readln (N);
x := 1; y := N ; a := 0 ; dem := 1;
Repeat
Write (’Lan đoan thu’, dem, ’ :’);
a := (y + x) div 2;
Writeln (a)
Writeln (’ 1 - So ban nghi nho hon’);
Writeln (’ 2 - So ban nghi lon hon’);
Writen (’ 3 - Chinh xac’);
Write (’ Traloi:’);
Readln (traloi);
Case Traloi of
1 : y := a;
2 : X := a;
End;
If dem mod 4 = 0 then Clrscr,
Inc (dem);
Until (dem > In (N) / In (2) + 1) or (traloi = 3);
If traloi <> 3 then Writeln (‘ Ban da thang')
Else Writeln (‘ So ban nghi la’, a);
Readln;
END.
```

Bài 5 : Bạn nhận được một túi chứa 3N đồng tiến vàng. Bạn biết rằng trong túi này có chứa một đồng tiền giả nhẹ hơn tiền thật. Với một chiếc cân có hai đĩa cân, bạn hãy tìm ra đồng tiền giả với số lần cân ít nhất.

Giả sử mỗi đồng tiền đều được đánh số từ 1 đến 3N

Mỗi lần cồn, bạn thòng báo ra màn hỉnh những đồng tiền nòm trên đĩa cân bên phải, dĩa cân bên trái và sau đó bạn sẽ nhận được câu trả lời về tình trạng cân : "L" nếu càn lệch vế bèn trái, "R" nếu cân lệch về bên phải, "B" nếu cân thăng bằng.

Giới hạn số lần cân không quá N lần.

VÍ DỤ:

N = 3, đồng tiền giả : 6

Lần cân 1

Dĩa cân bên phải:

1

2

3

4

5

6

7

8

9

Dĩa cân bên trái:

10

11

12

13

14

15

16

17

18

L

Lần cân 2

Dĩa cân bên phải:

1

2

3

Dĩa cân bên trái:

4

5

6

R

Lần cân 3

Dĩa cân bên phải : 4

Dĩa cân bên trái : 5

B

Đồng tiền giả : 6

Ví dụ 2:

N = 3, đồng tiền giả : 27

Lần cân 1

Dĩa câm bên phải: 1

2

3

4

5

6

7

8

9

Dĩa cân bên trái: 10

11

12

13

14

15

16

17

18

B

Lần cân 2

Dĩa cân bên phải:

10

20

21

Dĩa cân bên trái:

22

23

24

B

Lần cân 3

Dĩa cân bên phải : 25

Dĩa cân bên trái : 26

B

Đồrig tiền giả : 27

HƯỚNG DẪN GIẢI THUẬT

Chia số tiền làm 3 nhóm bằng nhau. Mỗi lần cân 2 nhóm (giả sử là nhóm 1 và 2).

Nếu cân không thăng bằng thì xét trò lại với nhóm tiền nhẹ hơn

Nếu cân thăng bằng thì xét trở lại với nhóm tiền thứ 3.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_25;
Uses CRT;
Var N, A, X, y, i, j, dem : LongInt;
T : Array [1..3, 1..2] of Byte;
Traloi : Char,
BEGIN
Clrscr;
Write (’Cho biet N = ’);
Headln (N);
A := Trunc (exp (N*ln (3)));
x := 1; y := A;
dem := 0;
Repeat
Inc (dem);
Writeln (’Lan thu’, dem);
T[1,1] := x; T[1,2] := X + (y - x) div 3;
T[2, 1] := T[1, 2] + 1 ; T[2,2] := X + 2* (y - x) div 3;
Write (’Dia can ben phai :’);
For I := T[1,1] to T[1,2] do Write (i, ’ ’);
Writeln;
Write (’Dia can ben trai :’);
```

```
For i := T[2,1] to T[2,2] do Write (i, ' ');
Writeln;
Readln (Traloi);
Traloi := upease (Traloi);
Case Traloi of
'L' : j := 1;
'R' : j := 2;
'B' : j := 3;
End;
x := T[j, 1];
y := T[j,2];
Until x = y;
Writeln ('Dong tien gia :', x);
Readln;
END.
```

Bài 6: Hãy tìm và in ra màn hình tất cả các hoán vị của N số nguyên dương đầu tiên nhưng không được sử dụng đệ quy.

Ví dụ

n = 2

Hoán vị thứ 1 : 1, 2

Hoán vị thứ 2: 2, 1

n = 3

Hoán vị thứ 1: 1, 2, 3

Hoán vị thứ 2: 1, 3 , 2

Hoán vị thứ 3: 2, 1, 3

Hoán vị thứ 4: 2, 3, 1

Hoán vị thứ 5: 3, 1, 2

Hoán vị thứ 6: 3, 2, 1

HƯỚNG DẪN GIẢI THUẬT:

1. Gọi A[i] là giá trị của phần tử thứ i của hoán vị
2. Hoán vị đầu tiên là 1 2 3 4 5... n
3. Tìm giá trị i_0 lớn nhất sao cho $A[i_0] < A[i_0+1]$
4. Tìm giá trị j_0 lớn nhất sao cho $A[i_0]<A[j_0]$
5. Đổi chỗ $A[i_0]$ và $A[j_0]$
6. Dùng phép chiếu gương để viết ngược tất cả các giá trị A[i] với $i_0 < i =< N$
7. Xuất dãy A[i] ra màn hình.

8. Nếu không tồn tại i_0 ở bước 2 thì chấm dứt chương trình.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_6;
Uses CRT;
Var  N, i, j, i0, j0, k : Byte;
A, B : Array [1...100] of Byte;
dem : LongInt;
xong : Boolean;
BEGIN
Clrscr;
Write (’N =’); Readln (N);
For i := 1 to N do A[i] := i;
dem := 1;
Write (’Hoan vi thu’, dem/:’);
For i := 1 to N do Write (A[i],’ ‘)’
Writeln;
Repeat
xong := True;
For i := N - 1 downto 1 do
If A[i] < A[i + 1] then
Begin
i0 := i; j0 := 0;
For j := N down to i + 1 do
If (j0= 0) and (A[i0] < A[j]) then
Begin
j0 := j;
j := i + 1;
End;
xong := False;
i := 1;
End;
If n0t xong then
Begin
k := A[i0];
A[i0] := A[j0];
A[j0] := k;
```

```
Move (A, B, i0);
For i := i0 + 1 to N do
B[N - 1 + 10 + 1] := A[i];
Move (B, A, SizeOf (B));
Inc (dem);
Write ('Hoan vi thu', dem,' : ');
For i := 1 to N do Write (A[i],' ');
Writeln;
End;
Until xong;
END.
```

Bài 7. Tìm tất cả các số nguyên tố X(10=< x =<65535) thỏa mãn :

+ X là số nguyên tổ

+ X là số Fibonacci

+ Ít nhát có một số nguyên tố được tạo thành khi thay đổi vị trí các chữ số của X

Kết quả : xuất ra file OUT.BL7 Mỗi dòng gồm số X và sô Xn tạo thành từ số X.

HƯỚNG DẪN GIẢI THUẬT

- Tìm tất cả các số Fibonacci nhỏ hơn 65535 (25 số)
- Với mỗi số Fibonacci, kiểm tra số này có là số nguyên tố hay không
- Nếu là số nguyên tố thì tìm mọi hoán vị của các chữ số trong số này, nếu tồn tại một hoán vị tạo thành số nguyên tố, thì in kết quả

Có tất cả 3 số thỏa đề bài.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_7;
Var f : Text;
X, Y : Word;
S : String;
A : Array[1 .. 5] of Char;
B : Array[1 .. 5] of Boolean;
Fib : Artfay [1 .. 25] of LongInt;
Fib_Num : Integer;
xong : Boolean;
Functi0n   NguyenTo (X : Word) : Boolean;
Var  i : Word;
```

```
Begin
NguyenTo := False;
For i := 2 to Trunc (sqrt (X)) do
If X mod i = 0 then Exit;
NguyenTo := True;
End;
Procedure Test;
Var  Err : Integer;
S1 : String;
Begin
S1 := ’‘;
For Err := 1 to Length (S) do
S1 := S1 + A[Err];
If S1 = S then Exit;
Val (S1, Y, Err);
If NguyenTo (Y) then xong := True;
End;
Procedure Tim (VT : Byte);
Var  i : Byte;
Begin
If n0t xong then
If Vt = Length (S) + 1 then Test
Else
For i := 1 to Length (S) do
If n0t B[i] then
Begin
B[i] := True;
A[VT] := S[i];
Tim (VT + 1);
B[i] := False;
A[VT] := #0
End;
End;
Functi0n Doicho (X : Word) : Boolean;
Begin
xong := False;
```

```
Str (X,S)
FillChar (A, SizeOf (A), 0);
FillChar (B, SizeOf (B), 0);
Tim (1);
Doicho := xong;
End;
Functi0n KiemTra (X : Word) : Boolean;
Begin
KiemTra := False;
If n0t NguyenTo (X) then Exit;
If n0t Doicho (X) then Exit;
KiemTra := True;
End;
Procedure Fibonacci;
Begin
Fib_Num := 2;
FillChar (Fib, SizeOf (Fib), 0);
Fib[1] := 1;
Fib[2] := 1;
Repeat
Inc (Fib_Num);
Fib [Fib_Num] := Fib[Fib_Num-1] + Fib[Fib_Num-2];
Until Fib[Fib_Num-1] + Fib[FibNum] >= 65535;
End;
BEGIN
Assign (f,'OUT.BL?');
Rewrite (f);
Fibonacci;
For x := 8 to Fib Num do
If KiemTra (Fib[X]) then
Writeln (f, Fib[X],' `, Y);
Close (f);
END.
```

Bài 8 : Cho một đồ thị vô hướng được biểu diễn bằng ma trận kề A[i, j] trong đó A[i,j] = 1 nếu có đường đi trực tiếp giữa i tới j và A[i,j] = 0 nếu không có

đường đi trực tiếp giữa i và j. Hãy tìm mọi chu trình trong đồ thị này và xuất kết quả ra màn hình. Dữ liệu cho trong file INP.BL9 gồm n + 1 dòng:

+ Dòng 1 : ghi số n là số lượng đỉnh của đồ thị

+ Dòng i + 1 (l<i<n) ghi n số A[i,1] A[i,2] ... A[i,n]

Các số ghi trên cùng một dòng ghi cách nhau ít nhát 1 dấu cách

Ví DỤ:

HƯỚNG DẪN GIẢI THUẬT:

— Đỉnh đầu tiên của chu trình lần lượt là 1, 2, N — 1

— Với mỗi đỉnh đầu tiên này ta tìm mọi đường đi có thể có từ nó đến các đỉnh còn lại và cứ tiếp tục như vậy cho đến khi gặp lại đỉnh đầu tiên (Trên đường đi ta chỉ chọn rao đỉnh tiếp theo có số thứ tự lớn hơn đỉnh đã chọn trước đó hay trùng với đỉnh đầu tiên để bảo đảm không chọn trùng lại chu trình.

(Ví dụ 1- 2- 1 cũng là 2- 1- 2)

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_8 ;
Uses CRT;
Var  A : Array[1 .. 10,1 .. 10] of Byte;
W : Array[1 .. 20] of Byte;
n, i : Byte;
dem : Integer;
Procedure Nhap;
Var  f : Text;
i , j : Byte;
Begin
dem := 0;
Assign (f, ` INP.BL8’);
Reset (f);
Readln (f,n);
FillChar (A,SizeOf (A), 0);
For i := 1 to n do Begin
For j := 1 to n do Read (f, A[i, j]);
Readln (f);
End;
Close (f);
End;
Procedure Xuat (VT : Byte);
```

```
Begin
Inc (dem);
Write (’Loi giai thu’, dem,’ : ’);
For i := 1 to VT do
Write (W[i], ‘ => ‘);
Writeln (W[1]);
End;
Procedure Tim (VT : Byte);
Var  i : Byte;
Begin
If <W[VT - 1] = W[l] and (VT > 2) then Xuat (VT - 2)
Else
For i := to N do
If A[W[VT-1], i] > 0 then
If (i = W[1]) or (i > W[VT - 1]) then
Begin
A[W[VT - 1], i] := 0;
A[i, W[VT -1]] := 0;
W[VT] := i;
Tim (VT + 1);
W[VT] := 0;
A[W[VT - 1], i] := 1;
A[i,W[VT - 1]] ;= 1;
End;
End;
Procedure Xuly;
Var  i : Byte;
Begin
FillChar (W,SizeOf (W), 0);
For i := 1 to N - 1 do
Begin
W[1] := i;
Tim (2);
End;
End;
BEGIN
```

Nhap;

Xuly;

END.

Bài 9: Bạn là người đoạt giải trong một cuộc thi do Hãng hàng không CANADA tổ chức. Giải tHƯỚNG là một chuyến du lịch không mất tiền vòng quanh Canada bằng máy bay của hãng. Hành trình chỉ được đi từ Tây sang Đông cho đến khi gặp thành phố ở phía Đông nhất rồi quay trở lại theo hướng từ Đông sang Tây cho đến khi gặp lại thành phố xuất phát. Không được thăm một thành phố nào quá một lần ngoại trừ thành phố xuất phát mà bạn phải thăm đúng hai lần (một lần khi xuất phát và một lần khi kết thúc hành trình). Bạn không được sử dụng máy bay của hãng khác hay bất cứ phương tiện giao thông nào khác.

Bài toán phải giải là : Cho một danh sách các thành phố và một danh sách các tuyến bay trực tiếp giữa hai thành phố. Hãy vạch ra hành trình thỏa mãn các điều kiện nêu trên và cho phép bạn thăm được càng nhiều thành phố càng tốt.

Dữ liệu : Cho trong file INP . BL9

+ Dòng đầu : chứa số N là số lượng thành phố và V là số lượng các tuyến bay trực tiếp.

+ Mỗi dòng trong N dòng tiếp theo : Tên thành phố. Tên thành phố được sắp theo thứ tự từ Tây sang Đông nghĩa là thành phố thứ i sẽ là thành phố phía Đông của thành phố j khi và chỉ khi i > j (không có hai thành phố nào cùng ở trên một kinh tuyến). Tên của mỗi thành phố là một xâu nhiều nhất là 15 kí tự, mỗi kí tự là chữ số hoặc chữ cái Latonh, ví dụ : AGR34 hoặc BEL4

+ Mỗi dòng trong V dòng tiếp theo : tên của hai thành phố có trong danh sách tên các thanh phố, tên hai thành phố đó cách nhau bằng một dấu cách. Nếu cặp CITY1 CITY2 xuất hiện ở một dòng nào đó thì có nghĩa là tồn tại tuyến bay trực tiếp từ CITY1 đến CITY2 và đồng thời cũng có tuyến bay trực tiếp từ CITY2 đến CITY1.

Kết quả : ghi ra file OUT.BL9 theo quy định 4 Dòng đầu là số lượng các thành phố trong tốp dữ liệu vào

+ Dòng 2 là số lượng M các thành phố khác nhau mà hành trình đi qua

+ M + 1 dòng tiếp theo mỗi dòng ghi tên một thành phố theo đúng trình tự đi của hành trình.

Nếu kết quả là vô nghiêm thì trong file kết quả chỉ ghi 2 dòng

+ Dòng đầu là số lượng các thành phố trong tập dữ liệu vào

+ Dòng 2 ghi thòng báo “NO SOLUTI0N”

VÍ DỤ:

INP.BL9
=> OUT.BL9
8 9
=> 8
Vancouver
=> 7
Yellowknife
Vancouver
Edmonton
Edmonton
Calgary
Montreal
Winnipeg
Halifax
Toronto
Toronto
Montreal
Winnipeg
Halifax
Calgary
Vancouver
Edmonton
Vancouver
Calgary
Calgary
Winnipeg
Winnipeg
Toronto
Toronto
Halifax
Montreal
Halifax
Vancouver
Edmonton
Montreal
Edmonton

Yellowknife

Edmonton

Calgary

INP.BL9

=> OUT.BL9

5 5

=>5

C1

=> n0 SOLUTI0N

C2

C3

C4

C5

C5

C4

C2

C3

C3

C1

C4

C1

C5

C4

HƯỚNG DẪN GIẢI THUẬT:

- Thủ tục đệ quy Solve (Count, Kind : Byte):

Kind = 1 nếu trên đường đi từ Tây sang Đông

Kind = 2 nêu trên đường đi từ Đông sang Tây

- Count : Số thành phố đã qua.

- Nếu thời gian thực hiện quá MaxTime (giây) thì xuất kết quả.

- Nếu Count + Số thành phố chưa đến < số lượng nhiều nhất các thành phố trong kết quả trước đó thì không thực hiện.

- Nếu Kind = 1 thì tìm đường đi từ thành phố hiện giờ đến các thành phố có số thứ tự lớn hơn chưa đến

- Nếu Kind = 2 thì tìm đường đi từ thành phố hiện giờ đến các thành phố có số thứ tự nhỏ hơn chưa đến. (trừ thành phố xuất phát)

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_9;
UsesCrt;
Const
MaxCity = 100;
Max Time = 20;
Var  n,k  : Byte;
Way, Save Way : Array [1..MaxCity] of Byte;
Tp : Array[1..MaxCity] of String;
A: Array[1..MaxCity, 1.. MaxCity] of Boolean;
Ct : Array[1..MaxCity] of Boolean;
Max,p : Byte;
T : LongInt Absolute 0 : $46C;
T1, T2 : LongInt;
Procedure InputData;
Var f : Text;
i, x, y, j : Byte;
s1 , s2 , s  : String;
FileName  : String;
Begin
FillChar (Ct, SizeOf (Ct), True);
FillChar (Way , SizeOf (Way), 0);
FillChar (A, SizeOf (A), False);
Clrscr;
Filename := 'INP.BL9';
Assign (f , FileName);
Reset (f);
If I0Result <> 0 then
Begin
Writeln ('Cannot open file', FileName);
Readln;
Halt ;
End;
Headin (f,n,k);
For i := 1 to n do
Readln (f,TP[i]);
For i := 1 to n do
```

```
Begin
Readln (f,s);
j := 0;
s1 := ’‘,
s2 := ’‘,
Repeat
Inc (j);
s1 := s1 + s|j];
Until s[j] = ’‘,
Delete (s1 , Length (s1), 1);
s2 := copy (s , j + 1 , Length (s) - j);
For j := 1 to n do
If s1 = Tp [j] then x := j;
For j := to n do
If s2 = Tp[j] then y := j;
A[x,y] := True;
A[y,x] := True;
End;
Close (f) ;
Max := 0 ;
FillChar (SaveWay, SizeOf (SaveWay), 0); End;
Procedure Test;
Var  CCity:Byte;
Begin
CCity := 1;
While Way[CCity] > 0 do Inc (CCity);
Dec (CCity);
If CCity > Max then
Begin
Max := CCity;
Save Way := Way;
End;
End;
Procedure OutPut;
Var  f: Text;
Begin
```

```
Assign (f, 'OUT.BL9');
Rewrite (f);
If Max = 0 then
Begin
Writeln (f,n);
Writeln (f, 'NO SOLUTI0N');
End
Else
Begin
Writeln (f,n);
Writeln (f,Max -1);
For p := 1 to Max do
Writeln (f, Tp[SaveWay[p]])
End;
Close (f);
Halt;
End;
Procedure Solve (Count, Kind : Byte);
Var  i : Byte;
NumCityLeft : Byte;
Begin
If T –T1 > MaxTime * 18.2 then OutPut;
If Count < Max then
Begin
NumCityLeft := 0;
For i := 1 to n do
If Ct[i] then Inc (NumCityLeft);
End;
If Count + NumCityLeft >= Max then Bogin
If Kind = 1 then Begin
For i := Way [Count -1] + 1 to n do
If (Ct[i]) and (A[i, Way [Count -1]]) then
Begin
Way [Count] := 1;
Ct[i] := False;
A[i, Way[Count —1]] := False;
```

```
A[Way[Count -1], i] := False;
If i < n then Solve (Count + 1,1)
Else Solve (Count + 1, 2);
Way [Count] := 0;
Ct[i] := True;
A[i,Way[Count -1]] := True;
A[Way[Count —1] , i] := True;
End;
End
Else
Begin
For i := Way[Count —1] —1 Down to 1 do
If(Ct[i]) and (A[i,Way[count -1]]) then
Begin
Way[Count] := i;
Ct[i] := False;
A[i,Way[Count —1]] : = False;
A[Way [Count -1] , i] := False;
If i > 1 then Solve (Count + 1,2)
Else Test;
WayfCount] := 0;
Ct[i] := True;
A[i,Way[Count -1]] := True;
A[Way[Count -1]] , i] := True;
End;
End;
End;
End;
BEGIN
T1 := T;
Clrscr,
InputData;
Way[1] := 1;
Solve (2, 1);
OutPut;
END.
```

Bài 10 : File ASCII INP.BLO chứa N số nguyên (N không cho biết trước, N =< 600000)

Các số cách nhau ít nhất một dấu cách hoặc xuống dòng

Số nguyên X thuộc file được gọi là số chính nếu nó xuất hiện trong file hơn N/2 lần

Hãy xác định file đã cho có số chính hay không và thông báo ra màn hình kết quả kiểm tra cũng như bản thân số chỉnh nếu tồn tại.

Yêu cầu kĩ thuật : Với N < 100000 thời gian tính toán không qua một phút, với N > 100000 thời gian tính toán không quá 6 phút.

VÍ DỤ:

INP.BLO

40001

40000

40001

40000

40001

40000

40000

40001

40000

40000

40001

- Kết quả

Số chính : 40001

INP.BLO

502

951

802

428

527

702

304

870

635

747

- Kết quả

Không có số chính

HƯỚNG DẪN GIẢI THUẬT:

- Gọi mảng dem[i,j] là số lượng các số trong N số đã cho có bit thứ j mang giá trị i(i = 0, 1 ; 1 =< j =<16)

- Nếu Min {Max (dem[0j] demi[1,j]} | 1 =< j =< 16} > N/2 thì

Gọi P|j] = 1 nếu dem[1j] > dem[0j] và ngược lại

Tìm trong file số có bit j bàng P[j] và xuất hiện quá N/2 lần Nếu có thì đó là số chính cần tìm

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_2_ 10;
Uses CRT;
Const A:Array[1.. 16] of Word = (1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 20 4096, 8192, 16384, 32768);
Var  f : Text;
Filename : String;
n, X, y, t, z, saved : Longint;
dem : Array [0.. 1,1..16] of Longint;
i : Byte;
Begin
Clrscr;
Assign (f, 'INP.BL0');
Reset (f);
FillChar (Dem,SizeOf (Dem), 0);
n := 0;
While n0t (EOF(f)) do
Begin Inc (n);
Readln (f,x);
For i := 1 to 16 do
Begin
If x and A[j] = 0 then Inc (Dem[0,i])
Else Inc (Dem[1,i]);
End;
End;
Close (f);
Assign (f, 'INP.BL1');
Reset (f);
```

```
t := 0;
For y := 1 to n do
Begin
Readln (f,x); z := 0;
For i := 1 to 16 do
If dem[(x and A[i]) div A[i], i] >
dem[(((x and A[i]) div A[i]) + I) mod 2,i] then
Inc (z)
Else
If dem[0,i] = dem[l,i] then
Begin
Writeln (’Không co so chinh’);
Close (f);
Halt;
End;
If z = 16 then
Begin
Inc (t);
Saved := x;
End;
End;
Close (f);
If t > n div 2 then
Writeln (’So chinh : ‘, Saved)
Else
Writeln (’Không co so chinh’);
Readln;
End.
```

§ 3. SẮP XẾP

1. Khái niệm

Sắp xếp là một quá trình tổ chức lại một dãy các dữ liệu theo một trật tự nhất định.

Mục đích của việc sắp xếp là nhằm giúp cho việc tìm kiếm dữ liệu được dễ dàng và nhanh chóng. Sắp xếp là một việc làm hết sức cơ bản và được dùng rộng rãi trong các kĩ thuật lập trình nhằm xử lí một dáy các dữ liệu.

2. Phân loại

Các giải thuật sắp xếp được phân thành hai nhóm chính là:

• Sắp xếp trong (Hay sắp xếp mảng).

• Sắp xếp ngoài (hay sắp xếp tập tin).

3. Sắp xếp trong

Toàn bộ dữ liệu cần sắp xếp phải được đưa vào bộ nhớ chính của máy tính do đó nó thường được sử dụng khi khối lượng dữ liệu không vượt quá dung lượng bộ nhớ chính.

Phương pháp sắp xếp trong thường dùng các kĩ thuật sau

• Đếm số phần tử

• Chèn thêm phần tử

• Chọn phần tử

• Đổi chỗ hai phần tử

• Trộn hai dãy các phán tử

4. Sắp xếp ngoài

Áp dụng trong trường hợp ta phải sắp xếp các tập tin chứa nhiều mấu tin và mỗi mẫu tin có chiều dài tương đối lớn do đó ta không thể nạp toàn bộ tập tin này vào bộ nhớ chính để sắp thứ tự. Vì vậy ta phải có những phương pháp thích hợp cho việc sắp thứ tự tập tin.

5. Sắp xếp đơn giản

HƯỚNG DẪN GIẢI THUẬT

- Bắt đầu từ phần tử đầu tiên của đáy, so sánh với các phần tử còn lại, nếu không thỏa điều kiện sắp xếp thì hoán vị 2 phần tử đó với nhau, cứ tiếp tục như vậy cho đến phần tử cuối cùng.

- Cụ thể là khi ta phải sắp xếp một bảng A[1.. n] các số nguyên. Ta dùng hai vòng lặp lồng vào nhau được điều khiển bởi i và j:

Cho i đi từ 1 đến n - 1

Cho j đi từ i + 1 đến n

Nếu A [i] > A [j] thì

Hoán vị A [i] và A [j]

CHƯƠNG TRÌNH MẪU

```
Procedure Simplesort (Var A : Array [1.. n] of integer);
Var i, j, TG : integer;
Begin
For i := 1 to n — 1 do
For j := i + 1 to n do
If A [l] > A [j] then
```

Begin

TG := A [i];

A [i] := A [j];

A [j] := TG;

end;

End;

6. Sắp xếp nhanh

HƯỚNG DẪN GIẢI THUẬT

- Chia dãy cần sắp xếp thành hai phần, lấy giá trị ở giữa X làm chuẩn để so sánh.

- Đi tìm một phấn tử A ở dãy trên có giá trị lớn hơn X.

- Đi tìm một phần tử B ở dãy dưới có giá trị nhỏ hơn X.

- Hoán vị 2 phần tử A, B.

- Tiếp tục như vậy cho đến khi chúng ta đạt được dãy trên chứa các giá trị nhỏ hơn X, dãy dưới chứa các giá trị lớn hơn X.

- Tiếp tục áp dụng thuật toán trên vào hai dãy con trên và dưới (Đê quy) cho đến khi không chia được nữa thì việc sắp xếp hoàn

- Cụ thể là xét một đoạn của dãy từ thành phần thứ L thành phần thứ R ta thực hiện các thao tác:

* Lấy giá trị của thành phần thứ (L + R) div 2 gán vào biến

* Cho i ban đầu là L.

* Cho j ban đầu là R

* Lặp lại.

- Chừng nào còn A [i] < X thì tăng i.
- Chừng nào còn A [j] > X thì giảm j.
- Nếu i = < j thì

— Hoán vị A [i] và A (j]

— Tăng i

— Giảm

Cho đến khi i > j.

* Sắp xếp đoạn từ A [L] đến A Ịj].

* Sắp xếp đoạn từ A [i] đến A [R].

CHƯƠNG TRÌNH MẪU

```
Procedure Quicksort (Var A : Array [1.. n] of integer);
Procedure Sort (L, R : byte);
Var  i, j : Byte;
```

```
TG, X : integer;
Begin
X := A [(L + R) div 2];
i := L;
j := R;
Repeat
While (A [i] < X) Do Inc (i);
While (A [j] > X) Do Dec (j);
IF i <= j then Begin
TG := A [i];
A [i] := A 01;
A U] := TG;
Inc (i);
Dec (j); end;
Until i > j;
IF L < j then Sort (L, j);
IF i < R then Sort (i, R);
End;
Begin
Sort(l, n);
End;
```

7. Các bài toán sắp xếp cơ bản

Bài 1 : Hãy viết chương trình in ra màn hình tất cả các hoán vi của n số nguyên dương đầu tiên. Số nguyên dương n nhập từ bàn phím (n =<10).

VÍ DỤ:

n = 2

Hoán vị thứ 1:12

Hoán vị thứ 2:21

n = 3

Hoán vị thứ 1:1 2 3

Hoán vị tlụí 2:1 3 2

Hoán vị thứ 3 : 2 1 3

Hoán vị thứ 4 : 2 3 1

Hoán vị thứ 5: 3 1 2

Hoán vị thứ 6: 3 2 1

HƯỚNG DẪN GIẢI THUẬT

- Với mỗi vị trí của hoán vị ta lần lượt chọn các giá trị từ 1 đến n (chưa được chọn) vào vị trí này và đãnh dấu số đã chọn.

- Tiếp tục tìm các số ở vị trí tiếp theo cho đến khi chọn đủ n số vào n vị trí của hoán vị và sau đó xuất kết quả ra màn hình.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_1;
UsesCRT;
Const MaxN = 10 ;
Var  n : Byte;
A, B : Array [1.. MaxN] of Byte;
dem : Longint;
Procedure Nhap:
Begin
Clrscr;
Write (‘Nhap n = ‘);
Readln (n);
FillChar (A, SizeOf (A), 0);
B := A;
dem := 0;
End;
Procedure Xuat;
Var  i : Byte;
Begin
Inc (dem);
Write (‘Hoan vi thu’, dem, ‘ : ‘);
For i := 1 to n do Write (A [i], ’ ‘);
Writeln;
End;
Procedure Tim (VT : Byte);
Var  i : Byte;
Begin
If VT = n + 1 then Xuat
Else
Begin
For i := 1 to n do
If B [i] = 0 then
```

```
Begin
B [i] := VT;
A [VT] := i;
Tim (VT + 1);
A [VT] := 0;
B [i] := 0;
End;
End;
End;
BEGIN
Nhap;
Tim (1);
Readln;
END.
```

Bài 2. Hãy viết chương trình xuất ra màn hình tất cả các chỉnh hợp chập k của n số nguyên dương đầu tiên. Số n, k nhập phím.

VÍ DỤ:

n = 2, k = 1

Chỉnh hợp thứ 1 : 1

Chỉnh hợp thứ 2 : 2

n = 3, k = 2

Chỉnh hợp thứ 1 : 1 2

Chỉnh hợp thứ 2: 1 3

Chỉnh hợp thứ 3: 2 1

Chỉnh hợp thứ 4: 2 3

Chỉnh hợp thứ 5: 3 1

Chỉnh hợp thứ 6: 3 2

HƯỚNG DẪN GIẢI THUẬT

— Với mỗi vị trí của chỉnh hợp ta lần lượt chọn các giá trị từ 1 đến n (chưa được chọn) vào vị tri này và đánh dấu số đã chọn

— Tiếp tục tìm các số ở vị trí tiếp theo cho đến khi chọn đủ số vào k vị trí của chỉnh hợp và sau đó xuất kết quả ra màn hình.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_2;
Uses CRTl
Const MaxN = 10;
```

```
Var n, k : Byte;
A, B : Array [1.. MaxN] of Byte;
dem : LongInt;
Procedure Nhap;
Begin
Clrscr;
Write (’Nhap n = ’);
Readln (n);
Write (’Nhap k = ’);
Readln (k);
FillChar (A, SizeOf (A), 0);
B : = A;
dem := 0;
End;
Procedure Xuat;
Var  i : Byte;
Begin
Inc (dem);
Write (’Chinh hop thu’, dem,’.’);
For i := 1 to k do
Write (A [i], ’ ’);
Writeln;
End;
Procedure Tim (VT : Byte);
Var  i : Byte;
Begin
If VT = k + 1 then Xuat
Else
Begin
For i := 1 to n do
If B[i] = 0 then
Begin
B [i] := VT;
A [VT1 := i;
Tim (VT +1);
A [VT]:<= 0;
```

```
B [i] := 0;
End;
End;
End;
BEGIN
Nhap;
Tim (1);
Readln;
END.
```

Bài 3. Hãy viết chương trình xuất ra màn hình tất cả các tổ hợp chập k của n số nguyên dương đầu tiên. Số k nhập từ bàn phim.

VÍ DỤ

n = 2, k = 1

Tổ hợp thứ 1 : 1

Tổ hợp thứ 2 : 2

n = 3, k = 2

Tổ hợp thứ 1 : 1 2

Tổ hợp thứ 2:13

Tổ hợp thứ 3:23

HƯỚNG DẪN GIẢI THUẬT

- Với mỗi vị trí của tổ hợp ta lần lượt chọn các giá trị từ 1 đến n (chưa được chọn và có giá trị lớn hơn giá trị của phần tử trước đó, nếu có) vào vị trí này và đánh dấu số đã chọn.

- Tiếp tục tìm các số ở vị trí tiếp theo cho đến khi chọn đủ k số vào k vị trí của tổ hợp và sau đó xuất kết quả ra màn hình.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_3;
Uses CRT;
Const MaxN = 10;
Var n, k : Byte;
A, B : Array [0.. MaxN] of Byte;
dem : LongInt;
Procedure Nhap;
Begin
Clrscr;
Write (‘Nhap n = ‘);
```

```
Readln (n);
Write (’Nhap k = ’);
Readln (k);
Fillchar (A, SizeOf (A), 0);
B :=A; dem : = 0;
End;
Procedure Xuat;
Var  i : Byte;
Begin
Inc (dem);
Write ('Chinh hop thu', dem, ’ : ');
For i :=1 to k do Write (A [i], ’ ’);
Writeln;
End;
Procedure Tim (VT : Byte);
Var i : Byte;
Begin
If VT = k + 1 then Xuat
Else
Begin
For i := A [VT - 1] + 1 to n do
If B [i] = 0 then
Begin
B [i] := VT;
A [VT] := i;
Tim (VT + 1);
A [VT] := 0;
B [i] := 0;
End;
End;
End;
BEGIN
Nhap;
Tim (1);
Readln;
END.
```

Bài 4 : Bạn cần xếp hạng cho học sinh của một lớp.

Giả sử mỗi môn học đều là hệ số 1.

Bạn hãy chia điểm trung bình cho các học sinh trong lớp và sắp theo thứ tự giảm dần của điểm trung bình. Nếu hai học sinh có cùng điểm trung bình thì sáp họ tên theo thứ tự tự điển ABC.

Dữ liệu: cho trong file INP.BL4

+ Dòng đầu ghi số N là số lượng học sinh và số M là số môn học. (N =< 100 và M =< 10)

+ N dòng tiếp theo, mỗi dòng ghi một chuôi kí tự là tên học sinh.

+ N dòng tiếp theo, mỗi dòng ghi M số dương là điểm của mỗi học sinh.

+ Mỗi số ghi cách nhau ít nhất một dấu cách.

Tên học sinh là các ký tự in hoa.

Kết quả ; xuất ra màn hình danh sách học sinh sau khi xếp hạng.

HƯỚNG DẪN GIẢI THUẬT

— Giải thuật BUBBLE SORT

For i := 1 to n do

For j := n downto i + 1 do

If GT [j] > GT [j - 1] (có thể GT [j] < GT u - 1]) then

Hoán_vị (GT [j], GT [j - 1]);

— Sử dụng giải thuật này, trước tiên ta sắp xếp danh sách học sinh theo thứ tự tự điển ABC. Sau đó sắp xếp lại theo điểm trung bình.

CHƯƠNG TRÌNH MẪU

```
- PROGRAM CHUONG_TRINH_MAU_BAI_3_4;
Uses CRT;
Const MaxN = 100;
MaxM = 10;
Type TH =Record
Ten : String;
c : Char;
Diem : Array [1.. MaxM] of Real;
TB : real;
End;
Var M, N, i, j : Byte;
HS : Array [1.. MaxN] of TH;
Procedure Nhap;
Var  f : Text;
```

```
Begin
Assign (f, 'INP.BL4');
Reset (f);
Readln (f, N, M);
FillChar (HS, SizeOf (HS), 0);
For i := 1 to N do
With HS [i] do
Begin
Readln (f, Ten);
C := Ten [1];
For j := 1 to Length (ten) - 1 do
If Ten [j] = #32 then C := Ten [j + 1];
End;
For i := 1 to N do Begin
With HS [i] do
For j := 1 to M do
Read (f, Diem [j]);
Readln (f);
End;
Close (f);
End;
Procedure Sort1;
Var  tam : TH;
Begin
For i := 1 to N - 1 do
For j : = N down to i + 1 do
If HS [j] . C < HS [j - 1] . C then
Begin
tam := HS [j];
HS [j] := HS [j - 1];
HS [j - 1 ] : = tam;
End;
End;
Procedure Sort2;
Var  tam : TH;
Begin
```

```
For i := 1 to N - 1 do
For j := N down to i + 1 do
If HS [j] . TB > HS [j - 1] . TB then
Begin
tam := HS JJ];
HS [j] := HS [j - 1];
HS[j- 1] := tam;
End;
End;
Procedure Tinh;
Begin
For i := 1 to N do
With HS [i] do
Begin
TB := 0;
For j := 1 to M do
TB := TB + Diem [j];
TB := TB/M;
End;
Sort1;
Sort2;
End;
Procedure Xuat;
Begin
Clrcr;
For i := to N do
Writeln (i, '—` HS [i] . Ten); Readln;
End;
BEGIN
Nhap;
Tinh;
Xuat;
END.
```

Bài 5 : Xét n khung chữ nhật độ rộng 1 (n < 26). Khung được chia thành các ô kích thước 1 x 1. Tát cả các ô của mỗi khung được điền chữ cái Latinh in hoa. (Hình 1).

(Hình 1).

Các khung được xếp đè lên nhau sao cho các cạnh vẫn song song với trục tọa độ và tọa độ các đỉnh đều nguyên.

Từ trên nhìn xuống ta chỉ thấy phân của mỗi khung. Hãy sắp xếp các kí tự ứng với mỗi khung sao cho bất kì khung i nào năm dưới khung j thì kí tự của khung i phải nòm sau kí tự của khung j.

Dữ liệu: cho trong file INP.BL5

+ Dòng đầu chứa hai số nguyên H,L với H,L =< 30, cách nhau ít nhất một dấu cách. H dòng tiếp theo mỗi dòng 1 xâu L kí tự quan sát được trong cửa sổ độ cao H và độ rộng 1.. Vị trí trống (khung thuộc khung nào) được đánh dấu bằng dấu'.' (dấu chấm).

Kết quả: đưa màn hình xâu kí tự ứng với trình tự các khung.

HƯỚNG DẪN GIẢI THUẬT

— Xác định vị trí các đỉnh của khung:

+ Tọa độ đỉnh trên bên trái sẽ là điểm mang kí tự của khung và có giá trị tọa độ nhỏ nhất, (trong ví dụ 1 ở trên, khung E có đỉnh trên bên trái là (2, 1)).

+ Tọa độ đỉnh dưới bên phải sẽ là điểm mang kí tự của khung và có giá trị tọa độ lớn nhất (trong ví dụ 1 ở trên, khung E có đỉnh dưới bên phải là (9, 6)).

— Thiết lập ma trận vuông A biểu hiện quan hệ giữa các khung (nếu có)

1. Nếu khung i nằm dưới khung j thì GRA[i, j] = TRUE và ngược lại.

2. Nếu GRA[i, j] = True và GRA[i, k] = True thì GRA [i, k] = True.

3. Lập lại bước 2 đến khi không thực hiên được nữa.

— Ta gọi Khung i nhỏ hơn khung j nếu Gra [i, j] = True và áp dụng giải thuật BUBBLE SORT ở bài 4 để sắp xếp các khung theo thứ tự giảm dần.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_5;
Uses CRT;
Type FramType = Record
x, y : Byte;
End;
Saved Type = Record
Letter : Char;
Numbr : Byte;
End;
Var  h, l : byte;
Fr : Array [1.. 100, 1.. 100] of Char;
```

```
Ch : Array [1.. 26] of SavedType;
Count : Byte;
Lt : Array [1.. 26, 1.. 2] of FramType
Gra : Array [1.. 26, 1.. 2] of Boolean;
Flag : Array [1.. 26] of Byte;
Functi0n Max (m1, m2 : Byte) : Byte;
Begin
If m1 > = m2 then Max := m1
Else Max := m2;
End;
Functi0n Min (m1, m2 : Byte) : Byte;
Begin
If m1 >= m2 then Min := m2
Else Min := m1;
End;
Procedure Nhap;
Var  f : text;
Filename : String;
i, j : Byte;
Begin
Filename := 'INP.BL5';
{$I-}
Assign (f, Filename);
Reset(f);
If I0Result < > 0 then
Begin
Writeln (#7, 'Cannot open file', Filename);
Halt;
End;
{$I+}
FillChar (Fr, SizeOf (Fr), #0);
For i := 1 to 26 do
Begin
Ch [i] . Letter := #0;
Ch [i] . Numbr := 0;
End;
```

```
FillChar (Flag, SizeOf (Flag), 0);
FillChar (Lt, SizeOf (Lt), 0);
Readln(f, h, l);
Count := 0;
For i := 1 to h do
Begin
For j := 1 to 1 do
Begin
Read (f, Fr [i, j]);
If Fr[i, j] =’ ’ then read (f, Fr [i, j]);
If Fr[i, j] <> ‘.’ then
If Flag [Ord (Fr [i, j]) - 64 = 0 then
Begin
Inc (Count);
Flag [Ord (Fr [i, j]) – 64 := 0 Count;
Ch [Count].Letter := Fr [i, j];
Ch [Count].Numbr := Count;
Lt [Count, 1].x := j;
Lt [Count, 1].y := i;
Lt [Count, 2].x := j;
Lt [Count, 2].y := i;
End
Else
Begin
Lt [Flag [Ord (Fr [i, j]) - 64], 1].x := Min
(Lt [Flag [Ord (Fr [i, j]) - 64], 1].x, j);
Lt [Flag [Ord (Fr [i, j]) - 64], 1].y :=
Min (Lt [Flag [Ord (Fr [i, j]) - 64], 1].y, i);
Lt [Flag [Ord (Fr [i, j]) - 64], 2].x : = Max
(Lt [Flag [Ord (Fr [i, j]) - 64], 2].x, i);
Lt [Flag [Ord (Fr [i, ji) - 64], 2].y := Max
(Lt [Flag [Ord (Fr [i, j]) - 64], 2].y, 0;
End;
End;
Readln (0;
End;
```

```
End;
Procedure Xuly;
Var  i, j, k : Byte;
done : Boolean;
Begin
FillChar (Gra, SizeOf (Gra), False);
For k := 1 to Count do
Begin
j := Lt [k, l].x;
For i := Lt [k, l].y to Lt [k, 2].y do
If Fr [i, j] <> Ch [k].Letter then
Gra [k, Flag [Ord (Fe [i, j]) - 64]] := True;
j := Lt [k, 2].x;
For i := Lt [k, 1].y to Lt [k, 2].y do
If Fr [i, j] <> Ch [k] .Letter then
Gra [k, Flag [Ord (Fe [i, j]) - 64]] := True;
i := Lt [k, l].y;
For j := Lt [k, 1] . X to Lt [k, 2].x do
If Fr [i, j] <> Ch [k].Letter then
Gra [k, Flag [Ord (Fe [i, j]) - 64]] := True; i := Lt [k, 2].y;
For j := Lt [k, l].x to Lt [k, 2] . X do
If Fr [i, j] <> Ch [k].Letter then
Gra [k, Flag [Ord (Fe [i, j]) - 64]] := True;
End;
Repeat
done := True;
For k := 1 to Count do
For i := 1 to Count do
For j := 1 to Count do
If Gra [k, i] and Gra [i, j] and
Gra [k, j] = False then
Begin
Gra [k, j] := true;
done := False;
End;
Until done;
```

```
End;
Procedure Sort;
Var  i, j : Byte;
Tempt : SavedType;
Begin
For i := 1 to Count - 1 do
For j := Count downto i + 1 do
If Gra [Ch [j -1].Numbr, Ch [j].Nurabr] then
Begin
Tempt := Ch [j -1];
Ch [j -1] := Ch [j -1];
Ch [j] := Tempt;
End;
End;
Procedure Xuat
Var  i : Byte;
Begin
For i := 1 to Count do
Write (Ch [i] . Letter, ' ');
Readln;
End;
BEGIN
Nhap;
Xuly;
Sort;
Xuat;
END.
```

Bài 6. Giả sử P = (P1, P2,… Pn) là một hoán vị của 1, 2,… n.

Bảng nghịch thế của hoán vị p là dãy T = (t1,t2m,…,tn} trong đó t1 bằng số các phần tử của hoán vị P đứng bên trái i và lớn hơn i.

Viết chương trình cho phép từ bảng nghịch thế T xây dựng lại hoán vi P tương ứng.

Dữ liệu nhập từ bàn phím và kết quả xuất ra màn hình.

VÍ DỤ:

T = (2, 4, 8, 6, 0, 2, 2, 1, 0)

=> P = (5, 9, 1, 8, 2, 6, 4, 7, 3)

T = (2, 1, 0, 0, 0)

=> P = (3, 2, 1, 4, 5)

HƯỚNG DẪN GIẢI THUẬT

Ta tìm vị trí của mỗi số i (1 =< i =< N) trong hoán vị P:

1. j := 0; k :=0;

2. Tăng k lên 1 đơn vị. Nếu P [k] = 0 thì tăng j lên 1 đơn vị.

3. Trở lại bước 2 cho đến khi nào j > T [i].

4. Gán p [k] := i.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_6;
Const NN = 100;
Var  n, i, j, k : Integer;
P, L : Array [1.. NN] of Integer;
BEGIN
Write (‘N =’);
Readln (N);
FillChar (P, SizeOf (P), 0);
For i := 1 to N do
Begin
Write (‘T [‘,i,’] = ’); Readln (L [i]);
End;
For i := 1 to N do
Begin
j := 0; k := 0;
Repeat
Inc (k);
If P [k] = 0 then j := j + 1;
Until j > L [i]; p [k] := i;
End;
For i := 1 to N do
Writeln (P [i]);
END.
```

Bài 7 : Cho các số tự nhiên n (n > 2), m và mảng 3 chiều A[1.. m, 1.. m, 1.. n -1]. Tìm giá trị bé nhất của biểu thức:

R = A[i1, t2, 1]+A[i2,i3, 2] +...+ A[i_{n-1}, i_n n—1] đối với mọi bộ số có thể i1, i2,…, in

Dữ liệu nhập từ bàn phóm va kết quả xuất ra màn hình.

HƯỚNG DẪN GIẢI THUẬT

— For k := 1 to n-1 do

+ For j := 1 to m do

* r = Min {B [i] 4- A [i,j,k] I i = 1.. m}

* C [j] := r

+ B := C

— r = Min {B [i] I i = 1.. m}

— r chính là kết quả cần tìm

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_7;
Const MM = 10;
NN = 10;
Var  m, n, i, j, k : Integer;
x, r : real;
A : Array [1.. MM, 1.. MM, 1.. NN] of Real;
B, C : Array [1.. MM] of Real;
BEGIN
Write (‘M = ’); Readln (M);
Write (‘N= ’); Readln (N);
For i := 1 to m do
Begin
B [i] := 0;
For j : = 1 to m do
For k := 1 to n -1 do
Begin
Write (‘A [‘, i, ‘, ‘, j, ‘, ‘, k, ‘] = ’);
Readln (A [i, j, k]);
End;
End;
For k := 1 to n -1 do
Begin
For j := 1 to m do
Begin
r := B [i] + A [1, j, k];
For i := 2 to m do
```

```
Begin
X := B [i] + A [i, j, k];
If x < r then r := x;
End;
C [j] := r;
End;
For j := 1 to m do B [j] := C [j];
End;
r := B [i];
For i := 2 to m do If B [i] < r then r := B [i]
Writeln (r);
END.
```

Bài 8. Cho dãy số (A) :A1,A2,...,AN và dãy số (B): B1, B2,...,BM các số nguyên.

Hãy sáp xếp hai dãy số trên thành dãy sô (C)

C1 =< C2 =< … =< CM+N

Dữ liệu: cho trong file INP.BL8 gồm 3 dòng

+ Dòng 1: chứa hai số N và M

+ Dòng 2:ghi N số A1, A2,…,AN

+ Dòng 3: ghi M số B1, B2,...,BM

Các số ghi trên cùng một dòng cách nhau nhát một dấu cách

Kết quả: xuất ra màn hình

Cho biết N và M rất lớn.

VÍ DỤ :

HƯỚNG DẪN GIẢI THUẬT

+ Đọc toàn bộ giá trị của hai dãy số (A) và (B) vào dây (C)

+ Sắp xếp lại toàn bộ dãy (C).

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_8 ;
Uses CRT ;
Const Max = 30000 ;
Var  c : Array [1.. Max] of Integer
m, n, i, j : Integer ;
Procedure Nhap ;
Var  f : Text ;
Begin
```

```
Assign (f, 'INP.BL8') ;
Reset (f) ;
Readln (f, n, m) ;
FillChar (C, SizeOf (C), 0) ;
For i := 1 to n do
Read (f, cm) ;
Readln (f) ;
For i : = 1 to m do
Read (f, C[n + i]) ;
n := n + m ;
Close (f) ;
End ;
Procedure Sort ;
Begin
For i := 1 to n -1 do
For j := n down to i + 1 do
If C [j] < C[j - 1] then
Begin
m := C[j] ;
C[j] := C[j - 1] ;
C[j - 1] := m ;
End ;
End ;
Procedure Xuat ;
Begin
For i := 1 to n do
Writeln ('C [', i, '] = ', C [i]) ;
Readln ;
End ;
BEGIN
Nhap ;
Sort ;
Xuat ;
END.
```

Bài 9. Cho mảng A [1..N, 1..M] chứa các số nguyên.

Hãy sắp xếp lại các giá trị của các ô trong A sao cho

A[i,1] =< A[i,2]... =< A[i,M]

A[1,j] < A[2,j]... < A[N,j]

Dữ liệu: cho trong file INP.BL9 gồm N + 1 dòng

+ Dòng 1: chứa hai số N và M

+ Dòng i+1(1 =< i =< N) ; ghi M số A [i, 1], A [i, 2],…, A [i,N]

Các số ghi trên cùng một dòng cách nhau ít nhất một dấu cách. Kết quả xuất ra màn hình

VÍ DỤ

HƯỚNG DẪN GIẢI THUẬT

- Đọc toàn bộ n x m giá trị của mảng vào một mảng C[1..nxm].

Sắp xếp toàn bộ mảng C theo thứ tự tăng dần.

-Điền mảng C vào lại ma trận theo thứ tự từ tái qua phải và từ trên xuống dưới.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_9 ;
Uses CRT ;
Const Max = 30000 ;
Var C : Array[1.. Max] of Integer ;
m,n,I,j,k : integer ;
Procedure Nhap ;
Var f : Text ;
Begin
Assign ( f, 'INP.BL9') ;
Reset (f) ;
Readln (f,n,m) ;
FillChar (C, SizeOf (C), 0) ;
For I := 1 to n do
Begin
For j := 1 to m do
Read ( f, C [(i – 1) * n + j]) ;
Readln (f) ;
End ;
Close (f) ;
n := n * m ;
End ;
Procedure Sort ;
```

```
Begin
For I := 1 to n do
For j := n downto i + 1 do
If C [j] < C [j = 1] then
Begin
K := C [j] ;
C [j] := C [j = 1] ;
C [j – 1] := k ;
End ;
End ;
Procedure Xuat ;
Begin
N := n div m ;
For I : = 1 to n do
Begin
For j := 1 to m do
Write (C [(i = 1) * m + j], ' ') ;
Writeln ;
End ;
Readln;
End;
BEGIN
Nhap ;
Sort ;
Xuat ;
END.
```

Bài 10. Nhập vào từ bàn phím một chuỗi kí tự S.

Hãy xuất ra màn hình tất cả các hoán vị khác nhau của các kí tự trong chuỗi S đã cho (Biết các kí tự trong S không nhất thiết khác nhau).

Ví Dụ:

S= AABB

Hoán vị thứ 1 : AABB

Hoán vị thứ 2 : ABAB

Hoán vị thứ 3 : ABBA

Hoán vị thứ 4 : BAAB

Hoán bị thứ 5 : BABA

Hoán vị thứ 6 : BBAA

S = AABBA

Hoán vị thứ 1: AABBA

Hoán vị thứ 2 : AABAB

Hoán vị thứ 3 : AAABB

Hoán vị thứ 4: ABABA

Hoán vị thứ 5: ABAAB

Hoán vị thứ 6 : ABBAA

Hoán bị thứ 7: BAABA

Hoán vị thứ 8 : BAAAB

Hoán vị thứ 9 : BABAA

Hoán vị thứ 10: BBAAA

HƯỚNG DẪN GIẢI THUẬT

- Tương tự giải thuật tìm hoán vị một tập hợp những chỉ khác khi chọn một phần tử nào trong S vào một vị trí nào đó của hoán vị, ta phải kiểm tra xem phần tử đó đã sử dụng hay chưa và đồng thời có phần tử nào trong S giống với phần tử đang xét và có vị trí xuất hiện trong S trước phần tử này mà chưa được sử dụng hay không.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_3_10 ;
Uses CRT;
Var s: String ;
KT : Array [1.. 255] of Boolean ;
B : Array [1.. 255] of Char ;
n : Byte ;
Cnt : LongInt ;
Functi0n OK (i : Byte) : Boolean;
Var j : Byte ;
Begin
OK := False ;
For j := 1 to i – 1 do
If ( S [i] = S [j] and (not KT [j] then Exit ;
Ok := True ;
End ;
Procedure Xuat ;
Var i : Byte ;
```

```
Begin
Inc (Cnt) ;
Write (‘Hoan vi thu’, Cnt, ‘:’);
For i := 1 to n do
Write (B [i]) ;
Writeln ;
End ;
Procedure Hvi (Po : Byte) ;
Var I : Byte ;
Begin
If Po = n + 1 then Xuat
Else
For i := 1 to n do
If (not (KT [i])) and (OK (i)) then
Begin
KT [i] := True ;
B [Po] := S [i] ;
HVi (Po + 1) ;
KT [i] := False ;
B [Po] := #0;
End ;
End ;
BEGIN
Clrscr ;
Write (‘S = ’) ;
Readln (S) ;
n := Length (S) ;
Cnt := 0 ;
FillChar (KT, SizeOf (KT), 0) ;
FillChar (B, SizeOf (B), 0) ;
HVi (1) ;
Readln ;
END.
```

§4. QUY HOẠCH ĐỘNG

1. Khái niệm:

Một trong các hướng giải quyết các bài toán tối ưu trong tin học là sử dụng phương pháp QUY HOẠCH ĐỘNG. Tư tưởng chủ đạo của phương pháp này dựa trên nguyên lý tối ưu của Bellman phát biểu như sau:

— Nếu một dãy các lựa chọn là tói ưu thì mọi dãy con của nó cũng tối ưu

Ngoài ra khi thiết kế các thuật toán quy hoạch động ta thường dùng kỹ thuật "phân vùng để xử lí", nghĩa là để giải quyết một bài toán lớn ta chia nó thành nhiều bài toán con có thể giải quyết độc lập. Trong phương pháp quy hoạch động, việc thể hiện nguyên lý này được đẩy đến cực độ:

— Khi không biết chắc chắn cần giải quyết bài toán con nào, chúng ta giải quyết tất cả các bài toán con và lưu trữ những lời giải này với mục đích sử dụng chúng theo một sự phối hợp nào đó để giải quyết các bài toán tổng quát hơn.

2. Các thao tác tổng quát của quy hoạch động:

— Xây dựng hàm quy hoạch động

— Lập bảng lưu lại giá trị của hàm

— Tính các giá trị ban đầu của bảng

— Tính các giá trị còn lại theo kích thước tăng dần của bí cho đến khi đạt được giá trị tối ưu cần tìm.

— Dùng bảng lưu để truy xuất lời giải tối ưu

3. Hạn chế của quy hoạch động:

Phương pháp quy hoạch động không đem lại hiệu quả trong tình huống sau:

— Sự kết hợp lời giải của các bài toán con chưa chắc đã cho lời giải của bài toán lớn hơn.

— Số lượng các bài toán con cần giải quyết và lưu trữ kết quả có thể rất lớn, không thể chấp nhận được.

Cho đến nay, vẩn chưa có ai xác định được một cách chính xác những bài toán nào có thể giải quyết được một cách hiệu quả bằng phương pháp quy hoạch động. Có những vấn đề quá phức tạp và khó khăn mà xem ra không thể ứng dụng quy hoạch động để giải quyết được, trong khi đó cũng có những bài toán đơn giản khiến cho 1 sử dụng quy hoạch động để giải quyết lại kém hiệu quả hơn so dùng các thuật toán kinh điển.

4. Các bài toán quy hoạch động cơ bản:

Bài 1 : Cho n đồ vật. Thể tích và giá trị của vật i lần lượt là A[i], B [i]. Hãy chọn ra 1 số vật sao cho tổng thể tích của chúng không vượt quá thể tích V cho trước và tổng giá trị của chúng là nhất.

Cho biết $0 < n < 100$

$0 < V < 100$

0 < A [i] , B [i] < 256

Dữ liệu : Cho trong file INP.BL1 gồm n + 1 dòng

+ Dòng đầu là 2 số n, V

+ Dòng i+1(1 =< i =< n) ghi 2 số nguyên dương A [i], B [i,j.

Kết quả : xuất ra màn hình dưới dạng

+ Mỗi dòng ghi 3 số : i, A [i], B [i] của vật i được chọn

+ Dòng cuối là 3 số : tổng số vật, tổng thể tích và tổng giá trị của các vật được chọn.

Các số ghi trên cùng một dòng ghi cách nhau ít nhất một dấu cách, ví dụ ;

HƯỚNg DẪN GIẢI THUẬT

_ Gọi Fx (i, j) là tổng gió trị lớn nhất trong trường hợp có i vật từ 1 đến i và thể tích tối đa là j.

— For i := 1 to n do

For j := 1 to V do

Nếu j >= A [i] thì

Fx [i, j] := Max (Fx [i - 1, j - A [i]] + B [i], Fx [i-1, j])

Nếu không thì fx [i, j] := Fx [i - 1, j]

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_1 ;
UsesCRT ;
Const MaxN = 100 ;
MaxV = 100 ;
Var  Fx : Array [0 .. MaxN, 0 .. MaxV] of Word
Lay : Array [1 .. MaxN] of Boolean ;
A, B : Array [1 .. MaxN] of Byte ;
n, V : Byte ;
i, j : Word ;
Procedure Nhap ;
Var f : Text ;
Begin
Clrscr ;
Assign (f, 'INP.BL1') ;
Reset (f) ;
Readln (f, n, V) ;
FillChar (A SizeOf (A), 0) ;
B := A ;
```

```
For i := 1 to n do
Readln (f, A [i], B [i]) ;
Close [f] ;
End ;
Functi0n GetMnx (vl, v2 : Word) : Word ;
Begin
If v1 > v2 then GetMax := v1
Else Got Max := v2 ;
End ;
Procedure Xuly ;
Begin
FillChar (Fx, SizeOf (Fx), 0) ;
For i := 1 to n do
For j := 1 to V do
Begin
If j > = A [i] then
Fx [i, j] := GetMax (Fx [i-1, j-A[i]] + B[i],Fx[i-1, j])
Else
Fx [i, j] := Fx [i -1, j] ;
End ;
End ;
Procedure Xuat ;
Var  Max, vo, n0, VI : Word ;
Begin
Max := 0 ;
V0 := 0 ;
n0 := 0 ;
For i := 1 to V do
If Max <= Fx [n, i] then
Begin
Max := Fx [n, i] ;
V0 := i ;
End ;
V1 := V0 ;
For i := n down to 1 do
If Fx [i, VO] <> Fx [i - 1, V0] then
```

```
Begin
Inc (n0) ;
Writeln (’Vật thứ’, i, ‘- Thể tích’, A[i],’- Giá trị’, B [i]) ;
V0 := V0 - A [i] ;
End ;
Writeln (’Chọn’, n0,’vật — Tổng thể tích’ , V1,’- Tổng giá trị’ , Max) ;
End ;
BEGIN
Nhap ;
Xuly ;
Xuat ;
End.
```

Bài 2 : Có N một hàng và M nước (N < 40,M < 15)

Cho biết đơn giả mặt hàng i ở nước thứ j là C [i, J] (đơn vị tiền tệ ở nước j). Hàng hóa là không thể phân chia được. C [i, j] nguyên dương. Với số tiền ban đầu là S (đơn vị tiền tệ của nước 1, S =< 200) người ta chỉ cố thể mua một loại hàng ở nước 1 và mang sang bán ở nước thứ hai nào đó, dùng tiền thu được mua một loại hàng ở nước và mang sang bán ở nước thứ ba, ... Tiền còn thừa không được phép chuyển đổi. Không được phép mua hàng tại một nước và cũng bán hàng này tại nước đó. Hãy lập kế hoạch mua bán hàng hóa sao cho đúng k lần (k =< 10; mua bán hàng ở nước ngoài (một nước có thể được nhiều lẫn) người ta quay về nước 1, bán tất cả các hàng và thu được nhiều tiền nhất, (lần mua hàng đầu tiên tại nước 1 xem như lần mua bán thứ 0, lần bán hàng cuối cùng tại nước 1 xem như lần k + 1)

Dữ liệu : Cho trong file INP.BL2 gồm M + 1 dòng

+ Dòng 1 : gồm các số M N S K

+ Dòng i + 1 : Gồm N số C [i, 1] C [i, 2] .. C[i, N]

Các số trên một dòng cách nhau ít nhất một dấu cách.

Kết quả : xuất ra màn hình dưới dạng :

+ Dòng 1 : Tổng số tiền thu được theo phương án tối ưu.

+ k + 2 nhóm dòng tiếp theo (mỗi nhóm gồm 3 dòng) :

- Dòng thứ nhất mỗi nhóm ghi số thứ tự của lần mua bán hàng và số thứ tự của nước.- Dòng thứ hai mỗi nhóm ghi số thứ tự của mặt hàng bán ra, số lượng, đơn giá của mặt hàng này tại nước đó và tổng số tiền thu được khi bán hàng (tính theo đơn vị tiền tệ của nước đó).

- Dòng thứ ba mỗi nhóm ghi số thứ tự của mặt hàng mua vào, số lượng, đơn giá của mặt hàng này tại nước đó và tổng số tiền bỏ ra khi mua hàng (tính theo đơn vị tiền tệ của nước đó).

HƯỚNG DẪN GIẢI THUẬT:

- Gọi S0 là số tiền ban đầu, K0 là số lần mua bán
- Gọi S (i, j) là số hàng i tối đa mua tại nước j.
- Gán S = 0
- S [1, j] := S0 div C [1, j]
- For k := 1 to kO do
+ Gán mảng S2 bằng 0
+ For i := 1 to m do
For j := 1 to n do
For i0 := 1 to m do
Nếu i0 khác i thì
For j0 := 1 to n do
Nếu j0 khác j thì
nếu S [i0j0] * C [i,j0] div C [i,j] > S2 [i,j]
thì S2 [i,j] := s [i0j0] * C [i,j0] div C [i,j]
+ Gán s bàng S2.

CHƯƠNG TRÌNH MẪU :

```
($R — S-I-C — V— B — X — T-P-)
PROGRAM CHUONG_TRINH_MAU_BAI_4_2 ;
Uses CRT ;
Const MaxN = 40 ;
MaxM = 15 ;
MaxS = 200 ;
MaxK = 10 ;
Var SS, m, n, S0, k0 : Integer ;
S, S2, C : Array [1.. MaxM, 1.. MaxN] of Integer ;
Next : Array [1.. MaxK, 1.. MaxM, 1.. MaxN] of Record _i, _j : Byte End ;
S1  : Integer ;
Money  : Long Int ;
Procedure Error (S : String) ;
Begin
Writeln (S) ;
Halt ;
```

```
End ;
Procedure Nhap ;
Var  f : Text ;
i, j : Integer ;
Begin
Clrscr ;
Assign (f, 'INP.BL2') ;
Reset (f) ;
Readln (f, m, n, S0, k0) ;
If k0 > MaxK then Error ('K qua lon') ;
If m > MaxM then Error ('M qua lon') ;
If n > MaxN then Error ('N qua lon') ;
For i := 1 to m do
Begin
For j := 1 to n do
Read (f, C[i, j]) ;
Readln (f) ;
End ;
Close (f) ;
End ;
Procedure PMax (Var a : Integerl; b : Integer) ;
Begin
If b > a then a := b
End ;
Procedure Print (k, i, j : Integer) ;
Begin
If k > 0 then
With Next [k, i, j] do
Print (k - 1, _i, _j) ;
If k = 0 then
Begin
S1 := ss div C [i, j] ;
Writeln ('Lan 0 : Nuoc 1') ;
Writeln ('Ban hang : khong co') ;
Writeln ('Mua hang ', j, ' : ', S1, ' x ', C[i, j],' = ', S1 * C [i, j] ;
End
```

```
Else With Next [k, i, j] do
Begin
Money := S1 * C [i,_j] ;
Writeln (’Lan’, k, ’ : Nuoc’, i) ;
Writeln (’Ban hang ’,_j,’ : ‘, S1, ’ x ‘, C [i,_j]. ’ = ‘, Money) ;
S1 :=  Money div C [i, j] ;
Writeln (’Mua hang ’, j, ’ : ’, S1, ’ x ’, C [i, j]. ’ = ‘, S1 * C [i, j]) ;
End ;
End ;
Procedure main ;
Var  Ci,j, max, i, j, i0, j0, tmp, k : Integer ;
p  : ^ Integer ;
Begin
FillChar (S, SizeOf (S), 0) ;
For j := 1 to n do S [1, j] := S0 div C [1, j] ;
For k := 1 to k0 do
Begin
FillChar (S2, SizeOf (S2), 0);
For i := 1 to m do
For j := 1 to n do
Begin
p := CS2[i,j]; Ci,j := C[i,j];
For i0 := 1 to m do
If i < > i0 then
For j0 := 1 to n do
If j < > j0 then
Begin
tmp := S[i0j0]*C[i,j0] div Ci,j;
If tmp > p^ then
Begin
p^ := tmp;
With Next[k,i,j] do
Begin
_i := i0; _j := j0
End;
End;
```

```
End;
End;
S := S2;
End;
max := Low (max);
For i := 2 to m do
For j := 1 to n do
Begin
tmp := S[i,j]*C[1,j];
If tmp > max then
Begin
max := tmp;
i0 := i;
j0 := j
End;
End;
SS := S0;
Writeln ('Tong so tien thu duoc toi da la', max);
Print (k, i0, j0);
Money := S1*C[i,j0];
Writeln ('Lan', k + 1' : Nuoc 1');
Writeln ('Ban hang', j0, S1, 'x', C[1,j0], '=', Money);
Writeln ('Mua hang : khong co');
End;
BEGIN
Nhap;
Main;
END.
```

Bài 3 : Xét bảng số nguyên dương A kích thước N X N (N =< 40) A[i,j] =< 100. Thay mỗi phần tử A[i,j của bảng bằng giá trị f(A[i,j]), trong đó y = f(x) là hàm cho số lượng lớn nhất các số nguyên tố có tổng bằng x, mỗi số sử dụng một lần, trừ một số nguyên tố nào đó có thể sử dụng đúng 2 lần nếu cần.

Ví dụ : x = 5 -> f(x) = 2

(5 = 3 + 2)

x = 7 -> f(x) = 3

(7 = 3 + 2 + 2)

1 không phải là số nguyên tố. Quy ước f(1) = 1

Dữ liệu : cho trong file INP.BL3 gồm N + 1 dòng:

+ Dòng đầu là số nguyên dương N

+ Dòng i + 1 (1 =< i =<N) gồm N số A[i,1] A[i, 2] ... A[i,N]

Kết quả : xuất ra màn hình gồm N dòng

+ Dòng i (1 <i<N) gồm N số A[i,2] ... A[i, N]

Các số ghi trên cùng một dòng ghi cách nhau tí nhất một dấu cách

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT

- Gọi Spt (x) = f(x)
- Gọi NT là mảng các số nguyên tố
- Spt (1) = 1. Spt (2) = 1. Spt (3) = 1
- Spt (i) = Max { Spt (i-NT[k]) + 1 | i > NT[k]}
- Gán mảng TAM bằng Spt
- Spt (i) = Max {Spt (0, TAM (i-NT[k] | i > NT[k]}

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_3;
UsesCRT;
Const MaxN = 40;
MaxVal = 100;
NT : Array[1.. 25] of Byte =
(2, 3, 5, 7, 11, 13, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79,
83, 89, 97);
Dt : Array[l.. 8, 1.. 2] of Shortint =
((-1,-1), (-1,0), (-1,1), (0,-1), (0, 1), (1,-1), (1,0), (1,1));
Var n, i, j, k : Byte;
A : Array [1.. MaxN, 1.. MaxN] of Byte;
max : Integer;
Spt, Tam : Array[1.. MaxVal] of Integer;
Procedure Nhap;
Var  f : Text;
Begin
Clrscr;
Assign (f,'INP.BL3');
Reset (f);
Readln (f,n);
```

```
FillChar (A SizeOf (A), 0);
For i := 1 to n do
Begin
For j := 1 to n do
Read (f,A[i,j]);
Readln (f);
End;
Close (f);
End;
Procedure F;
Begin
FillChar (Spt,SizeOf (Spt), 0);
Tam := Spt;
Spt[1] := 1; ; Spt[2] := 1 ; Spt[3] := 1;
Tam[2] := 1; Tam[3] := 2;
For i := 4 to max do
Begin
For k := 1 to 25 do
Begin
If i < NT[k] then
k := 25
Else
Begin
If i > NT[k] + 1 then
If (Spt[i -NT[k]] + 1 > Spt[i]) and
(Tam[i-NT[k]] < k) then
Begin
Tam[i] := k;
Spt[i] := Spt[i-Nt[k]] + 1; End ;
If (i = NT[k]) and (Spt[i] = 0) then
Begin
Tam[i] := k;
Spt[i] := 1;
End;
End;
End;
```

```
End;
Tam := Spt;
For i := 4 to Max do
Begin
For k := 1 to 25 do
If i <= NT[k] then k := 25
Else
Begin
If Tam[i-NT[k]] + 1 > Spt[i] then
Begin
Spt[i] := Tam[i-NT[k]] + 1;
End;
End;
End;
End;
Procedure Thuchien;
Begin
max := 0;
For i := 1 to n do
For j := 1 to n do
If max < A[i,j] then max := A[i,j];
F;
For i := 1 to n do
For j := 1 to n do
A[i,j] := Spt [A[i,j]];
For i := 1 to n do
Begin
For j := 1 to n do
Write (A[i,j] : 5);
Writeln;
End;
End;
BEGIN
Nhap;
Thuchien;
END.
```

Bài 4. Trong một cửa hàng, mỗi loại hàng có một giá. Ví dụ một bông hoa là 2 đồng và giá một cái bình là 5 đồng. Để thu hút nhiều khác hàng, cửa hàng đề ra một số cách bán đặc biệt.

Một cách bán đặc biệt liên quan đến việc bán một hay một số mặt hàng với giá chung được giảm.

Ví dụ : 3 bông hoa bán với giá 5 đồng thay vì 6 đồng.

2 cái bình và 1 bông hoa bán với giá 10 đồng thay vì 12 đồng

Viết chương trình tính giá mà một khách hàng phải trả cho nhu cầu mua hàng để tận dụng một cách tối ưu các cách bán định nghĩa là phải trả ít nhất. Ví dụ : Với các giá và các cách bán được nêu trên, giá thấp nhát để mua 3 bông hoa và 2 cái bình là 14 + 2 bình và 1 hoa là 10 đồng ; 2 hoa với giá thường là 4 đồng.

Dữ liệu : Cho trong 2 file INP.BL4 và OFFER.B14

File thứ nhất mô tả nhu cầu mua File thứ hai mô tả các cách bán đặc biệt. Trong cả hai file, chỉ có các số nguyên dương.

File INP.BL4 gốm b + 1 dòng:

+ Dòng dầu chứa số b là số loại hàng cần mua (0 =< b =< 5)

+ Mỗi dòng trong b dòng tiếp theo ghi 3 số c,k,p. Giá C là mã của loại hàng (1 =< C =<999). Giá tri k là số đơn vị hàng cần mua với mã C (1 =< k =< 5). Giá trị p là giá bình thường của một đơn vị hàng với mã C (1 =< p =< 999).

Chú ý : Trong một yêu cầu có không quá 5*5 = 25 đơn vị hàng.

File INP.BL4 gồm s + 1 dòng:

+ Dòng đầu chứa số s là số cách bán đờc biệt (0 =< s =< 99)

+ Mỗi dòng trong s dòng tiếp theo mô tả một cách bán đặc biệt. Số đầu tiênn n của mỗi dòng như vậy là số loại hàng trong cách bán đặc biệt tương ứng với dòng đó (1 =< n =< 5); n cặp số tiếp theo (c, k) trong đó C là mã loại hàng, k là số đơn vi hàng đó (1 =< k =< 5; 1 =< C =< 999). Số p cuối cùng trong dòng là giả đã được giảm trong lô hàng này. Giá đã được giảm nhỏ hơn tổng các giá bình thường.

Các số ghi trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

Kết quả : xuất ra màn hình : Thông báo chi phí thấp nhất phải trả cho nhu cầu mua trong file vào

HƯỚNG DẪN GIẢI THUẬT

- Gọi A[Cs1,Cs2,Cs3,Cs4,Cs5] là chi phí nhỏ nhất phải trả khi mua Csi đơn vị hàng i.

- Với mỗi S = A[Cs1,Cs2,Cs3,Cs4,Cs5] ta thực hiện:

For i := 1 to 5 do

+ For j := 1 to 5 do Gán D[j] := 0

+ For j := 1 to Cs[i] do

D[i] := j

Nếu A[cs[1] —d[1], cs[2] -d[2], cs[3] -d[3], cs[4] -d[4], cs[5] —d[5] + Gt[i] * j < S thì gán cho S giá trị này.

For i := 1 to M do

+ Nếu có thể sử dụng cách mua đặc biệt i và số hàng mỗi loại trong cách mua i đều không vượt quá các Csi thì tính lại giá chi phí theo cách mua đặc biệt này và so sánh với S

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_4;
UsesCRT;
Type Mang=Array [1.. 5] of  Record
Vat, sl : Integer;
End;
Var  ten,sl,gt : Array[1.. 5] of Integer;
num : Array[1.. 100] of Integer;
c : Array[l.. 100] of Mang;
cost : Array[1.. 100] of Integer,
b : Array[l.. 100] of Boolean;
a : Array[0..5,0..5,0..5,0..5,0..5] of Longint;
cs,d : Array[1..5] of Byte;
m,n : Byte;
Procedure Nhap;
Var  f : Text;
i,j : Integer,
Begin
Assign(f,'INP.BL4');
Reset(f);
Readln(f,n);
For i := 1 to n do
Readln (f,Ten[i],Sl[i],Gt[i]);
Close (f);
Assign (f,'OFFER.BL4');
Reset (f);
Readln (f,m):
```

```
For i := 1 to m do
Begin
Read (f,Num[i]);
For j := 1 to Num[i] do
Read (f,C[i,j]. Vat,C[i,j].Sl);
Readln (f,Cost[i]);
End;
Close (f);
End;
Function   Tim(x : Integer) : Byte;
Var  i : Integer;
Begin
For i := 1 to n do
If n[i] = X then
Begin
Tim := i;
Exit;
End;
Tim := 0;
End;
Procedure Khoitao;
Var  i,j,k : Integer;
ab : Mang;
Begin
If n < 5 then
Begin
For i := n+1 to 5 do
Begin
Ten[i] := 0; sl[i] := 0; gt[i] := 0;
End; n := 5;
End;
For i := 1 to m do
Begin
B[i] := True;
ab := c[i];
For j := 1 to 5 do
```

```
Begin
C[i,j]. Vat := 0; C[i,j].sl := 0;
End;
For j := 1 to Num[i] do
If B[i] then
Begin
k := Tim(ab[j].Vat);
If k = 0 then B[i] := False
Else
Begin
C[i,k].Vat := ab[j].Vat;
C[i,k].sl := ab[j].sl;
End;
End;
End;
End;
Procedure Tinh1;
Var  s,i,minj : Longint;
Begin
s := 0;
For i := 1 to 5 do s := s+Gt[i]*Cs[i];
Min := s;
For i := 1 to 5 do
Begin
For j := 1 to 5 do d[j] := 0;
For j := 1 to cs[i] do
Begin
d[i] := j;
s := A[cs[l]-d[l],cs[2]-d[2],cs[3]-d[3],cs[4]-d[4],cs[5]-d[5]]+Gt[i]*j;
If s < min then min := s;
End;
End;
A[cs[1],cs[2],cs[3],cs[4],cs[5]] := min;
End;
Procedure Tinh2;
Var  min,s,i : Longint;
```

```
Begin
Min := a[cs[1],cs[2],cs[3],cs[4],cs[5]];
For i := 1 to m do
If b[i] and (cs[1] >= c[i,l].sl) and (cs[2] >= c[i,2].sl)
and (cs[3] >= c[i,3].sl and (cs[4] >= c[i,4].sl)
and (cs[5] >= c[i,5].sl then
Begin
S := a[cs[1]-c[i,1].sl,cs[2]-c[i,2].sl, cs[3]-c[i,3].sl, cs[4]-c[i,4].sl, cs[5]-
c[i,5].sl]+cost[i];
If s < min then min := s;
End;
A[CS[1],CS[2J,CS[3],CS[4], CS[5]] := min;
End;
Procedure Xuly;
Begin
Khoitao;
A[0,0,0,0,0] := 0;
For cs[1] := 0 to al[1] do
For cs[2] := 0 to sl[2] do
For cs[3] := 0 to sl[3] do
For cs[4] := 0 to sl[4] do
For cs[5] := 0 to sl[5] do
Begin Tinh1;
Tinh2;
End;
Clrcsr;
Writeln('Tong chi phi it nhat la: ', A[sl[1], sl[2],sl[3],sl[4],sl[5]]);
Readln;
End;
BEGIN
Nhap;
Xuly;
END.
```

Bài 5 : Có N thành phố. Biết rằng đường đi giữa 2 thành phố bất kì (nếu có) đều là đường đi hai chiều. Sơ đồ mạng lưới giao thông của N thành phố này cho bởi

ma trận kề A[i,j] trong đó + A [i,j] là độ dài đường đi từ thành phố i đến thành phố j

+ A[i,jJ = 0 nếu khòng cỏ đường đi từ thành phó i đến thành phố j

+ Ali,j] = A[j,i]

+ A[i,i] = 0

+ A[i,j] nguyên, không âm.

Hãy xác định đường đi ngắn nhất giữa hai thành phố P và Q hay thông báo không tồn tại lời giải.

Dữ liệu : cho trong file INP.BL5 gồm N+2 dòng

+ Dòng đầu chứa số N (N nguyên dương, N < 50)

+ Dòng i+1 (1 =<i=< N) ghi N số A[i,j] A[i,2],... A[i,N]

+ Dòng N+2 ghi hai số P và Q.

Các số ghi trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

Kết quả : xuất ra màn hình.

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT :

- Gọi A[i,j] là độ dài đường đi ngắn nhất từ i đến j

- Gọi C[i,j] là điểm cần phải qua trên đường đi ngắn nhất từ i đến j

- For k := 1 to N do

For i := 1 to N do

For j := 1 to N do

Nếu có đường đi từ i đến k và từ k đến j thì

Nếu không có đường đi từ i đến j hay đường đi này dài hơn đường đi từ i đến k + đường đi từ k đến j thì A[i,j] := A[i,k]+A[kj] và C[i,j] := k

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_5;
UsesCRT;
Var  A,C : Array[l.. 50, 1..50] of Integer;
     N,P,Q : Integer;
     i,j,k : Integer
Procedure Nhap;
Var  f : Text;
Begin
Assign(f,'INP.BL5');
Reset(f);
Readln(f,N);
```

```
FillChar (A,SizeOf(A),0);
C := A;
For i := 1 to n do
Begin
For j := 1 to n do Read(f,A[i,j]);
Readln(f);
End;
Readln(f,P,Q);
Close(f);
End;
Procedure Xuat (i,j : Integer);
Begin
If C[i,j] = 0 then
Write('->', j)
Else
Begin
Xuat (i, C[i,j]);
Xuat(C[i,j],j);
End;
End;
Procedure Xuly;
Begin
Clrcsr;
For k := 1 to N do
For i := 1 to N do
If A[i,k] > 0 then
For i := 1 to N do
If (A[j,k] > 0) and (i < > j) then
If (A[i,j] = 0) or (A[i,j] > Aakl-hAfkj]) then
Begin
A[i,j] := A[i,k] + A[k,j];
C[i,j] := k;
End;
If A[P,Q] = 0 then Writeln ('Không co duong di tu ', p, ' den ',Q)
Else
Begin
```

Writeln (’Duong di ngan nhat tù’, p, ’ den Q, ’ dai ‘, A[P,Q]);

Write (P);

Xuat (P,Q);

End;

End;

BEGIN

Nhap;

Xuly;

END.

Bàl 6. Có N công trình cần vật liệu thi công. Công trường i cần cung cốp D[i] đơn vị hàng. Hàng được cung cấp từ hai kho A uà B. Cước vận chuyển một đơn vị hàng từ kho A đến còng trường i là A[i]. Cước vận chuyển một đơn vị hàng từ kho B đến công trường i là B[i]. Biết khi A có r đơn vị hàng và tổng số hàng của cả hai kho vừa đủ cung cấp cho N công trường.

Hãy phân phối hàng từ hai kho đến các công trường sao cho tổng cước phí vận chuyển là ít nhất.

Dữ liệu : cho trong file INP.BL6 gồm 4 dòng:

+ Dòng 1 : Chứa 2 số N và r

+ Dòng 2 : Chứa N số D[1J D[2] ... D[NJ

+ Dòng 3 : Chứa N số A[1J A[2J ... A[N]

+ Dòng 4 : Chứa N số B[1J B[2J ... B[N]

Dữ liệu là các số nguyên dương. (N =< 100)

Kết quả : xuất ra file OUT.BL6 gồm 3 dòng:

+ Dòng 1 : Ghi một số nguyên dương là tổng chi phí vận chuyển ít nhất.

+ Dòng 2 : Ghi N số nguyên dương tương ứng số đơn vị hàng mà khi A cung cấp cho từng công trường theo thứ tự.

+ Dòng 3 : Ghi N số nguyên dương tương ứng số đơn vị hàng mà khi B cung cấp cho từng công trường theo thứ tự. Các số ghi trên cùng một dòng ghi cách nhau ít nhát một dấu cách.

VÍ DỤ :

HƯỚNG DẪN GIẢI THUẬT

S = Min { $\sum$ A[i]X[i] + $\sum$ B[i] (D[i] - X[i]) }

1 .. n

1 .. n

S[i] (r) Cước phí nhỏ nhất mà kho 1 phải cung cấp r đơn vị hàng cho các công trường 1 .. i

S[1] (r) = Min {A[1]X[1] + B[l] (D[l] - X[l])}

X[l] <= D[1]

X[l] <= r

S[i+1] (r) = Min {A[i+1]X[i+1] + B[i+1] (D[i+1] - X[i+1]) + S[i] (r — X[i+1]) }

X[i+1] <= D[i+1]

X[i+1] <= r

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_6;
Uses CRT;
Const Max = 100;
Var n,r : Integer;
A,B,D : Array[1.. Max] of Word;
S,L : Array[L.Max, 0..Max] of Word;
i,j,x : Integer;
Procedure Nhap;
Var f:Text;
Begin
Clrscr;
Assign (f,'INP.BL6');
{$I -} Reset (f); {$I +}
Readln (f,n,r);
For i := 1 to n do
Read (f,D[i]);
Readln (f);
For i := 1 to n do
Read (f,A[i]);
Readln (f);
For i := 1 to n do
Read (f,B[i]);
Readln (f);
Close (f);
End;
Functi0n Min (v1, v2 : Word) : Word;
Begin
If v1 < v2 then Min := v1
```

```
Else Min := v2;
End;
Procedure Xuly;
Var  M : LongInt;
Luu : Integer;
Begin
FillChar (S,SizeOf (S), 0);
L := S;
For j := 0 to r do
Begin
M := 1000000;
For x := 0 to Min (j,D[1]) do
If A[1]* x + B[1]* (D[1]-x) < M then
Begin
Luu := x;
M := A[1] x + B[1]* (D[1]-x);
End;
S[1j] := M;
L[1j] := Luu;
End;
For i := 2 to n do
For j := 0 to r do
Begin
M := 1000000;
For X := 0 to Min (j,D[i]) do
If M > A[i]*x +B[i]*(D[i]-x)+S[i-1,j-x] then
Begin
Luu := x;
M := A[i]* x+B[i]*(D[i]-x)+S[i-1, i-x];
End;
S[i, j] := M;
L[i,j] := Luu;
End;
End;
Procedure Xuat;
Var  f : Text;
```

```
Hang : Array[l..Max] of Word;
Begin
Assign(f,'OUT.BL6');
Rewrite(f);
Writeln(f,S[n,r]);
For i := n downto 1 do Begin
Hangii] := L[i,r];
Dec(r,L[i,r]);
End;
For i := 1 to n do
Write(f,Hangti],' ');
Writeln (f);
For i := 1 to n do
Write (f,D[i]-Hang[i],' ');
Close(f);
End;
BEGIN
Nhap;
Xuly,
Xuat;
END.
```

Bài 7 : Một ngân hàng có N loại tiền mệnh giá A[1], A[2], A[3],..., A[N] VỚI số lượng tiền mỗi loại không giới hạn. Còn chi trả cho khách hàng một số tiền là M đồng. Hãy cho biết cần bao nhiêu tiền mỗi loại để chi trả sao cho số lượng tờ là ít nhất. Cho biết : N =<;100; A[i] < 256; Q =< 10000

Dữ liệu : cho trong file INP.BL7 gồm 2 dòng:

+ Dòng 1 : Ghi 2 số N và M

+ Dòng 2 : Ghi N số nguyên dương A[1] A[2] ... A[N]

Dữ liệu cho là các số nguyên dương Kết quả : xuất ra file OUT.BL7 gồm 2 dòng

+ Dòng 1 : Ghi số lượng tờ phải trả

+ Dòng 2 : Ghi N số nguyên khống àm ứng với số tờ cán trả cho mồi loại tiên.

Các số ghi trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

HƯỚNG DẪN GIẢI THUẬT:

- Fx (i) là SỐ lượng tờ ít nhất để trả số tiền

- Fx (i) = Min {Fx (i-A(J]) | i >= A[j]}

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_7;
Uses CRT;
Var  N : Byte;
A : Array[1 ..100] of Byte;
M : Integer;
Fx,L : Array[0..10000] of Word;
i,j : Integer,
Procedure Nhap;
Var  f : Text;
Begin
Assign(f,'INP.BL7);
Reset(f);
FillChar (A,SizeOf(A),0);
Readln(f,n,m);
For i := 1 to n do
Read(f,A[i]);
Close(f);
End;
Procedure Xuly;
Begin
FillChar (Fx,SizeOf(Fx),0);
L := Fx;
For i := 1 to M do
Begin
For j := 1 to N do
If A[j] <= i then
If (Fx[i] = 0) or (Fx[i-A[j + 1 < Fx[i]) then
Begin
Fx[i] := Fx[i-A(j]] + 1;
L[i] := j;
End;
End;
End;
Procedure Xuat;
```

```
Var f:Text;
S1 : Array[1..100] Of Word;
Begin
As8ign(f,'OUT.BL7');
Rewrite(f);
FillChar(Sl,SizeOf (S1),0);
Repeat
Inc (Sl[L[M]]);
M := M-A[L[M]];
Until Fx[M] = 0;
Writeln(f,Fx[i]);
For i := 1 to N do
Write(f,Sl[i],' ');
Close(f);
End;
BEGIN
Nhap;
Xuly;
Xuat;
END.
```

Bài 8 : Quầy ăn của một khách sạn cần sử dụng D[1], D[2],..., D[N] khăn trải bàn cho N ngày liên tiếp đánh số từ 1 đến N. Khách sạn có thể mua khăn trải bàn mới với giá là A đồng một khăn, hoặc thuê hiệu giặt, trả nhanh (nhận lại khăn giặt sạch vào ngay hơm sau) với giá B đồng một khăn, hoặc thuê hiệu giặt trả chậm (khăn dùng trong ngày được gởi giặt và trả lại vào ngày i + 2) với giá C đồng một khăn. Giả sử trong ngày 1 khách sạn chưa có khăn. Hãy lập kế hoạch mua-giặt khăn bảo đảm yêu cầu về khăn cho N ngày với chi phí nhỏ nhất.

Dữ liệu: cho trong file INP.BL8 gồm 2 dòng

+ Dòng 1: gồm 4 số nguyên dương N, A, B , C, D (N<100, A>B>C>0)

+ Dòng 2: gồm N số nguyên dương D[1], D[2]… D[N]

Các số trên cùng một dòng ghi cách nhau ít nhất một dán cách.

Kồ (Ịuà :Xuất ro fileOUT.BL8 gôm N + 1 dòng;

+ Dòng 1:Ghi tổng chi phí nhỏ nhát + Dòng i + 1 (I<iíN) ; Ghi 3 số nguyên khống Ồm

F[i] S[i] theo thứ tự là số khăn cồn mua, giặt trả nhanh, giặt trả chậm trong ngày i

HƯỚNG DẪN GIẢI THUẬT

— MaxD = Max {D[i]}

— somax := MuxD

— For i := n - 1 down to 1 do

Tong[i] := Tong[i + 1] + D [i + 1]

— Repeat

Gán mảng M, G1, G2 bằng 0

Stop := True

du := so max; m[1] := du;

t1 := 0; t2 := 0;

giá := a*du;

For i := 1 to n do

+ j := du + t1 + t2;

+ Nếu j - d[i] < tong[i] thì

Nếu j - d[i] + g2[i - 1] < d[i + 1] thì g1[i] := d[i + 1] + d[i] -j - g2[i - 1] h := j + g2[i - 1] - d[i] + gl[i] - d[i + 1];

Nếu h < tong[i + 1] thì

g2[i] := min(d[i] – g1[i], tong(i + 1] - h) + t2 := g2[i - 1]; tl := gl[i];

+ dư := j - d[i];

+ giá := giá + b* gl[i] + c* g2[i];

+ Nếu giá >= giá min thì Thoát

stop := false;

Nếu stop = False thi ghi nhận kết quả

Until Stop hay (Somax > tong[1] + D[1])

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_8;
UsesCRT;
Conts max = 100;
fname = ’b11.inp’;
limit = 1000000000;
Type Mang = array[0..max] of longint;
Var a,b,c,gia, n, giamin : longint;
somax, i : longint;
tong,d, m, gl, g2 : mang;
mo, gol, go2 : mang;
stop : boolean;
```

```
f : text;
Procedure Nhap;
Var i, j : Byte;
Begin
Assingn (f,’INP.BL8’); Reset(f);
Readln (f, n, a, b , c); For i := 1 to n do
Begin
Read(f, d[i]);
End;
Close(f);
End;
Function    MaxD : Longint;
Var  i; j : Longint;
Begin
i := d[l];
For j := 2 to n do
If d[j] > i then i := d[j];
Maxd := i;
Procedure Timtong;
Var  i, j : Longint;
Begin
For i := n - 1 downto 1 do
tong[i] := tong[i + 1] + d[i + 1];
End;
Procedure Khoitao;
Begin
FillChar(m,Sizeof(m), 0);
FillChar (gl, Sizeof(g1), 0);
FillChar (g2, Sizeof (g2), 0);
Stop := true;
Gia := 0;
End;
Function Min (i, j : Longint) : Longint;
Begin
Min := i;
If i < j then exit;
```

```
Min := j;
End;
Procedure Xuly;
Var  i,j,du,t1,t2,h : Longint;
Begin
du := somax; m[1] := du;
t1 := 0; t2 := 0;
gia := a*du;
For i := 1 to n do Begin
j := du + tl + t2;
If j - d[i] < tong[i] then
Begin
If j-d[i]+g2[i-1]<d[i+1] then gl[i] :=d[i+1]+d[i]-j-g2[i-1];
h := j+g2[i-1]-d[i]+g1[i]-d[i+1];
If h < tong [i + 1] then
g2[i] := min (d[i] – g1[i], tong[i + 1] - h);
End;
t2 := g2 [i - 1]; t1 := g1[i]; du := j - d[i];
gia := gia + b* gl[i] -h c* g2[i];
If gia >= giamin then exit;
End;
stop := false;
End;
Procedure Ghinhan;
Begin
giamin := gia;
go1 := g1;
go2 := g2;
mo := m;
End;
Procedure Chuanbi;
Begin
FillChar (tong, sizeof (tong), 0);
somax := maxd;
Timtong;
giamin := limit;
```

```
End;
Procedure Print;
Var  i : longint;
Begin
Assign (f, 'OUT.BL8');
Rewrite(f);
Writeln (f,giamin);
* For i := 1 to n do
Writeln (f, mo[i] : 4, go l[i] : 4, go2[i] : 4);
Close(f);
End;
BEGIN
Nhap;
Chuanbi;
Repeat
Khoitao;
Xuly;
If n0t stop then Ghinhan;
Inc(somax);
Until (somax > tong[1] + d[1]) or (stop);
Print;
END.
```

Bài 9 : Trong ngày sinh nhật, hai anh em Tom vầ Alice nhận được N đồ chơi (N<40). Trên đồ chơi i có ghi giá tiên Xi. Hai anh em quyết định mỗi người phải có trách nhiệm bảo quản một phần số quà và phân chia sao cho chênh lệch tổng giá trị tiền đồ chơi mà mỗi người phải bảo quản là ít nhất. Hãy giúp Tom và Alice phân chia trách nhiệm.

Dữ liệu : cho trong file INP.BL9 gồm 2 dòng:

+ Dòng 1 : số nguyên dương N

+ Dòng 2 : N số nguyên dương X1…,X2,… Xn

Kết quả: xuất ra file OUT.BL9 gồm 2 dòng: Mỗi dòng ghi số tương ứng với các đồ chơi của một người và cuối mỗi dòng là tổng giá trị các đồ chơi tương ứng.

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT:

— Gọi A là nửa tổng giá trị của N đồ chơi

— Gọi Fx(i, j) là tổng giá trị lớn nhất trong trường hợp có i vật từ 1 đến i và giá trị tối đa là j. (nghĩa là tổng giá trị của các vật được chọn không quá j)

— For i := 1 to n do

For j := 1 to A do

Nếu j >= X[i] thì

Fx[i, j] := Max(Fx[i . 1, j . X[i + X[i], Fx[i - 1, j])

Nếu không thì Fx[i, j] := Fx[i - 1, j]

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_9;
UsesCRT;
Const Max N = 40;
Max Val = 500;
Var  X, Y : Array[l..MaxN] of Word;
Fx : Array[0..MaxN, 0..Max  Val] of Word;
n, i, j : Integer;
A, B : Word;
Procedure Nhap;
Var  f : Text;
Begin
Clrscr;
Assign (f,'INP.BL9');
Reset(f);
Readln(f, n);
FillChar(X,SizeOf(X), 0);
Y := X;
For i := 1 to n do
Read(f, X[i]);
Close(f);
End;
Function GetMax (v1, v2 : Word) : Word;
Begin
If v1 > v2 then GetMax := v1
Else GetMax := v2;
End;
Procedure Xuly;
Begin
```

```
B := 0;
For i := 1 to N do
B := B + X[i];
A := B div 2;
FillChar (Fx, Sizeof(Fx), 0);
For i := 1 to n do
Begin
If j > = X[i] then
Fx [i, j] := GetMax (Fx[i - i, j -X[i + X[i], Fx[i - 1, j])
Else
Fx[i, j] := Fx [i - 1, j];
End;
End;
Procedure Xuat;
Var  Max,A0, n0, V1 : Word;
f : Text;
Begin
Assign (f, 'OUT.BL9');
Rewrite(f);
Max := 0; A0 := 0; n0 := 0;
For i := 1 to A do
If Max <= Fx[n, i] then Begin
Max := Fx[n, i];
A0 := i;
End;
VI:« A0;
P'or i := n down to 1 do
If Fx[i, A0] <> Fx[i - 1, A0] then Begin Inc(nO);
Y[i] := 1;
A0 := A0 - X[i];
End;
For j := 0 to 1 do
Begin
For i := 1 to N do
If Y[i] = j then Write(f, i,' ');
If j = 0 then Writeln (f,B - A)
```

```
Else Writeln (f,A);
End;
Close(f);
End;
BEGIN
Nhap;
Xuly;
Xuat;
END.
```

Bài 10 : Có N địa điểm dân cư đánh số từ 1 đến N. Giữa M cặp địa điểm trong số N địa điểm nói trên có tuyến đường nói chung. Cần xây dựng một trung tâm dịch vụ tổng hợp tại một địa điểm hoặc là trùng với một trong số các địa điểm dân cư hay là nằm trên tuyến đường nối hai địa điểm nào đó, sao cho tổng khoảng cách từ trung tàm dịch vụ đến N địa điểm dân cư là nhỏ nhất. Ta gọi khoảng cách giữa hai địa điểm là độ dài đường đi ngắn nhất nối chúng. Già sử rằng N địa điểm trên liên thông với nhau.

Dữ liệu : cho trong file INP.BL10 gồm M + 1 dòng:

+ Dòng 1 chứa hai số N và M

+ Dòng i + 1 (1=< i =<M) ghi 3 số nguyên dương : hai số đầu là chỉ số của hai địa điểm dân cư được nối nhau bởi tuyến đường này còn số thứ ba là độ dài của tuyến đường.

Dữ liệu là các số nguyên dương.

Kết quả : Xuất ra màn hình thông báo vị trí trung tâm dịch vụ là tổng khoảng cách từ trung tâm dịch vụ đến các địa điểm dân cư.

Nếu điểm tìm được nòm trên tuyến đường thì cần chỉ rõ hai đầu của tuyến đường và khoảng cách từ địa điểm xây dựng đến đầu thứ nhất.

HƯỚNG DẪN GIẢI THUẬT .

— Tính khoảng cách ngắn nhất giữa mọi cặp địa điểm

— Trung tâm dịch vụ đặt tại địa điểm có tổng khoảng cách các địa điểm khác là nhỏ nhất.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_4_10;
Uses CRT;
Var  A, C : Array[1..50, 1..50] of Integer;
N, p, Q, M : Integer; i, j, k : Integer,
Luu, Min : Integer;
```

```
Procedure Nhap;
Var f:Text;
x,y,z:Byte;
Begin
Assign (f,'INP.BL0');
Reset(f);
Readln(f,N,M);
FillChar(A,SizeOf(A),0);
C := A;
For i := 1 to M do Begin
Readln(f, x, y, z);
A[x, y] := z;
A[y, x] := z;
End;
Readln(f, P, Q);
Close(f);
End;
Procedure Xuly;
Var  D : Integer;
Begin
Clrscr;
For k := 1 to N do For i := 1 to N do If A[i, k] > 0 then For j := 1 to N do
If (A[j, k] > 0) and (i < > j) then
If (A[i,j]=0) or (A[i,j] >A[i,k]+A[kj]) then
Begin
A[i,j] := A[i, k] + A[kj];
C[i, j] := k;
End;
Luu := 0;
Min := MaxInt;
For i := 1 to N do
Begin
D := 0;
For j := 1 to N do
Begin
If i <> j then D := D + A[i, j];
```

```
If (i < > j) and (A[i, j] = 0) then
Begin
D := MaxInt;
j := N;
End;
End;
If D < Min then
Begin
Luu := i;
Min := D;
End;
End;
Writeln ('Trung tam dich vu dat tai dia diem', Luu);
Write ('Tong khoang each la ', Min);
Readln;
End;
BEGIN
Nhap;
Xuly;
END
```

§5. ĐỒ THỊ (GRAPH)

1. Khái niệm.

Trong nhiều tình huống, chúng ta thường vẽ những sơ đồ, gồm những điểm biểu thị các đối tượng được xem xét (người, tổ chức, đội bóng, thành phố ...) và nối một số điểm với nhau bằng những đoạn cong hoặc thẳng tượng trưng cho quan hệ nào đó giữa các đối tượng. Các sơ đồ như vậy được dùng ở khắp nơi : Sơ đồ mạng điện, sơ đồ tổ chức, sơ đồ giao thông... Đó là những ví dụ về Graph.

Một Graph là một tập hợp hữu hạn các điểm (Gọi là đính của graph) cùng với tập hợp các đoạn đường cong hay thảng (gọi là cạnh của graph) có các đầu mút tại các đỉnh của graph.

2. Biểu diễn Graph

Có nhiều cách biểu diễn Graph, ở đây ta chỉ xét một cách biểu diễn đơn giản đó là dùng một mảng vuông 2 chiều (ma trận) có kiểu phần tử là Boolean:

Trong đó:

* n : chỉ số đỉnh của Graph.

* A[i,j] := TRUE (1) nếu giữa đỉnh i và đỉnh j có cạnh nối.

* A[i,j] := FALSE (0) nếu giữa đỉnh i và đỉnh j không có cạnh nối.

Ví dụ:

A[1..n, 1..n] of Boolean.

A thường được gọi là ma trận nối của Graph G.

Ví dụ:Graph

Có ma trận nối là:

— Nếu tồn tại i, j sao cho A[i,j] khác A[j,i] thì graph G sẽ được coi là có hướng.

Ví dụ : Graph có ma trận nối là:

— Nếu có độ dài 1ij của đường đi từ đỉnh i đến đỉnh j thì ta có thể lập thêm ma trận độ dài L[1..n, 1..n] như sau:

L[i,j] := lij

3. Kiểm tra xem có đường đi từ đỉnh Đ đên đỉnh C trên một đồ thị có hướng hay không

a) Bài toán

— Cho đồ thị có hướng G có ma trận nối là A và 2 đỉnh Đ, C của G. Hãy tìm xem có đường đi từ Đ đến C không.

b) Thuật toán

— Dùng mảng C[l..n] of Boolean để đãnh dấu các đỉnh đã đi qua.

— Dùng procedure lantu(i) một cách đệ qui để lan từ đỉnh Đ đến đỉnh một đỉnh nào đó có thể được, rồi từ đó lại lan tiếp đến các đỉnh khác chưa lan tới cho đến khi gặp đỉnh C thì biến Boolean kq cho kết quả true nghĩa là có đường đi.

c) Thể hiện bằng PASCAL

```
Procedure Lantu (i : byte);
Var j : byte;
Begin
If (i = c) then kq := True
Else
Begin
For j := 1 to n do
If A[i,j] and n0t c[j] then Begin
C[j] := true;
Lantu (j);
End;
End;
End;
```

```
Begin {chương trình chính}
Kq := False;
For j : = 1 to n do C[j] := False;
Lantu(D);
If kq then Writeln (’co duong đi’)
Else
Writeln (‘khong co’)
End.
```

4. Tìm đường đi ngắn nhất

a) Bài toán

Tìm đường đi ngắn nhất từ đỉnh nguồn (1) đến một đỉnh khác (j) của một đồ thị có hướng G có ma trận nối là A và ma trận độ dài L chứa chiều dài các cung:

L[i,i] = 0 với mọi i = 1,...n.

L[i,j] >= 0 nếu tồn tại cung từ đỉnh i đến đỉnh j.

L[i,j] = [vô cực] nếu không tồn tại cung từ đỉnh i đến đỉnh j.

b) Thuật toán (Dijkstra)

— Nguyên lí chính của thuật toán này là : Nếu tồn tại một đường đi ngắn nhất từ đỉnh i đến đỉnh j và đỉnh k ở trên đường đi này thì k sẽ chia đường đi thành 2 đoạn và 2 đoạn đó phải là 2 đoạn ngắn nhất nối từ i đến k và từ k đến j.

— Ta dùng 2 mảng C và S

C : chứa các đỉnh chưa được chọn

S : chứa các đỉnh đã được chọn.

— Tại mỗi thời điểm tập S chứa các đỉnh mà khoảng cách nhỏ nhất từ đỉnh nguồn đến, chúng đã được xác định tập C chứa các đỉnh còn lại.

— Bắt đầu S = {Đỉnh nguồn}

Kết thúc S = tập các đỉnh của G

Tại mối bước ta chọn một đỉnh x của C mà khoảng cách D[x] từ đỉnh nguồn đến X là nhỏ nhất.

— Ta nói rằng đường đi từ đỉnh nguồn đến một đỉnh khác là độc biệt nếu tất cả các đỉnh trung gian trên đường đi này đều ở trong tập S.

— Tại mỗi bước của thuật toán có mảng một chiều D chứa chiều dài của đường đi đặc biệt ngắn nhất đến mối đỉnh của đồ thị.

— Mỗi lần ta đưa đỉnh X mới vào s, đường đi riêng biệt ngắn nhất đến X cũng chính là đường đi ngắn nhất trong tất cả các đường đi dến X.

— Khi thuật toán kết thúc, tất cả các đỉnh của G đều thuộc s suy ra mọi đường đi từ nguồn đến các đỉnh đều là đặc biệt.

c) Thể hiện bằng Pascal

```
Procedure Shortestpath;
Var D, P, C : Array [1.. n] of integer;
i, j, k, mk, rax, min, X : integer;
Begin
K := n — 1; {số phần tử của C}
{Xây dựng các mảng C, D và P}
For i := 2 to n do
Begin
C[i -1] = i;
D[i] = L[1,i];
P[i] = 1;
End;
For j := 1 to n - 2 do
Begin {tìm giá trị nhỏ nhất của mảng D}
Min := D[C[1]];
mx := 1;
For i := 2 to k do
If d[C[i]] < Min then
Begin
Min := D[C[i]]; mx := i End;
{loại bỏ đỉnh C[mx] ra khỏi C}
mk := C[mx]; C[mx] := C[k];
k := k -1;
For i := 1 to k do
Begin
X := C[i];
If D[x] > D[mx] + L[mk, x] then
Begin
D[x] := D[mx] + L[mk, x];
P[x] := mk;
End;
End;
End;
```

Để chỉ ra đường đi ngắn nhất từ đỉnh nguồn đến đỉnh j ta có thủ tục sau:

```
Procedure Path (x : integer);
```

```
Begin
If X < > 1 then
Begin
Path(p[x]);
Write(x:3);
End;
End;
```

Khi sử dụng ta chỉ viêc gọi Path(j);

5. Những bài toán cơ bản về GRAPH:

Bài 1 : Viết chương trình kiểm tra tính liên thông của 1 đồ thị vô hướng A[i, j] với A[i, j] = 1 nếu có đường đi từ i tới j

A[i, j] = 0 nếu không có đường đi từ i tới j

Dữ liệu cho trong file INP.BL1

+ Dòng đầu ghi số n là số đỉnh của đồ thị (0 < n < 100)

+ Dòng i + 1 (1 <i< n) chứa n số A[i, 1] A[i, 2] ... A[i, n]

Thông báo kết quả ra màn hình : "Đồ thi liên thông" hav "Đồ thị không liên thông"

HƯỚNG DẪN GIẢI THUẬT:

- Ta đánh dấu đỉnh số 1 của đồ thị.

— Từ một đỉnh i đã đánh dấu, ta đánh dấu đỉnh j nếu A[i, j] >0 và đỉnh j chưa đánh dấu. Cứ thực hiện như vậy đến khi không thực hiện được nữa.

— Nếu còn đỉnh nào chưa đánh dấu thì đồ thị không liên thông và ngược lại.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_1;
Const Max = 100;
Var  n,i,j : Byte;
A : Array[1..Max, 1..Max] of Byte;
L : Array [1..Max] of Boolean;
Procedure Nhap;
Var  f : Text;
Begin
Assign (f, 'INP.BLD);
Reset (f);
Readln (f,n) : FillChar(ASizeOf(A),0);
For i := 1 to n do
Begin
```

```
For j := 1 to n do Read (f,A[i,j]);
Readln(f);
End;
Close (f);
End;
Function  Lienthong: Boolean;
Var  xong, lt: Boolean;
Begin
FillChar(L, SizeOf(L), 0);
L[1] := True;
Repeat
xong := True;
For i := 1 to n do
If L[i] then
For j := 1 to n do
If (A[i,j]<>0) and (not L[j]) then
Begin
L[j] := True; xong := False;
End;
Until xong;
Lienthong := False;
For i := 1 to n do
If not L[i] then Exit;
Lienthong := True;
End;
BEGIN
Nhap;
If Lienthong then
Writeln ('DO THI LIEN THONG')
Else
Writeln ('DO THI KHONG LIEN THONG');
Readln;
END.
```

Bài 2 : Viết chương trình đếm số thành phần tính liên thông của 1 đồ thi vô hướng A[i, j] với

A[i, j] = 1 nếu có đường đi từ i tới j

A[i, j] = 0 nếu không có đường đi từ i tới j

Dữ liệu cho trong file INP.BL2

+ Dòng đầu ghi số n là số đỉnh của đồ thị (0 < n < 100)

+ Dòng i+1 (1 =< i =< n) chứa n số A[i,1] A[i,2] ... A[i,n] Thông báo kết quả ra màn hình.

HƯỚNG DẪN GIẢI THUẬT:

1. Ta đãnh dấu một đỉnh chưa được đãnh dấu.
2. Từ một đỉnh i đã đãnh dấu, ta đãnh dấu đỉnh j nếu A[i,j] >0 ri đỉnh j chưa đãnh dấu. Cứ thực hiện như vậy đến khi không thực lộn được nữa
3. Trở lại bước 1 cho đến khi mọi đỉnh đều được đãnh dấu.
4. SỐ thành phần liên thông là số lần thực hiện bước 1.

HƯƠNG TRÌNH MAƯ

```
PROGRAM CHUON G_TRINH_MAU_B AI_5_2;
Const       Max = 100;
Var  n,i,j,k : Byte;
A : Array[1..Max, 1..Max] of Byte; L : Array[1..Max] of Byte;
Procedure Nhap;
Var  f : Text;
Begin
Assign (f,TNP.BL2'); Reset(f);
Readln(f,n);
FillChar(A,SizeOf(A),0);
For i := 1 to n do Begin
For j := 1 to n do Read(f,A[i,j]);
Readln(f);
End;
Clooe(f);
End;
Function   Dem : Byte;
Var  xong,th : Boolean;
D : Byte;
Begin
FillChar(L,SizeOf(L),0);
D := 0;
Repeat
xong := True;
```

```
For i := 1 to n do
If L[i] = 0 then
Begin Inc(D);
L[i] := D;
Repeat
th := True;
For j := 1 to n do
If L[j] = D then
Begin
For k := 1 to n do
If (A[j,k]<>0) and (L[k]=0)
then
Begin
th := False;
L[k] := D;
End;
End;
Until th;
End;
Until xong;
Dem := D;
End;
BEGIN
Nhap;
Writeln('DO THI CO' Dem,'THANH PHAN LIEN THONG'); Readln;
END.
```

Bài 3 : Có N thành phố. Biết hng đường đi giữa hai thành phố bất kì (nếu có) đêu là đường đi hai chiều. Sơ đồ mang lưới giaữ thòng của N thành phố này cho bởi ma trộn kề A[i,jJ trong đó

+ A[i,j] là độ dài đường đi từ thành phó i đến thành phố j
+ Afi,j] = 0 nếu khống cỏ đường đi từ thành phó i đến thành phốj
+ Afi,j] = A[j,i]
+ A[i,i] = 0
+ A[i,j] nguyên, khống àm

Hãy xác định đường đi ngắn nhất giữa hai thành phố P và Q hay thông báo không tồn tại lời giải.

Dữ liệu : Cho trong file INP.BL3 gồm N+2 dòng

+ Dòng đầu chứa số N (N nguyên dương, N =< 50)

+ Dòng i+1 (1 =< I =< N) ghi N số A[i,1] A[i,2] .. A[i,N]

+ Dòng N+2 ghi hai số P và Q.

Các số ghi trên cùng một dòng ghi cách nhau ít nhất một dấu cách

Kết quả : xuất ra màn hình.

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT

- Kc[i] là chiều dài đường đi ngắn nhất từ P đến i
- Pre[i] là đỉnh đứng trước i trên đường đi ngắn nhất từ P đến i
- Kc[P] := 0; Pre[P] := 0;
- Kc[j] := Infinitive với j khác p
- List := [P]
- Trong khi List khác rỗng

+ Lấy 1 phần tử i ra khỏi LIST

+ Nếu Kc[j] > Kc[i] + A[i,j] thì

Kc[j] := Kcli] + A[i,j]

Pre[j] := i;

Nếu j không thuộc LIST thì LIST := LIST+{j}

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRỊNH_MAU_BAI_5_3
Uses CRT;
Var  A,C:Array[1..50,1..50] of Integer;
N,P,Q : Integer;
i,j,k : Integer;
Proceduce Nhap;
Var  f;Text;
Begin
Assign(f,'INP.BL3');
Reset (0;
Readln(f,N);
FillChar(A; SizeOf(A),0);
C := A;
For i := 1 to n do
```

```
Begin
For j := 1 to n do
Read(f,A[i,j]);
Readln(f);
End;
Readln(f,P,Q);
Close(f);
End;
Var Pre,Kc:Array[1..50] of Word;
W:Array[1..50] of Byte;
xong : Boolean;
Procedure Xuly;
Begin
Clrscr;
FillChar (Kc,SizeOf(Kc), $FF);
Pre := Kc;
Kc[P] := 0;
Repeat
xong := True;
For i := 1 to N do
If Kc[i]< >$FFFF then
For j := 1 to N do
If (A[i,j]>0) and (Kci,j]>Kc[i]+A[i,j]) then
Begin
Kc[j] := Kc[i]+A[i,j];
Pre[j] := i;
xong := False;
End;
Until xong;
End;
Procedure Xuat;
Begin
FillChar(W,SizeOf(W),0);
If Pre[Q]=$FFFF then
Writeln('Khong co duong di tu',P,'den',Q)
Else
```

```
Begin
j : =1;
i :=  Q;
W[j]  := i;
Repeat
i := Pre[i];
Inc(j);
W[j] := i;
Until Pre[i] = $FFFF;
For i := j down to 2 do
Write (W[i],’ -> ’);
Writeln (Q);
Writeln(’Do dai duong di la ’,Kc[Q]);
End;
Readln;
End;
BEGIN
Nhap;
Xu ly;
Xuat;
END.
```

Bài 4: Một lưới giao thông hai chiều giữa n địa điểm được cho bởi ma trận A[i,j] trong đó A[i,j]=1 nếu địa điểm i nối với địa điểm j còn A[i,j]=0 trong trường hợp ngược lại. Một người đưa thơ cần đi qua tất cả các con đường này để phát thơ, mỗi đường chỉ đi qua một lần. Hãy xác định lộ trình của người này hay thông báo không tồn tại đường đi như vậy.

Dữ liệu : cho trong file INP.BL4 gồm n+1 dòng

+ Dòng 1: Ghi số nguyên dương n (n=<20)

+ Dòng i+1 (1 si<n) : ghi n số nguyên không âm A[i,1] A[i,2]... A[i.n]

Kết quả: xuất ra màn hình.

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT.

— Điều kiện : Đồ thị hoặc không có hoặc chỉ có 2 đỉnh bậc lẻ.

— Trường hợp mọi đỉnh đồ thị đều có bậc chẵn thì ta luôn được lời giải (cứ đi theo các cạnh của đồ thị cho đến hết (trong trường hợp đồ thị liên thông).

— Trường hợp có 2 đỉnh bậc lẻ, ta thêm vào đồ thị một C giả nối liến hai đỉnh này và trở lại trường hợp trên. Sau khi có đường đi thì ta xóa bỏ đường đi giả và có kết quả.

CHƯƠNG TRÌNH NẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_4;
UsesCRT;
Const MaxN = 20;
Var  A : Array[1.. MaxN, 1..MaxN] of Byte;
B, C : Array[1.. MaxN*(MaxN+1) div 2] of Byte;
D, 1 : Byte;
Lưu : Array[1..2] of Byte;
xong : Boolean;
n, i, j : Byte;
Procedure Nhap;
Var  f : Text;
Begin
Assign(f,'INP.BL4');
Reset (f);
Readln(f,n);
FillChar(A, SizeOf(A), 0);
For i := 1 to n do
Begin
For j := 1 to n do
Readif, Ati,j]);
Read ln(f);
End;
Close(f);
End;
Procedure Xuly;
Begin
FillChar(B, SizeOf (B),0);
1 := 0;
FillChar(Luu, SizeOf(Luu),0)
For i := 1 to n do
Begin
D := 0 ;
```

```
For j := 1 to n do
D := D + A [i, j] ;
If odd (D) then
Begin Inc (1) ;
If 1 = 3 then Begin
Writeln (’Khong co duong đi’) ;
Readln ;
Halt ;
End ;
Luu 01 := i ;
End ;
End ;
If 1 = 2 then
Begin
Inc (A [Luu [11, Luu [2]]) ;
Inc (A [Luu [2], Luu [1]]) ;
End ;
If 1 = 2 then
B [1] := Luu [1]
Else B [1] := 1 ;
D := 1 ;
Repeat
Xong : = True ;
For i := 1 to N do
If A [i, B [D]] > 0 then
Begin
Dec (A [i, B [D]]) ;
Dec (A [B, [D], i]) ;
Inc (D) ;
B [D] := i ;
Xong := False ;
Break ;
End ;
Until xong ;
If 1 = 2 then Begin i := 0 ;
Repeat
```

```
Inc (i) ;
Until A[B [i], B [i + 1]] = A[Luu [1], Luu [2]] ;
Move (B [i + 1] , C [1], D - i) ;
Move (B [1] , C [D - i + 1], i) ;
B := C;
Dec (D) ;
End ;
Clrscr ;
Writeln ('Duong di B [1]) •
For i := 2 to D do
Writeln (->' , B [i]) ;
Writeln ;
If B [1] = B [D] then
Writeln ("Day la mot chu trình') ;
Readln ;
End ;
BEGIN
Nhap ;
Xu ly ;
END.
```

Bài 5 : Một người khách du lịch muốn đi thăm n thành phố được đánh số từ 1 đến n. Mạng lưới giao thông giữa n thành phố này là hai chiều và được cho bởi ma trận A [i, j] trong đó A [i, j] = nếu có đường đi giữa thành phố i và thành phố j, A [I, j] = 0 trong trường hợp ngược lại.

Hãy thiết lập lộ trình cho người khách hay thông báo không tồn tại lời giải.

Dữ liệu : cho trong file INP.BL5 gồm n + 1 dòng

+ Dòng 1 : Ghi số nguyên dương n(n < 20) .

+ Dòng i + 1(1 < i < n) : ghi n số nguyên không âm A [i, 1] A [i, 2] ... A [i, n]

Kết quả : xuất ra màn hình.

HƯỚNG DẪN GIẢI THUẬT

— Tìm hết tất cả mọi khả năng cùa đường đi (sau khi di qua đường đi nào thì xóa bỏ đường đi đó) và kiểm tra xem đường đi này có qua đủ N đỉnh của đô thị hay không.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_5 ;
```

```
Uses CRT ;
Const MaxN = 20 ;
Var  A : Array [1.. MaxN, 1.. MaxN] of Byte ;
B : Array [1.. MaxN] of Boolean ;
C : Array [1.. MaxN] of Byte ;
n, i, j : Byte ;
Procedure Nhap ;
Var  f : Text ;
Begin
Assign (f, 'INP.BL5') ;
Reset (f) ;
Readln (f, n) ;
FillChar (A, SizeOf (A), 0) ;
For i := 1 to n do
Begin
For j := 1 to n do Read (f, A [i, j]) ;
Readln (f) ;
End ;
Close (f) ;
End ;
Procedure Test;
Begin
Clrscr ;
If A [C [1], C [N]] > 0 then
Begin
Writeln ('Chu trinh HAMILTON') ;
For i := 1 to N do Write (C [i], '->') ;
Write (C [1]) ;
Readln ;
Halt ;
End ;
End ;
Procedure Tim (VT : Byte) ;
Var  i : Byte ;
Begin
If VT = N + 1 then Test
```

```
Else
For i := 1 to N do
If (A [C [VT - 1], i] > 0) and (not B [i]) then
Begin
A [C [VT - I], i] := 0 ;
C [VT] := i ;
B [i] := True ;
Tim (VT + 1) ;
A [C [VT - 1], i] := 1 ;
C [VT] := 0 ;
B [i] := False ;
End ;
End;
Procedure Xuly ;
Begin
FillChar (B, SizeOf (B), 0) ;
Move (B, c, SizeOf (B)) ;
C [1] := 1 ;
B [1] := True ;
Tim (2) ;
Writeln (’Khong ton tai loi giai’) ;
End ;
BEGIN
Nhap ;
Xu ly ;
END.
```

Bài 6. Có n thành phố được đánh số từ 1 đến n. Mạng lưới giao thông giữa các thành phố là các đường một chiều. Trên đường đi (nếu có) từ thành phố j đến thành phố người ta không được mang quá A [i,j] đơn vị hàng. Nếu không có đường đi từ thành phố i đến thanh phố j thì xem như A [i, j] = 0. Cần vận chuyển hàng từ thành phổ s đến thành phố d. Hãy lập kế hoạch vận chuyển sao cho tổng khối lượng hàng vận chuyển là nhiều nhất.

Dữ liêu: cho trong file INP.BL6 gồm n + 1 dòng

+ Dòng 1 : gồm 3 số nguyên dương n, s, d

+ Dòng i + 1(1 < i < n) : ghi n số A [i,1], A [i,2], ... , A[i,n]

Kết quả : xuất ra màn hình gồm n + 1 dòng

+ Dòng 1 : ghi tổng lượng hàng vận chuyển.

+ Dòng i+1(1 < i < n) ; ghi n số F[i,1], F[i,2],... F trong đó F [i, j] ứng với lượng hàng vận chuyển từ thành phố i thành phố j.

Các số trên cùng một dòng ghi cách nhau ít nhất một dấu cách

VÍ DỤ:

HƯỚNG DẪN GIẢI

1. Gán nhãn 1 cho đỉnh s

Sau đó, nếu i là một đỉnh đã có nhãn thì

+ Gán nhãn +i cho tất cả các đỉnh j chưa có nhãn và F(i, j) < A(i, j)

+ Gán nhãn -i cho tất cả các đỉnh j chưa có nhãn và F(j, i) > 0

Lập lại công việc trên cho đến khi không thể gán nhãn được hay đỉnh d đã có nhãn.

2. Nếu d có nhãn thì ta xác định một dây chuyền từ s đến d bằng cách lần ngược các nhãn. Với mỗi cung (i, j) thuộc dây chuyền

F(i, j) := F(i, j) + 1 nếu (i, j) hướng từ s đến d

F(i, j) := F(i, j) - 1 nếu (i, j) hướng từ d đến s

3. Nếu d không có nhản thì dừng, nếu ngược lại thì trở lại bước 1.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_6 ;
UsesCRT ;
Const MaxN = 50;
Var  A, F : Array [1.. MaxN, 1.. MaxN] of Byte ;
L : Array [1.. MaxN] of Integer ;
w : Array [1.. MaxN] of Byte ;
n : Byte ;
s, d : Byte ;
i, j : Integer ;
Xong : Boolean ;
Tong, dt : Integer ;
Procedure Nhap ;
Var  g : Text     ;
Begin
Assign (g, 'INP.BL6') ;
Reset (g) ;
Readln (g, n, s, d) ;
FillChar (A, SizeOf (A), 0) ;
```

```
F := A ;
For i := 1 to n do
Begin
For j := 1 to n do
Read (g, A [i, j]) ;
Readln (g) ;
End ;
Close (g) ;
End ;
Functi0n Min (xl, x2 : Integer) : Integer ; Begin
If x1 < x2 then Min := x1
Else Min := x2 ;
End ;
Procedure Xuly ;
Begin
Tong := 0 ;
Repeat
For i := 1 to n do
L [i] := MaxInt ;
L [s] := 0 ;
Repeat
Xong := True ;
For i := 1 to n do
If L [i] < > MaxInt then
Begin
For j := 1 to n do
If L [j] = MaxInt then
Begin
If (A [i, j] > 0) and
(F [i, j] < A [i, j]) then
Begin
L [j] := + i ;
Xong := False ;
End ;
If (A [j,i] > 0) and (F [j,i] > 0) then
Begin
```

```
L [j] := -i ;
Xong := False ;
End ;
If L [d] < > MaxInt then
Begin
j := n ; i := n ;
Xong := True ;
End ;
End ;
If L [d] < > MaxInt then
Begin
i := n ;
Xong := True ;
End ;
End ;
Until xong ;
If L [d] < > MaxInt then
Begin
dt := MaxInt ;
FillChar (W, SizeOf (W), 0) ;
i := 1
W [1] := d ;
Repeat
Inc (i) ;
W [i] := Abs (L [W [i - 1]]) ;
If L [W [i - 1]] >= 0 then
dt := Min (dt, A[W [i], w [i-1]] - F [W[i], W [i-1]]);
If L [W [i - 1]] < 0 then
dt := Min (dt, F [W [i - 1], W [i]]) ;
Until w [i] = s ;
Inc (tong, dt) ;
For j := 2 to i do
Begin
If L [W j - 1]] >= 0 then
Inc (F [W [j], W[j - 1]], dt) ;
If L [W [j - 1]] < 0 then
```

```
Dec (F [W [j - 1], W [j]], dt) ;
End ;
End ;
Until L [d] = MaxInt ;
End ;
Procedure Xuat ;
Begin
Clrscr ;
Writeln (’Tong luong hang là’, tong) ;
For i := 1 to n do
Begin
For j := 1 to n do
Write (F [i, j], ’ ’) ;
Writeln ;
End ;
Readln ;
End ;
BEGIN
Nhap ;
Xu ly ;
Xuat ;
END.
```

Bài 7 : Có n trường học lộp đội tụyển thi chay việt dã. Mỗi đội gồm k người. Mỗi lẫn thi đấu mỗi trường cử một người ra chạy. Như vậy có k lần thi tất cả. Mỗi lần thi như vậy người chạy nhanh nhất được nhận huy chương. Trường nào nhận được nhiều huy chương nhất thì thắng cuộc. Trường phổ thông “Siêu đẳng” không khá về thể thao lắm nên tìm cách thi có lợi nhất. Do Ban tổ chức sẽ lần lượt gọi danh sách theo thứ tự đăng kí mà các trường đã gởi lên. Vì vậy trường đã thu thập thành tích của tất cả vận động viên của các đội bạn (tính theo thời gian chạy 1000m). Đợi cho tất cả các trường khác đăng kí xong họ tìm cách lấy danh sách đó để sắp xếp số thứ tự của đội mình (dĩ nhiên họ nắm rõ trình độ đội nhà).

Cho biết nếu nhiều vận động viên về đích cùng lúc thì đều giành được huy chương.

Bạn hãy giúp trường “Siêu đẳng” sắp xếp danh sách đội tuyển sao cho số huy chương đoạt được là nhiều nhất.

Dữ liệu : cho trong file INP.BL7 gồm n 4* 1 dòng

+ Dòng 1 : chứa 2 số nguyền dicơng n,k(k < 20)

+ Dòng i+1(1 < i < n) gồm k số ứng với thành tích của k vận động viên của trường thứ i

Quy ước trường thứ n là trường "Siêu đẳng"

Kết quả : xuất ra file OUT.BL7 gồm 2 dòng

+ Dòng 1 : ghi tổng số huy chương mà trường nhận được.

+ Dòng 2 : chứa số thứ tự của vận động viên (sổ thứ tự căn cứ theo thứ tự của vận động viên trong dữ liệu nhập vào).

HƯỚNG DẪN GIẢI THUẬT:

— Gọi mảng X1 là mảng chứa thành tích cao nhất của N -1 trường ứng với mỗi đợt thi.

— Gọi mảng X2 là mảng chứa thành tích của trường N

— Xây dựng mảng A [1.. 2 * K + 2, 1.. 2 * K + 2] :

+ Đỉnh 2 * K + 1 là đỉnh phát, (đỉnh s)

+ Đỉnh 2 * K + 2 là đỉnh thu. (đỉnh d)

+ A [s, i] := 1, i = 1 đến k

+ A [i + k, d] := 1, i = 1 đến k

+ A [i, k + j] := 1 nếu X2[i] =< X1[j]

— Tìm giá trị luồng cực đại theo thuật toán FORD-FULKERSON ở bài 6.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_7 ;
Uses CRT ;
Const MaxK = 20 ;
Max = 2 * MaxK + 2 ;
Var X1, X2 : Array [1.. MaxK] of Byte ;
m, n, k : Integer ;
i, j, X : Byte ;
A, F : Array [1.. Max, 1.. Max] of Byte ;
L : Array [1.. Max] of Integer ;
w : Array [1.. Max] of Byte ;
s, d : Integer ;
xong : Boolean ;
tong, dt : Integer ;
Procedure Nhap ;
Var  f : Text ;
Begin
```

```
Assign (f, 'INP.BLT) ;
Reset (f) ;
Readln (f, n, k) ;
FiUChar (X1, SizeOf (XI), $FF) ;
X2 := X1 ;
For i := 1 to n - 1 do
Begin
For j := 1 to k do
Begin
Read (f, x) ;
If X < X1[j] then
X1[j] := X ;
End ;
Readln (0 ;
End ;
For j := 1 to k do
Read (f, X2 [j]) ;
Close (f) ;
End ;
Procedure TaoGraph ;
Begin
FillChar (A, SizeOf (A), 0) ;
F := A ;
s := 2 * k + 1 ;
d := 2 * k + 2 ;
m := 2 * k + 2 ;
For i := 1 to k do
A [s, i] := 1 ;
For i := 1 to k do
A [i + k, d] := 1 ;
For i := 1 to k do
For j := 1 to k do
If X2 [i] <= X1 [j] then
A [i, k + j] := 1 ;
End ;
Function Min (x1, x2 : Integer) : Integer ;
```

```
Begin
If x1 < x2 then Min := x1
Else Min := x2 ;
End ;
Procedure Thuchien ;
Begin
Tong := 0 ;
Repeat
For i := 1 to m do L [i] := MaxInt ;
L [s] := 0 ;
Repeat
Xong := True ;
For i := 1 to m do
If L [i] < > MaxInt then
Begin
For j := 1 to m do
If L [j] = MaxInt then
Begin
If (A [i,j] > 0) and (F [i,j] < A [i,j]) then
Begin
L [j] := +i ;
Xong := False ;
End ;
If (A [j, i] > 0) and (F [j, i] > 0) then
Begin
L [j] := -i ;
Xong := False ;
End ;
If L [d] < > MaxInt then
Begin
j := m ;
i := m ;
Xong := True ;
End ;
End ;
If L [d] < > MaxInt then
```

```
Begin
i := m ;
Xong := True ;
End ;
End ;
Until xong ;
If L [d] < > MaxInt then
Begin
dt := MaxInt ;
FillChar (W, SizeOf (W), ) ;
i := 1 ;
W [1] := d ;
Repeat
Inc (i) ;
W [i] := Abs (L [W [i - 1]]) ;
If L [W [i - 1]] >= 0 then
dt := Min (dt,A [W[i],W [i-1]]-F[W[i],W[i-1]]);
If L [W [i - 1]] < 0 then
dt := Min (dt, F [W [i - 1], w [i]]) ;
Until w [i] = s ;
Inc (tong, dt) ;
For j := 2 to i do
Begin
If L [W [j - 1]] >= 0 then
Inc (F [W [j], W [j - 1]], dt) ;
If L [W [j - 1]] < 0 then
Dec (F [W [j - 1], W [j]], dt) ;
End ;
End ;
Until L [d] = Max Int ;
End ;
Procedure Xuat ;
Var  g : Text ;
B : Array [1.. Max K] of Byte ;
Begin
Assign (b,'OUT.BL7') ;
```

```
Rewrite (g) ;
Writeln (g, tong) ;
FillChar (B, SizeOf (B), 0) ;
For i := 1 to k do For j := 1 to k do
If F [i, j + k] > 0 then
Begin
B [j] := i ;
j := k ;
End ;
For i := 1 to k do
If F [s, i] = 0 then
For j := 1 to k do
If B [j] = 0 then
Begin
j := k ;
End ;
For i := 1 to k do
Write (g, B [i],' ') ;
Close (g) ;
End ;
BEGIN
Nhap;
TaoGraph;
Thuchien;
Xuat;
END.
```

Bài 8. Cho n số nguyên dương. Hay xác định tập lớn nhất các số đã cho sao cho không có hai phần tử nào thuộc tập này mà phần tử này là ước số của phần tử kia

Dữ liệu : cho trong file INP.BL8 gồm 2 dòng.

+ Dòng 1 : Chứa n

+ Dòng 2 : Chứa n số nguyên dương.

Các số trên cùng một hàng ghi cách nhau ít nhất một dấu cách

Kết quả : Xuất ra màn hình thông báo số phần tử và các số được chọn.

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT:

1. Mảng So[1..N,1..2] of Set of Byte

Mảng Sohang [1..1000] of Set of Byte

2. So[i,1] := {i}

So[i,2] := {j |X[i] mod X[j] = 0}

3. Sohang [1] := So[1,1]

So hang [2] := So[1,2]

LSohang := 2;

For k := 2 to N do

Gán Tam bằng Sohang

For i := to LSohang do

For j := 1 to 2 do

Xét A = Tam[i]+so[k,j]

Nếu tồn tại tập A0 là một phần tử của Sohang và A0 là tập con của A thì không xét đến A nữa. (1)

Nếu điều kiện (1) thỏa thì kiểm tra xem có phần tử A1 nào của Sohang mà A là tập con của A1 hay không. Nếu có thì xóa phần tử A1 khỏi Sohang.

Kết nạp A vào Sohang.

4. Tìm phẩn tử A trong Sohang sao cho A có số phần tử ít nhất. Xuất ra kết quả là tất cả các phần tử không thuộc A.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_8;
UsesCRT;
Const MaxN= 100;
Maxsohang=500;
Typet tich=Set Of Byte;
tong=Array[1. .Maxsohang] of Tich;
Var  so : Array[1..MaxN,1..2] of Tich;
sohang, tam : tong;
n,i,j,m : Integer;
X : Array[1..MaxN] of Integer;
lsohang : Word;
dem : Word;
tichso : tich;
t,k : Integer;
Procedure Nhap;
Var f : Text;
```

```
Begin
Assign (f,'INP.BL8');
Reset (f);
FillChar(X,SizeOf(X),0);
Readln(f,n);
For i := 1 to n do
Read(f,X[i]);
Close(f);
End;
Procedure TaoGraph;
Begin
m := 0;
FillChar(so,SizeOf(so),0);
For i := 1 to n do
Begin
Inc(m);
so[m,1] := [i];
For j : = 1 to n do
If (i < > j) and (X[i] mod X[j] = 0) then
so[m,2] := so[m,2]+[j];
If so[m,2] = [ ] then Dec(m);
End;
End;
Procedure Xuly;
Begin
sohang[1] := so[1,1];
sohang[2] := so[1,2];
sohang := 2;
For k := 2 to m do
Begin
Move (sohang,tam,32*Word(lsohang));
dem := 0;
For i := 1 to lsohang do
For j := 1 to 2 do
Begin
tichso := tam[i]+so[kj];
```

```
t := 1;
While (t <= dem) and
(not (sohangit] <= tichso)) do Inc(t);
If t > dem then
Begin
t := 1;
While t <= dem to
If sohang[t] > = tichso then
Begin
Move(sohang[t+1],sohang[t], 32*Word(dem-t));
Dec(dem);
End;
ElseInc(t);
Inc(dem);
sohang[dem] := tichso;
End;
End;
Lsohang := dem;
End;
End;
Procedure Xuat;
Begin
Clrscr;
m := MaxInt;
t := 0;
For i := 1 to lsohang do
Begin k := 0;
For j := 1 to n do
If j in sohang[i] then Inc(k);
If m > k then
Begin
m := k;
t := i;
End;
End;
End;
Writeln('So phan tu', n-m);
```

```
For i := 1 to n do
If not (i in sohang[t]) then
Write (X[i], ' ');
Readln;
End;
BEGIN
Nhap;
TaoGraph;
Xuly;
Xuat;
END.
```

Bài 9 : Cho n số nguyên dương. Hãy xác định tốp nhỏ nhất chứa các số đã cho sao cho với mọi số A trong số n số nguyên dương đã cho, muốn tìm được ít nhất một số B thuộc tập hợp đã chọn và là chia hết cho A

Dữ liệu : trong file INP.BL9 gồm 2 dòng

+ Dòng 1 : Chứa n (n =< 10)

+ Dòng 2 : Chứa n số nguyên dương.

Các số trên cùng một hàng ghi cách nhau ít nhất một dấu cách.

Kết quả : Xuất ra màn hình thống báo số phần tử thuộc tốp hợp các số được chọn.

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT:

— Tạo ma trận A[i,j] với A[i,j] = 1 nếu X[i] chia hết cho X[j]

— Tìm tất cả các tập con của n phần tử này và kiểm tra với điều kiện đề bài.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_5_9;
Uses CRT;
Const MaxN= 10
Var n,i,j : Byte;
X : Array[1..MaxN] of Integer;
A : Array[1..MaxN,1..MaxN+l] of Byte;
B,Luu : Array[1..MaxN] of Byte;
min : Integer;
Procedure Nhap;
Var f : Text;
Begin
```

```
Assign(f,'INP.BL9');
Reset (f);
FillChar(X,SizeOf(X),0);
Readln(f,n);
For i := 1 to n do
Read (f,X[i]);
Closed(f);
End;
Procedure TaoGraph;
Begin
FillChar(A,SizeOf(A),0);
For i := 1 to n do
For j := 1 to n do
If X[i] mod X[j] = 0 then
Begin
Inc(A[i,n+1]);
A[i,j] := 1;
End;
FillChar(B,SizeOf(B),0);
Luu :=B;
min := MaxInt;
End;
Procedure Kiemtra (dem:Byte);
Var KT:Boolean;
Begin
For i := 1 to n do
Begin
KT := False;
For j := 1 to n do
If (B[j] = 1) and (A[j,i] = 1) then KT := True;
If not KT then Exit;
End;
If dem < min then
Begin
min := deni;
luu := b;
```

```
End;
End;
Procedure Tim(VT,dem ; Byte);
Var  i : Byte;
Begin
If dem = min then Exit;
If VT=n+1 then Kiemtra (dem)
Else
For i := 1 down to 0 do
Begin
B[VT] := i;
Tim (VT+ 1,dem+i);
B[VT] := 0;
End;
End;
Procedure Xuat;
Begin
Clrscr;
Witeln (‘So phan tu : ‘, min);
For i := 1 to n do
If luu[i] = 1 then Write (X[i],’ ’);
Readln;
End;
BEGIN
Nhap;
TaoGraph;
Tim(1,0);
Xuat;
END
```

Bài 10 Một công ti cần thay toàn bộ hệ thống dây điện cho n phòng làm việc, cho biết sơ đồ mạng lưới điện hiện có của n căn phòng này được biểu diễn bằng ma trận A[i,j] trong đó A[i,j] chính là độ dài của dây điện nối liền giữa hai phòng i, j (A[i,j] = A[j,i], A[i,j] = 0 nếu không có dây nối giữa phòng i và phòng j). Hãy lập trình tính độ dài của dây dẫn cần sử sựng sao cho cả n phòng đều có điện và số lượng nào là ít nhất.

Dữ liệu: cho tron file INP. BL0 gồm N+1 dòng

+ Dòng đầu ghi số N

+ Dòng i+1 (1 =< i =< N) ghi N số A[i,1], A[i,2]… A [i,N]

Các số ghi trên cùng một dòng các nhau ít nhất một dấu cách.

Kết quả: Xuất ra màn hình.

Act qua Xuất ra màn hình.

HƯỚNG DẪN GIẢI THUẬT

Gọi E1, E2, … , EM là các cạnh của đồ thị

Sắp xếp các cạnh theo thứ tự tăng dần của trọng số cạnh

For i := 1 to M do

Nếu việc chọn cạnh Ei không tạo nên chu trình thì chọn cạnh Ei

CHƯƠNG TRÌNH MẪU

```
PHOGRAM CHUONG_TRINH_MAU_BAI_5_10;
Type Canh=Record
Dau : Byte;
Cuoi : Byte;
Trong_so : Real;
End;
Var  Duong,Luu : Array[1..100] of Canh;
So_dinh, So_canh, dem : Byte;
Dinh_da_di : Set of Byte;
cay : Boolean;
M,Matran : Array[1..10,1..10] of Real;
Procedure Khoitao;
Begin
dem := 0;
FillChar(Luu,Sizeof(Luu),0);
FillChart(Duong,Sizeof(Duong),0);
FillChar(Matran,Sizeof(Matran),0);
FillChar(M,Sizeof(M),0);
Dinh_da_di := [ ];
End;
Procedure Nhap;
Var  f : Text;
i,j : Byte;
Begin
Assign(f, 'INP BL0');
```

```
Reset(f);
Readln(f,So_dinh);
For i := 1 to So_dinh do
Begin
For j:= 1 to So_dinh do
Read(f,m[i,j])
Readln(f);
End;
Close(f);
So_canh := 0;
For i:= 1 to So_dinh-1 do
For j := i to So_dinh do
If m[i,j] > 0 then
Begin
Inc(So_canh);
Duong[So_canh].dau := i;
Duong[So_canh].cuoi := j;
Duong[So_Canh].Trong_so := m[i,j].
End;
End;
Procedure Sapxep;
Var  i,j : Byte;
X : Canh;
Begin
For i := 1 to So canh-1 do
For j := So_canh down to i+1 do
If Duong[j].Trong_so <= Duong[j-1].Trong_so then
Begin
X := Duong[j];
Duong[j] := Duong[j-1];
Duong[j-1] := X;
End;
End;
Procedure Kruskal;
Var  i : Byte;
Function ChuTrinh(dinh_dau, dinh_cuoi : Byte) : Boolean;
```

```
Var  TH : Set Of Byte;
Thoat : Boolean;
i,j : Byte;
Begin
TH:= [dinh_dau];
Repeat
thoat := True;
For i := 1 to So_dinh do
If (i in TH) then
For i := 1 to So dinh do
If (not (j in TH) and (Matran[i,j] > 0) then
Begin
TH := TH+[j];
thoat := False;
End;
Until (dinh_cuoi in TH) or thoat;
If dinh_cuoi in TH then ChuTrinh := True
Else ChuTrinh := False;
End;
Begin
i := 1;
Repent
If (not( (Duong[i].dau in Dinh_da_di) and (Duong[i].cuoi in Dinh_da_di) ) ) or ( (Duong[i].dau in Dinh_da_di) and (Duong[i].cuoi in Dinh_da_di) and (ChuTrinh(Duong[i].dau, Duong[i].cuoi)=False) ) then
Begin
Dinh_da_di := Dinh_da_di + [Duong[i].dau.Duong[i].cuoi];
Inc(dem);
Luu[dem] := Duong[i];
Matran[Duong[i].dau,Duong[i].cuoi] := 1;
Matran[Duong[i].cuoi,Duong[i].dau] := 1;
End;
Inc(i);
Until (i= So_canh+1) or (dem=So_dinh-1);
If dem < So_dinh-1 then cay := False
Else cay := True;
```

```
End;
Procedure Xuat;
Var  i : Byte;
Begin
If cay then
For i := 1 to dem do
Writeln(Luu[i].dau, '-->', Luu[i].cuoi,' = ',
Luu[i].Trong_so:2:2)
Else
Write ('Khong the thuc hien');
Readln;
End;
BEGIN
Khoitao;
Nhap;
Sapxep;
Kruskal;
Xuat;
END.
```

§ 6. TRÒ CHƠI

1. Khái niệin

Một trong các dạng toán quan trọng trong tin học là bài toán trò chơi. Thông thường nhất là loại trò chơi Đối kháng - Hữu hạn - Thông tin đầy đủ với các tính chất sau đây:

— Có hai người chơi, lần lượt mỗi người đi một nước (Đối kháng).

— Hai đấu thủ được nắm đầy đủ tất cả các thông tin về tình hình trận đấu (Thông tin đầy đủ).

— Trò chơi phải kết thúc sau một số hữu hạn bước bằng việc chiến thắng của một đối thủ, tức việc thua của đấu thủ còn lại hoặc hòa với cả hai đấu thủ (Hữu hạn).

2. Thành phần cơ bản của trò chơi

Mỗi trò chơi thuộc loại đã nêu ở trên bao gồm 4 thành phần cơ Itttn như sau:

— Trạng thái tất cả các thông tin về tình trạng để tiếp tục trò chơi (Ví dụ trong cờ vua, các thông tin này là đến lượt ai được quyền đi, ván cờ còn lại gồm những quân nào v.v).

— Quy tắc đi : Từ một trạng thái có thể đi những bước nào, tức có thể đưa về trạng thái nào.

— Trọng thái ban đãu : Là trạng thái lúc bắt đầu chơi.

— Các trạng thái kết thúc : Là các trạng thái để kết luận thắng, thua (Điều kiện kết thúc trò chơi). Các trạng thái kết thúc được chia làm 3 loại : thắng, thua, hòa.

3. Graph và trò chơi

Thông thường mỗi trò chơi được biểu diến dưới dạng một Graph, trong đó:

— Mỗi đỉnh biểu diễn một trạng thái.

— Mỗi cạnh nối từ đỉnh A đến đỉnh B biểu diễn qui tắc đi cho biết từ trạng thái A có thể đi đến trạng thái B.

— Đỉnh bắt đầu biểu diến trạng thái ban đầu.

— Đỉnh kết thúc biểu diến trạng thái kết thúc.

4. Chiến thuật

Khi trạng thái ban đầu được thiết lập là trò chơi bắt đầu. Ta định nghĩa một chiến thuật chơi là một quy tắc cần chơi như thế nào, hay nói một cách khác, chiến thuật chơi là quy tắc chuyển từ một trạng thái (không phải trạng thái kết thúc) sang một trạng thái xác định theo quy tắc đi. Một chiến thuật chơi được gọi là chiến thuật thắng nếu đấu thủ chơi theo chiến thuật này sẽ luôn luôn thắng không phụ thuộc vào lối chơi của đối phương.

Ví dụ 1 : (Trò chơi xếp chai bia)

Hai người chơi lần lượt đặt từng chai bia đứng lên bàn, mỗi người đến lượt minh, được đặt một chai. Ai không còn chỗ đặt nữa là thua.

HƯỚNO DẪN GIẢI THUẬT

Trong trò chơi này, có chiến thuật thắng cho người đi trước như sau:

— Người đi trước đặt chai bia đầu tiên giữa bàn.

— Kế tiếp cho dù người thứ hai đặt chai bia ở đâu, người thứ nhất chỉ cần đặt chai bia đối xứng với chai người kia vừa đặt qua chai giữa bàn.

— Do mặt bàn có tính dối xứng qua tâm nên người thứ nhất luôn có chỗ đạt sau khi người thứ hai đặt.

— Do diện tích mặt bàn hữu hạn nên sau một số hữu hạn bước người thứ hai không còn chỗ đặt nữa và sẽ thua nếu người thứ nhất cứ chơi theo chiến thuật thắng nêu trên.

Ví dụ 2 : (Trò chơi bốc kẹo)

Có N = 100 cái kẹo. Mỗi người đến lượt mình đi có thể bốc 1, 2, 3 hoặc 4 cái. Ai không còn kẹo để bốc là thua. Tìm chiến thuật chơi.

HƯỚNG DẪN GIẢI THUẬT

Trong trò chơi này có chiến thuật thắng cho người đi sau:

— Luôn luôn bốc để số kẹo còn lại chia hết cho 5.

— Nếu đối phương bốc K cái thi mình bốc 5 — K cái.

— Qua 20 lượt đi sẽ bốc số kẹo cuối cùng và chiến thắng.

CHƯƠNG TRÌNH MẪU (Số kẹo ban đầu tùy ý)

```
Program    BOCKEO;
Var  Trangthai : integer;
       MayDi : Boolean;
Procedure MayDi1 (Var Trangthai : integer);
Var  Sokeoboc : integer,
Begin
If trangthai Mod 5 <> 0 then
Sokeoboc := Trangthai Mod 5;
Else {boc bua}
Sokeoboc := Rand (Min (4, Trangthai) ) + 1; Trangthai := Trangthai — Sokeoboc;
Writeln (Trangthai);
End;
Procedure NguoiDi1 (Var Trangthai : integer);
Var  Sokeoboc : integer;
Begin
Repeat
Readln (Sokeoboc);
Until (Sokeoboc > 0) and
(Sokeoboc <= Min (4, Trangthai) )
Trangthai := Trangthai — Sokeoboc;
Writeln (Trangthai);
End;
Function Ketthuc (Trangthai : integer) : Boolean;
Begin
If (Trangthai = 0) then
Ketthuc := True;
Else
Ketthuc := False;
End;
```

```
Begin
Randomize;
Trangthai := Random(MaxInt);
MayDi := (Random (1) = 0);
Repeat
If MayDi then
Begin
MayDi1 (Trangthai);
MayDi := False;
End;
Else
Begin
NguoiDi1 (Trangthai);
MayDi := True;
End;
Until Ketthuc (Trangthai);
If MayDi then
Writeln (’Nguoi thang’);
Else Writeln (’May thang’);
End.
```

5. Các bài toán trò chơi cơ bản

Bài 1 : Hai người chơi với nhau trò chơi như sau : với một số a đang có sẵn, đến lượt mình chơi, người đó sẽ viết số a + 1 hay 2a với điều kiện số mới viết này không vượt quá số nguyên dương N cho trước. Với số bắt đầu là 1, cả viết được số N trước thì xem như tháng.

Hãy viết chương trình mô tả trò chơi sao cho khả nàng thắng của máy cao. Xem như máy là người đi sau.

Ví Dụ:

N = 10 => Người đi trước có chiến thuật thắng

N = 20 Người đi sau có chiến thuật thắng

HƯỚNG DẪN GIẢI THUẬT

- Một vị trí gọi là vị trí thua nếu mọi nước đi từ nó đều dẫn đến một vị trí thắng.

- Một vị trí gọi là vị trí thắng nếu tồn tại nước đi từ nó đến một vị trí thua.

_ Mảng A [1..N] ghi nhận vị trí thắng, thua. (A [i] = 0 nếu i là vị trí thua và ngược lại).

- A [N] = 0.

- Xây dựng mảng A ở mọi vị trí:

Nếu A [i] = 0 thì

Đánh dấu A [i - 1] = 1

Nếu i chẵn thì đánh dấu A [i/2] = 1

Nếu A [i] chưa đánh dấu và mọi vị trí xuất phát từ i đều có giá trị bằng 1 thì A [i] := 0.

- Chiến thuật:

+ Nếu đang ở vị trí đánh số 1 thì tìm cách đến số được đánh số 0.

+ Nếu đang ở vị trí đánh số 0 thì tìm cách đến số nào cũng được.

CHƯƠNG TRÌNH MÁU

```
PROGRAM CHUÔNG TRINH_MAU_BAI_6_1;
Uses CRT;
Const MaxN = 100;
Var  N, i, j dem : Byte;
A : Array [1.. MaxN] Of Byte;
xong : Boolean;
BEGIN
Clrscr;
Write (’Nhap N = ’);
Readln (N);
FillChar (A, SizeOf (A), $FF);
A [N] := 0;
Repeat
xong := True;
For i := 1 to N do
Begin
If A [i] = 0 then
Begin
If i > 1 then
If A [i - 1] = 255 then
Begin
A [i - 1] := 1; xong := False;
End;
If not (odd (i) then
If A [i div 2] = 255 then
```

```
Begin
A [i div 2] := 1;
xong := False;
End;
End;
If A [i] = 255 then
Begin
If 2 * i <= N then j := 2
Else j := 1;
If 2 * i <= N then
If A [2 * i] = 1 then Dec (j);
If A [i + 1] = 1 then Dec (j);
If j = 0 then
Begin
A [i] := 0;
xong := False;
End;
End;
End;
Until xong;
dem := 1; i := 1;
Repeat
Writeln (’So hien gio:’ i);
If odd (dem) then
Begin
Write (’Ban hay nhap so:’);
Readln (i);
End
Else
Begin
If A [l] = 1 then
Begin
If i * 2 < N then
If A [i * 2] = 0 then j := i * 2;
If A [i + 1] = 0 then j := i + 1;
End
```

```
Else
j := i + 1;
Writeln (‘So do may chon’, j);
i := j;
End;
Inc (dem);
Until i = N;
If odd (dem) then Writeln (‘Ban da thua’)
Else Writeln (‘Ban da thang’);
Readln;
END.
```

Bài 2 : Hai người chơi với nhau trò chơi như sau : với một cặp số (a, b) đang có sẵn, đến lượt mình chơi, người đó sẽ viết cặp số a, a + b) hay (a + b, b) với điều kiện các số viết ra này đều không vượt quá số nguyên dương N cho trước. Với cặp số bắt đầu là (1, 1) ai không viết được nữa xem như thua. Hãy viết chương trình mô tả trò chơi sao ho khả năng thắng của máy cao. Xem như máy là người đi sau.

VÍ DỤ:

N = 10 Người đi sau có chiến thuật thắng

N = 6 Người đi trước có chiến thuật thắng

HƯỚNG DẪN GIẢI THUẬT

- Gọi mảng A [1..N, 1..N] là mảng ghi nhận vị trí thắng thua.
- A [i, j] = 0 (i = 1.. N, j từ i + N - 1 đến N).
- Xây dựng mảng A ở mọi vị trí

+ Nếu A [i, j] = 0 thì

Nếu j > i thì A [i, j — i] := 1

Nếu i > j thì A [i — j, j] := 1

+ Nếu A [i, j] chưa có giá trị thì

Nếu A [i, i + j] = A [i + j, j] = 1 thì A [i, j] := 0

- Chiến thuật:

+ Nếu đang ở vị trí đánh số 1 thì tìm cách đến số được đánh số 0.

+ Nếu đang ở vị trí đánh số 0 thì tìm cách đến số nào cũng được.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_2;
Uses CRT;
Const MaxN = 100;
Var  N, i, j, k, dem : Byte;
```

```
A : Array [1.. MaxN, 1.. MaxN] of Byte;
xong : Boolean;
BEGIN
Clrscr,
Write (’Nhap N =’);
Readln (N);
FillChar (A, SizeOf (A), $FF);
For i := 1 to N do
For j := 1 + N - i to N do
A [i, j] := 0;
Repeat
xong := True;
For i := 1 to N do
For j := 1 to N do
Begin
If A [i, j] = 0 then
Begin
If j - i > 0 then
If A [i, j - i] = 255 then
Begin
A [i, j - i] := 1;
xong := False;
End;
If i — j > 0 then
If A [i - j, j] = 255 then
Begin
A [i - j, j] := 1;
xong := False;
End;
End;
If A [i, j] = 255 then
Begin
If i + j < N + 1 then
If (A [i, j + i] = 1 and (A [i + j, j] =1)
then
Begin
```

```
A [i, j] := 0; xong := False;
End;
End;
End;
Until xong;
dem := 1; i := 1; j := 1;
Repeat
Writeln (‘Cap so hien gio : (‘, i, ‘,’, j, ’) ’);
If odd (dem) then
Begin
Write (’Ban hay nhap cap so : ’);
Readln (i, j);
End
Else
Begin
If A [i, j] = 1 then
Begin
If A [i + j, j] = then i := i + j
Else j := i + j;
End
Else
If i + j <= N then
i := i + j
Else
Begin
Writeln (’Ban da thang’);
Readln;
Halt;
End;
Writeln (’Cap so do may chon (’, i,’,’, j,’)’);
End;
Inc (dem);
Until i + j > N;
If odd (dem) then Writeln (’Ban da thua’);
Else Writeln (’Ban da thang’);
Readln;
```

END.

Bài 3 : Ta tự tạo ra một hột xí ngầu như sau:

Lấy 1 khối lập phương, trên 6 mặt ghi 6 số khác nhau từ 1 đến 6.

Mỗi lần gieo hột xí ngầu ta chỉ thấy được 3 mặt có chung đỉnh.

Hãy lập trình xác định 3 cặp số trên 3 cặp mặt đối diện củi hột xí ngầu sau một số lần gieo.

Dữ liệu : cho trong file INP.BL3

+ Dòng đầu ghi số lần gieo.

+ Các dòng sau mỗi dòng ghi 3 số cách nhau ít nhất một dấu cho biết 3 mặt số thấy được sau mỗi lần gieo.

Kết quả : xuất ra màn hình như sau:

+ Xuất ra từng bộ 3 dòng, mỗi dòng ghi hai số, cách nhau ít một dấu cách, diễn tả 2 số nằm trên hai mặt đối diện của hột xí ngầu.

+ Nếu không tìm được thì ghi : KHỒNG TÌM ĐƯỢC.

HƯỚNG DAN GIẢI THUẬT

Tạo ra tất cả các hoán vị của các 9ố từ 1 đến 6 (720 trường hợp) và kiểm tra với điều kiện đề bài.

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_3;
UsesCRT;
Const n = 6;
Filename = 'INP.BL3';
Max = 255;
D : Array [1.. 8, 1.. 3] of Byte = ((1, 2, 3), (1, 2, 4), (1, 4, 5), (1, 3, 5), (6, 4, 2), (6, 2, 3), (6, 3, 5), (6, 4, 5));
Type DinhType = Array [ 1.. 3] of Byte;
Var  Sol : Integer;
Vao : Array [1.. Max] of DinhType;
VTThu : Array [0.. 6] of Byte;
KT : Array [0.. 6] of Boolean;
dem : Integer;
Procedure Nhap;
Var  f : Text;
i, j : Integer;
Begin
Clrscr;
```

```
Assign (f, Filename);
{$I-}
Reset (f);
{$I+}
If IOResult < > 0 then
Begin
Write ('File', Filename,' co loi');
Readln;
Halt;
End;
Readln (f, Sol);
If (Sol < 1) or (Sol > Max) then
Begin
Writeln ('Du lieu sai');
Readln;
Halt;
End;
FillChar (Vao, SizeOf (Vao), 0);
For i := 1 to Sol do
Begin
For j := 1 to 3 do
Read (f, Vao [i, j]);
Readln (f);
End;
Close (0;
End;
Procedure Xuat;
Var  i : Byte;
Begin
Inc (dem);
For i := 1 to 3 do
Writeln (VTThu [i] : 3, VTThu [7 - i] : 3);
End;
Procedure Test;
Var  TH, TH1 : Set Of Byte;
i, j : Integer;
```

```
OK : Boolean;
Begin
For i := 1 to 2 do
If VTThu [i] > VTThu [i + 1] then Exit;
For i := 1 to 3 do
If VTThu [i] > VTThu [7 - 1] then Exit;
For i := 1 to Sol do
Begin
OK := False;
TH := [Vap [i, 1], Vao [i, 2], Vao [i, 3]];
TH1 := [];
For j := 1 to 8 do
Begin
TH1 := [VTThu [D [j, 1]], VTThu [D 0, 2]] VTThu [D [j, 3]]];
If TH1 = TH then
Begin
OK := True;
j := 8;
End;
End;
If not OK then Exit;
End;
Xuat;
End;
Procedure Try (V : Byte);
Var  i : Byte;
Begin
If V = n + 1 then Test
Else
Begin
For i := 1 to 6 do
If not (KT [i]) then
Begin
KT [i] := True;
VTThu [V] := i;
Try (V + 1);
```

```
KT [i] := False;
VTThu [V] := 0;
End;
End;
End;
Procedure Xuly;
Begin
dem := 0;
FillChar (VTThu, SizeOf (VTThu), 0);
FillChar (KT, SizeOf (KT), 0);
Try (1);
If dem = 0 then
Writeln (’KHONG TIM DUOC’);
Readln;
End;
BEGIN
Nhap;
Xuly;
END.
```

Bài 4 : Hai người chơi với nhau trò chơi như sau : Người thứ nhất sẽ nghĩ ra một số nguyên dương trong khoảng từ 1 đến N (N được cho biết trước). Người thứ hai sẽ lần lượt đưa ra các số dự đoán. Với mỗi số dự đoán này người thứ hai sẽ nhận được câu trả lời cho biết số mình vừa nêu ra lớn hơn, nhỏ hơn hay bằng với số mà người thứ nhất đã nghĩ. Bạn hãy giúp người thứ hai chọn đúng số cần tìm với số lần đoán càng ít càng tốt.

Luật chơi giới hạn số lần đoán không quá $[\log_2 N]$.

VÍ DỤ:

N = 10

Số cần tìm là 4 Lần đoán thứ 1 : 5

1 - Số bạn nghĩ nhỏ hơn

2 - Số bạn nghĩ lớn hơn

3 - Chính xác

Trả lời : 1

N = 8

Số cần tìm là 7 Lần đoán thứ 1 : 4

1 - Số bạn nghĩ nhỏ hơn

2 - Số bạn nghĩ lớn hơn

3 - Chính xác

Trả lời : 2

Lần đoán thứ 2 : 6

1 – Số bạn nghĩ nhỏ hơn

2 - Số bạn nghĩ lớn hơn

3 - Chính xác

Trả lời : 2

Lần đoán thứ 3 : 7

1 - Số bạn nghĩ nhỏ hơn

2 – Số bạn nghĩ lớn hơn

3 - Chính xác

Trả lời : 3

Số bạn nghĩ là 7

Lần đoán thứ 2 : 3

1 - Số bạn nghĩ nhỏ hơn

2 – Số bạn nghĩ lớn hơn

3 - Chính xác

Trả lời : 2

Lần đoán thứ 3:4

1 - Số bạn nghĩ nhỏ hơn

2 - Số bạn nghĩ lớn hơn

3 - Chính xác

Trả lời : 3

Số bạn nghĩ là 4

HƯỚNG DẪN GIẢI THUẬT:

1. x := 1 ; y := N; a := 0

2. a := (y + x) div 2

3. Lần đoán thứ i : a

4. nếu a lớn hơn số cần tìm thì gán y := a

Nếu n nhỏ hơn 9ố cần tìm thì gán X := a

5. Nếu số lần đoán vượt quá $\log_2$ thì chấm dứt

Ngược lại thì trà lại bước 2

CHƯƠNG TRÌNH MẪU:

CHUONG_TRINH_MAU_BAI_6_4;

Uses CRT;

```
Const MaxN = 10000;
Var N : Integer;
x, y, a : Integer,
dem : Integer;
Traloi : Byte;
BEGIN
Clrscr;
PROGRAM
Uses
Const
Var
Write (’Xin cho biet gia tri N = ‘);
Readln (N);
x := 1 ; y := N ; a := 0 ; dem := 1;
Repeat
Write ('Lan doan thu dem, ' : ');
a := (y + x) div 2;
Writeln (a);
Writeln ('1 - So ban nghi nho hon');
Writeln (’2 - So ban nghi lon hon');
Writeln ('3 - chinh xac’);
Write ('Tra loi');
Readln(traloi);
Case Traloi of
1 : y := a;
2 : X := a;
End;
If dem mod 4 = 0 then Clrscr;
Inc(dem);
Until (deni > In (N)/ln (2) + 1) or (traloi = 3);
If traloi < > 3 then Writeln ('Ban da thang')
Else Writeln ('So ban nghi la ', a);
Readln;
END.
```

Bài 5 : Bọn nhộn được một túi chứa 3^N đồng tiền vàng. Bạn biết rằng trong túi này có chứa một đồng tiền giả nhẹ hơn tiền thật. Với một chiếc cân có hai đĩa cân, bạn hãy tìm ra đồng tiền giả với số lần cân ít nhất.

Giả sử mỗi đồng tiền đều được đánh số từ 1 đến 3^N.

Mỗi lần cân, bạn thông báo ra màn hình những đồng tiền nằm trên đĩa cân bèn phải, đĩa cân bên trái và sau đó bạn sẽ nhận được câu trả lời về tình trạng cân: "L" nếu cân lệch về bên trái, nếu lệch về bên phải, "B" nếu cân thăng bằng.

Giới hạn số lần cân không quá N lần.

Ví Dụ.

Ví dụ 1:

N = 3, đồng tiên giả :

Lần cân 1

Đĩa cân bên phải: 1 2 3 4 5 6 7 8 9

Đĩa cân bên trái : 10 11 12 13 14 15 16 17 18

L

Lần cân 2

Đĩa cân bên phải : 1 2 3

Đĩa cân bên trái : 4 5 6

R

Lần cân 3

Đĩa cân bên phải 4

Đĩa cân bên trái : 5

B

Đổng tiền giả : 6

Ví dụ 2:

N = 3, đồng tiền giả : 27

Lần cân 1

Đĩa cân bên phải: 1 2 3 4 5 6 7 8 9

Đĩa cân bèn trái: 10 11 12 13 14 15 16 17 18

B

Lần cân 2

Đĩa cân bên phải: 19 20 21

Đĩa cân bên trái: 22 23 24

B

Lần cân 3

Đĩa cân bên phải: 25

Đĩa cân bên trái: 26

B

Đồng tiền giả : 27

HƯỚNG DẢN GIẢI THUẬT:

Chia số tiền làm 3 nhóm bằng nhau. Mỗi lần cân 2 nhóm (giả sử là nhóm 1 và 2).

Nếu cân không thăng bằng thì xét trở lại với nhóm tiền nhẹ hơn

Nếu cân thăng bằng thì xét trở lại với nhóm tiền thứ 3.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_ 6_5;
Uses CRT;
Var  N, A, X, y, i, j, dem : LongInt;
T : Array [1.. 3, 1.. 2] of Byte;
Traloi : Char;
BEGIN
Clrscr;
Write (’Cho biet N = ’);
Readln (N);
A := Trunc (exp(N* In(3)));
X := 1 ; y := A;
dem := 0;
Repeat
Inc(dem);
Writeln (’Lan thu ‘, dem);
T[1,1] :=x;T[1,2] :=x+(y-x) div 3;
T [2,1] :=T[1,2]+1 ; T[2,2] :=x+2*(y-x) div 3;
T[3,1] :=T[2,2]+1; T[3,2] :=y,
Write (’Dia can ben phai :’);
For i := T[l, 1] to T[l, 2] do Write (i, ’ ’);
Writeln;
Write (’Dia can ben trai :’);
For i := T[2, 1] to T[2, 2] do Write (i, ’ ‘);
Writeln;
Readln(Traloi);
Traloi := Upcase (Traloi);
Case Traloi of
```

```
'L' : j := 1;
'R' : j := 2;
'B' : j := 3;
End;
x := T[j, 1];
y:= T[j, 2];
Until x = y;
Writeln ('Dong tien gia : ', x);
Readln;
END.
```

Bài 6 : Một vùng địa hình ABCD hình vuông được chia thành lưới ô vuông NxN. Trên một số ô của lưới có chướng ngại vật. Có tất cả M cần điều khiển vùng địa hình này (M=<15). Với mỗi căn điều khiển Ci(1=<i=<M) bạn có thể thay đổi tình trạng của một số ô thuộc lưới (biến ô có chướng ngại vật thành không có và ngược lại). Hai ố được gọi là liền kề nhau nếu chúng đều khống có chướng ngại vật và có chung nhau 1 cạnh. Bạn cần vượt qua vùng đất này trên các ô liền kề nhau từ một ô không có chướng ngại vật nào đó trên cạnh AB (cạnh chứa các ô (1,1), (1,2),...(1, N)) và ra khởi vùng đất này tại 1 ô nào đó trên cạnh CD.

Hãy xác định cần sử dụng những cần điêu khiển nào sao cho bạn có thể vượt qua vùng địa hình này với số cần điều khiển cần sử dụng là ít nhất.

Dữ liệu : cho file INP.BL6

+ Dòng đầu ghi hai số nguyên dương N và M

+ Dòng tiếp theo mỗi dòng ghi N số biểu diễn vùng địa hình với qui ước ô (i,j) = 1 nếu ô đó có chướng ngại vật và bằng 0 nếu không có.

+ N dòng tiếp theo mhi dòng ghi N số biểu diễn tác động của các cân điểu khiển lèn các ò của lưới với quy ước:

— Nếu cồn Cỵ tác động lèn ố (i,j) thỉ bit thứ k của giá trị tại ố /iờy sẽ bằng 1 và bàng 0 nếu ngược lọi.

Kết quả : Xuất ra màn hình:

+ Dòng 1 : Ghi số cần điều khiển cần sử dụng.

+ Dòng 2 : Ghi số thứ tự của các cần điêu khiển

VÍ DỤ:

HƯỚNG DẪN GIẢI THUẬT:

Thử tất cả các khả năng của các cần điều khiển

Trong trường hợp số cần điều khiển nhiều hơn hay tổng số lượng ít nhất trước đó (nếu có) thì không tìm tiếp nữa.

Trong mỗi trường hợp, xây dựng lại ma trận địa hình dưới tác động của các cần điều khiển và kiểm tra xem có tồn tại đường đi thỏa yêu cầu đề bài hay không.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_6;
Uses CRT;
Const MaxN= 100;
MaxM= 15;
Dt : Array[L.2,1 ..4] of ShortInt
=((-1,1,0,0),(0,0,-1,1) );
Var n,m,i,j : Byte;
A,D,E : Array[L.MaxN,l..MaxN] of Byte;
B : Array[l...MaxN,l..MaxN] of integer;
min : Integer;
L,C : Array[1..MaxN] of Byte;
Nhap;
f : Text;
Begin
Clrscr;
Assign(f,'INP.BL6');
Reset(f);
Readln(f,N,M);
FillChar(A,SizeOf(A),0);
For i := 1 to n do
Begin
For j := 1 to n do
Read(f,A[i,j]);
Readln(f);
End;
FillChar(B,SizeOf(B),0);
For i := 1 to n do Begin
For j := 1 to n do Read(f,B[i,j]);
Readln(f);
End;
Close(f);
min := MaxInt;
FillChar(C,SizeOf(C),0);
```

```
End;
Function KT(x : Integer) : Boolean;
Begin
If (x>0) and (x<n+1) then
KT :=True
Else
KT := False;
End;
Procedure KiemTra(dem : Byte);
Var  i,j,k : Integer;
xong : Boolean;
Begin
D := A;
For k :== 1 to M do
If C[k] = 1 then
For i := 1 to n do
For j := 1 to n do
If (B[i,j] shr (M-k)) and 1 = 1 then
D[i,j] := (D[i,j] + 1) mod 2;
FillChar(E,SizeOf(E),0);
For j := 1 to n do
If D[1 j]=0 then E[1 j] := 1;
Repeat
xong := True;
For i := 1 to N do
For j := 1 to N do
If E[i,j] = 1 then
For k := 1 to 4 do
If KT (i+Dt[1.k]) and KT(j+Dt[2,k]) then
If (E[i+Dt[1,k]j+Dt[2,k]]=0) and
(D[i+Dt[1,k]j+Dt[2,k]]=0) then
Begin
xong := False;
E[i+Dt[1,k]j+Dt[2,k]] := 1;
End;
Until (xong);
```

```
For j := 1 to n do
If E[n,j] = 1 then
Begin
min := dem;
L := C;
Exit;
End;
End;
Procedure Tim (VT, dem : Byte);
Var  i : Byte;
Begin
If dem = min then Exit;
If VT=M+1 then KiemTra(dem)
Else
Begin
For i := 0 to 1 do
Begin
C[VT] := i;
Tim(VT+1,dem+i);
C[VT] := 0;
End;
End;
End;
Procedure Xuat;
Begin
Writeln('So can dieu khien:' min);
For i := 1 to n do
Begin
For j := 1 ton do
If D[i,j] = 0 then Write (' y ')
Else Write('*');
Writeln;
End;
For i := 1 to M do
If L[i]=1 then Write(I, ' ');
BEGIN
```

```
Nhap;
Tim(1,0);
Xuat;
END.
```

Bài 7 : Giả sử X là tập hợp các điểm nguyên (p, q) của mặt phẳng Descartes trong đỏ p và q là các số nguyên dương hay bằng không và không vượt quá số N cho trước. Hai đấu thủ chơi bằng cách lần lượt chọn một điểm của tập hợp X; nếu (i,j) là điểm mới được chọn thì đấu thủ tiếp theo đến lượt mình phải chọn một điểm (i1,j1) hoặc ở trên đường vuông góc của (i,j) hạ xuống trục tung hay trục hoành. Người nào đánh dấu được điểm (0,0) thì tháng. Hãy viết chương trình mô tả trò chơi sao cho khả năng thắng của máy cao. Xem như máy là người di sau.

VÍ DỤ:

Ví dụ 1 : N = 10, (i,j) = (8,9)

Người đi trước có chiến thuật thắng

Ví dụ 2 : N = 20, (i,j) = (10,10)

Người đi sau có chiến thuật thắng

HƯỚNG DẪN GIẢI THUẬT:

— Mảng A[1..N,1..N] ghi nhận vị trí thắng, thua.

— Gán mảng A bằng 1

— Gán A[i,j] = 0 với i từ 0 đến N Chiến thuật:

+ Nếu đang ở vị trí đãnh số 1 thì tìm cách đến vị trí đánh số 0

+ Nếu đang ở vị trí đánh số 0 thì tìm cách đến vị trí nào cũng được.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_7;
Uses CRT;
Const MaxN = 100;
Var  N,i,j,k, dem,p,i1,j1: Byte;
A : Array[0,MaxN,0..MaxN] of Byte;
xong : Boolean;
BEGIN
Clrscr;
Write('Nhap N = ');
Readln(N);
FillChar(A,SizeOf(A),1);
For i := 0 to N do A[i,j] := 0;
dem := 1; i := N ; j := N;
```

```
Write('Nhap vi tri bat dau : '); Readln(i,j);
Repeat
Writeln ('Cap so hien gio : ('',i,'','',j,'')');
If odd(dem) then
Begin
Write('Ban hay nhap cap so : '');
Readln(i,j);
End
Else
Begin
If A[i,j] = 1 then
Begin
i1 := i; j1 := j;
For k := 0 to i-1 do
If A[k,j]=0 then i1 := k;
If i1=i then
For k := 0 to j-1 do
If A[i,k]=0 then j1 := k;
End
Else
i1 := i-1;
i := i1 ; j := j1;
Writeln('Cap so do may chon ('',i1,'','',j1,'')');
End;
Inc(dem);
Until i+j = 0;
If odd(dem) then Writeln('Ban da thua')
Else Writeln ('Ban da thang');
Readln;
END.
```

Bài 8 : Đạidương là một lưới N x N ô vuông. Trên đó bố trí một tàu có kích thước 1 X K

Một người bắn vào đại dương mỗi lần 1 ô cho đến khi bắn trúng tàu

Yêu Cầu:

- Hãy chỉ ra 1 phương án bắn ít đạn nhất để đảm bảo bắn trúng tàu

Sau mỗi lần bắn kết qua nhận được từ bàn phím

Xét bài toán trong 2 trường hợp sau:

Sau mỗi lần bắn kết quả nhận được từ bàn phím

Xét bài toán trong 2 trường hợp sau

+ M = 8 và K = 4

+ N = 10 và K = 3

HƯỚNG DẪN GIẢI THUẬT

- Xét mảng A[1..n,1..n]

- Gán A tòng 0

- Gán i=k

- Thực hiện cho đến khi i>2n

+ For j := 1 to i do

Nếu j =< n và i+1-j =< n thì gán A[j,i+1-j] := 1

+ Tăng i lên k đơn vị

- Lần lượt bắn vào các ô (i,j) mà A[i,j] = 1

CHƯƠNG THÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_8;
USES CRT,GRAPH;
VAR n,k,Cn:Byte;
A:Array[1.. 100,1.. 100] of Byte;
Procedure Input;
Begin
Clrscr;
Write('Nhap vao kich thuoc dai duong:');
Readln(n);
Write('Nhap vao chieu dai cua tau:');
Readln(k);
FillChar(A,SizeOf(A),0);
End;
Procedure O(x1,y1,x2,y2 : Ineger; color1,color2,color3,color4 : Byte);
Begin
SetFillStyle( 1,color 1);
Bar(x1,y1,x2,y2);
SetFillStyle(1,color2);
Bar(x1 + 1,y1 + 1,x2-1,y2-1);
SetFillStyle(1,color3);
Bar(x1 + 3,y1 + 3,x2-3,y2-3);
```

```
SetColor(color4);
Line(x2-1 ,y 1 +1 ,x2-1 ,y2-1);
Line(x1+1,y2-1,x2-1,y2-1);
Line(x2-2,y1+2,x2-2,y2-2);
Line(x1+2,y2-2,x2-2,y2-2);
End;
Procedure Drw(i,j:Byte,t:Byte);
Begin
Case t of
1 : 0((i-1)*30+10,(j-1)*30+10,1*30+ 10j*30+10,0,7,9,1);
2 : O((i-1)*30+ 10,(j-1)*30+10,i*30+10j*30+10,0,1,9,7);
3 : O((i-1)*30+ 10,(j-1)*30+ 10,i*30+ 10j*30+10,0,4,12,7);
End;
End;
Procedure Draw;
Var  i,j : Byte;
Begin
O(0,0,639,479,0,8,7,15);
For i := 1 to n do
For j := 1 to n do
Drw(i,j,1);
End;
Procedure Init;
Var  gd,gm : Integer;
Begin
gd := Detect;
Initgraph(gd,gm,'C:\BP\BGT');
End;
Procedure Solve;
Var  i,j: Byte;
Begin i := k;
Cnt := 0;
Repeat
For j := 1 to i do
If 0 <== n) find (i+1-j < =n) then
Begin
```

```
A(j,i+1-j] := 1;
Inc(Cnt);
End;
Inc(i,k);
Until i > 2*n;
End;
Procedure Play;
Var  Ans : Char;
i,j : Byte; s1,s2 : String;
Begin
Randomize;
Repeat
O(10,400,629,430,0,3,11,7);
O(10,440,629,470,0,3,11,7);
Repeat
i : = Random(n)+1;
j : = Random(n)+1;
Until (A[i,j]=1);
Dec(Cnt);
A[i,j] := 2;
Drw(I,j,2);
Str(i,s1);
Str(j,s2);
SetColor(4);
OytTextxy(30,410,'(`+s1','+s2+')');
OutTextxy(30,450,'Tau o day phai khong (Y/N)');
Ans := Upcase(ReadKey);
Drw(i,j,1);
Until (Ans=TT) or (Cnt=0);
Drw(i,j,3);
Readln;
CloseGraph;
End;
BEGIN
Input;
Init;
```

Draw;

Solve;

Play;

END.

Bài 9 : Cậu bé nghĩ một số (gọi là S) gồm bốn chữ số (không nhất thiết khác nhau) trong sáu chữ số từ 1 đến 6. Để tìm số đó, máy lần lượt đưa ra các số dự đoán (gọi là M), mỗi số gồm bốn chữ số (không nhất thiết khác nhau). Với mỗi số dự đoán, máy nhận được hai câu trả lời của cậu bé cho hai câu hỏi sau.

+ Có bao nhiêu chữ số trong M là chữ số trong s nhưng vi trí xuất hiện của mỗi chữ số đố là scd ĩ

+ Có bao nhiêu chữ số trong M là chữ số trong s và đồng thời vị trí xuất hiện của mỗi số đều đúng ?

Yêu cầu : Hãy hiện lên màn hình các số máy dự đoán và nói mỗi số đó nhận hai câu trả lời từ bàn phím của cậu bé cho đến khi được số đúng như cậu bé nghĩ. (Số lần dự đoán không quả sáu lần).

HƯỚNG DẢN GIẢI THUẬT:

— Thiết lập một mảng lưu trữ tất cả các số có khả năng xảy ra.

— Sau một số lần đoán ban đầu (có thể là một số ngẫu nhiên hay do bạn tự chọn), ta kiểm tra kết quả nhận được (về số lượng các chữ số xuất hiên đúng vị trí và sai vị trí). Ứng với mỗi số A mà máy đoán, ta so sánh số này với các số còn lại (giả sử đang xét số B), nếu số chữ số trong A và B giống nhau và có vị trí xuất hiện như nhau bằng với số chữ số trong A là chữ số trong s và vị trí xuất hiện của mỗi chữ số đó là đúng, đồng thời số chữ số trong A và B giống nhau nhưng vị trí xuất hiện khác nhau bằng với số chữ số trong A là chữ số trong s nhưng vị trí xuất hiện là sai thì ta xét tiếp số B này. Các số không thỏa xem như bị loại và đưa ra số dự đoán tiếp theo là một trong các số còn lại.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_9;
Uses CRT;
Const mn = 10000;
Type so = 0…7;
so1 = 1..6;
ma = Array[1..6] of so;
Var Doan : Array[1..7] of ma;{Luu cac lan doan}
Save : Array[1 .mn] of ma;
n : Integer;
```

```
Procedure DungSai(k:Byte);
Begin
Writeln;
Write('Dung so - Dung vi tri:');
Readln(Doan[k][5]);
Write('Dung so - Sai vi tri :')
Readln(Doan[k][6]);
If Doan[k][5]=4 then
Begin
Writeln;
Writeln('Chon dung so');
Readln;
Halt;
End;
Writeln;
End;
Procedure Chuan_bi(k:Byte);
Var  i:Byte;
Begin {Khoi dong cac so doan lan thu k <= 3}
Case k of
1:Begin
Doan[1][1] := 1;Doan[1][2] := 2;
Doan[1][3] := 3;Doan[1][4] := 4;
End;
2:Begin
Doan[2][1] := 2;Doan[2][2] := 1;
Doan[2][3] := 5;Doan[2][4] := 6;
End;
3. Begin
Doan[3][1] := 5;Doan[3][2] := 6;
Doan[3][3] := 4;Doan [3][4] := 3;
End;
End;
For i := 1 to 4 do Write(Doan[k] [i]:2);
DungSai(k);
End;
```

```
Function Check(t,k:Byte) : Boolean;
Var  t1, t2 : Array [1..4] of Boolean;
Dvt,Svt,i,j : Byte;
Begin
FillChar(T1,SizeOf(T1),False);
FillChar(T2,SizeOf(T2),False);
Dvt := 0;
Svt := 0;
For i := 1 to 4 do
If Save[t][i] = Doan[k][i] then
Begin
Inc(Dvt);
T2[i] := True;
T1[i] := True;
End;
If Dvt=Doan[k][5] then
For i := 1 to 4 do
If not T1[i] then
For j := 1 to 4 do
If not (T2[j] and (Save[t][i]=Doan[k][j])) then
Begin
Inc(Svt);
T2[j] := True; j :=4;
End;
Check := (Dvt=Doan[k][5]) and (Svt=Doan[k][6]);
End;
Procedure Loaidot1;
Var  i, i1, i2, i3, i4, i5, i6 Byte;
Begin
n := 1;
For i1 := 1 to 6 do
For i2 := 1 to 6 do
For i3 := 1 to 6 do
For i4 := 1 to 6 do
Begin
Save[n][1] := i1;
```

```
Save[n][2] := i2;
Save[n][3] := i3;
Save[n][4] := i4;
If Check(n,1) and Check(n,2) then Inc(n);
End;
Dec(n);
End;
Procedure Loai(k : Byte);
Var  i,i2 : Byte;
Begin i := 1;
While (i <= n) do
If not Check(i,k) then
Begin
For i2 := i to n-1 do
Save[i2] := Save[i2+1];
For i2 := 1 to 6 do
Save[n][i2] := 0;
Dec(n);
End
Else Inc(i);
End;
Procedure Duara(k : Byte);
Var  i : Byte;
Begin
i := Random(n) +1;
Doan [k] := Save[i];
For i := 1 to 4 do
Write(Doan[k][i]:2);
DungSai(k);
End;
Procedure Play;
Var deni : Byte;
Begin
Writeln ;
Write (’ ’) ;
Chuan_bi(1) ;
```

```
Chuan_bi(2) ;
Loaidot1;
dem :=3 ;
Repeat
Duara (dem) ;
Loai (dem) ;
Inc (dem) ;
Until dem > 1000 ;
End ;
BEGIN
Clrscr ;
FillChar (Save, SizeOf (Save), 0) ;
Randomize ;
Play ;
END.
```

Bài 10. Cho phương trình

?x^{2n}+ ? x^{2n-1}+…+ ?x +1 =0

Hai người lẫn lượt điền các số thực vào các dấu ?

Nếu phương trình có nghiệm thực thì người đi trước thắng

Hãy lập trình cho máy đề tim ra cách điển số cho người thứ nhất sao cho người này luôn luôn thắng.

VÍ DỤ

Ví dụ 1 : N = 2 Máy chọn vị trí 1

Hệ số : 0

Hãy chọn vị trí : 2

Nhập hê số : 3

Máy chọn vị trí: 4

Hệ số : -4

Hãy chọn vị trí : 3

Nhập hê số : -9

Phương trình trên luôn có nghiệm thực

Ví dụ 2 : N = 1 Máy chọn vị trí 1

Hệ số : 0

Hãy chọn vị trí : 2

Nhập hệ số : 3

Phương trình trên luôn có nghiệm thực

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_6_ 10;
UsesCRT ;
Const MaxN = 100;
Var  A : Array[0..2*MaxN] of Real;
P : Array[1..2] of Real;
i,N,VT,VT1,VT2 : Byte;
GT,c : Real;
Function Tinh (G : Integer) : Real ;
Var  Tmp: Real ;
Begin
Tmp := 0 ;
For i := 0 to 2 * N do
If A [i] < > le30 then
If Odd (i) then Tmp := Tmp + A [i] * G
Else Tmp := Tmp + A [i] * Abs (G) ;
Tinh := Tmp;
End ;
BEGIN
Clrscr ;
Write (‘Nhap va so N =’) ;
Readln (N) ;
For i := 1 to 2*N do A [i] := 1e30 ;
A [0] := 1 ;
For i := 1 to 2*n – 2 do
If Odd (i) then
Begin
Repeat
VT := Random (N) +1 ;
Until A [VT] = 1e30 ;
Writeln (‘May chon vi tri’, VT) ;
Writeln(‘He so : 0’) ;
A [VT] := 0 ;
End
Else
Begin
```

```
Writeln (‘Hay chon vi tri :’) ;
Readln (VT) ;
Writeln (‘Nhap he so :’) ;
Readlin (A [VT]) ;
End ;
For i := 1 to n do
Begin
If A [i * 2 – 1] = 1e30 then VT1 := -1 ;
If A [i * 2] = 1e30 then VT2 := i * 2 ;
End ;
P[1] := Tinh (1) ;
P[2] := Tinh(-1) ;
Writeln (‘May chon vi tri’, VT2) ;
A [VT2] := (P [1] + p [2])/2;
Writeln (’He so:’, A [VT2] : 0 : 4) ;
Write (’Hay chon vi tri:’) ;
Readln (VT) ;
Write (’Nhap he so:’) ;
Readln (A [VT]) ;
Writeln (’Phuong trinh tren luon co nghiem thuc’) ;
END.
```

§7. HÌNH HỌC

1. Khái niệm

Đa số các thuật toán đều tập trung vào văn bản và các con số, chúng được thiết kế và xử lý sẵn trong phần lớn các môi trường lập trình. Đối với các bài toán hình học thì tình huống khác hẳn, ngay cả các phép toán sơ cấp trên điểm và đoạn thẳng cũng có thể là một thách thức về tính toán.

Các bài toán hình học thì dễ hình dung một cách trực quan, nhưng chính điều đó lại có thể là một trở ngại. Nhiều bài toán có thể giải quyết ngay lập tức băng cách nhìn vào một mãnh giấy nhưng lại đòi hỏi những chương trình không đơn giản.

Ví dụ : Một điểm có nằm trong một đa giác hay không ?

2. Đối tượng hình hoc cơ bản

Trong các bài toán tin học thuộc loại hình học có 3 đối tượng cơ bản là : Điểm, đoạn thẳng và đa giác.

Điểm : Được xét bằng một cặp số nguyên là tọa độ của điểm đó trong hệ trục tọa độ Descartes thường dùng.

Đoạn thẳng : Là một cặp điểm được nối với nhau bởi một phần của đường thẳng

Đa giác : Là một dãy các điểm với hai điểm liên tiếp được nối bởi một đoạn thẳng, và điểm đầu nối với điểm cuối tạo thành một hình gấp khúc khép kín

3. Dữ liệu lưu trữ các đối tượng hình học cơ bản

Làm việc với các đối tượng hình học chúng ta cần quyết định thể hiện chúng như thế nào ? Thông thường ta dùng một mảng để biểu diên một đa giác, dù rằng trong một số trường hợp chúng ta có thể dùng danh sách liên kết hay các kiểu khác.

Trong háu hết các bài toán hình học chúng ta sẽ dùng kiểu lưu trữ đơn giản, dế hiểu như sau:

```
Type point as Record
X, y : Integer; end ;
Line = Record
pi, p2 : point; end ;
Var Polygon : Array [0.. Nmax] of Point ;
```

4. Các bài toán hình hoc cơ bản

Bài 1 : Cho N - giác lồi A1A2A3.. .AN-1AN với các đỉnh Ai(xi, yi) có tọa độ nguyên. Hãy tính diện tích đa giác trên.

Dữ liệu : Cho trong file INP.BL1 gồm 2 dòng

+ Dòng 1 : Chứa số nguyên dương N.

+ Dòng 2 : Chứa 2xN số nguyên dương x1y1x2y2 ...xNyN là tọa độ các đỉnh của đa giác. Mỗi số ghi cách nhau một dấu cách.

Kết quả : xuất ra màn hình diện tích đa giác

HƯỚNG DẪN GIẢI THUẬT:

— Gán S := 0

— For i := 1 to N do

$S := S+(x_{i+1}-x_i)\ (y_{i+1}+y_i)/2$

— Giá trị tuyệt đối của S chính là diện tích đa giác.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_1;
Uses CRT ;
Const MaxN = 100 ;
Type xy = Record x,y : Integer; End;
Var  Toado:Array[1..MaxN+1] of xy;
```

```
N,i : Integer ;
s : Real ;
Procedure Nhap;
Var  f : Text;
Begin
Assign (f,TNP.BL1') ;
Reset(f);
Readln(f,N);
For i := 1 to N do
Read (f,Toado[i].x,Toado[i].y) ;
Close (f) ;
Dientich ;
Procedure
Begin
S := 0 ;
Toado[N+1].x := Toado[1].x;
Toado[N+1].y := Toado[1].y;
For i := 1 to N do
S := S + (Toado[i+1].x - Toado[i].x)*(Toado[i+1].y + Toado[i].y)/2;
Writeln ('Dien tich da giac :', Abs (S) : 0 : 2) ;
Readln ;
End ;
BEGIN
Nhap ;
Dientich ;
END.
```

Bài 2 : Cho n đường thing AiBi(1 =< i =< n) phân biệt với Ai,Bi là các điểm cho trước. Hãy thông báo ra màn hình tất cả các cặp đường thẳng đòi mốt cắt nhau.

Dữ liệu : cho trong file INP.BL2 gồm N dòng (N khống biết trước). Dòng thứ i ghi 4 số thực xAi yAi xBi yBi . Các số trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

vi DỤ:

HƯỚNG DẪN GIẢI THUẬT:

Lập phương trình tất cả các đường thẳng (y1-y2)x + (x2-x1)y+(x1y2-x2y1) = 0

Ax + By + C = 0

- Với mỗi cặp đường thẳng, ta lập hệ phương trình

A1x+B1y+C1=0

A2x+B2y+C=0

- Nếu hệ này có D=A1B2-A2B1 khác 0 thì hai đường thẳng cắt nhau.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_2 ;
Uses CRT ;
Const MaxN = 10 ;
Type xy = Record x,y:Real; End ;
Dt = Record a, b, C : Real ; End ;
Var  Toado : Array [1..2,1..MaxN] of xy;
L : Array [1.. MaxN] of Dt;
N, i, j : Integer ;
Procedure Nhap;
Var f : Text ;
Begin
Clrscr ;
Assign (f,'INP.BL2') ;
Reset (f);
N := 0;
FillChar(Toado, SizeOf(Toado), 0);
Repeat
Inc (N) ;
Readln(f,Toado[1,N].x,Toado[1,N].y,Toado[2,N]x,Toado[2,N].y);
Until EOF (F) ;
Close (f):
End ;
Procedure Lap ;
Begin
Fill Char (L, Size Of (L), 0) ;
For i := 1 to N do
With L [i] do
Begin
a := Toado[1,i].y-Toado[2,i].y;
b := Toado[2,i].x-Toado[1,i].x;
c := Toado[1,i].x*Toado[2,i],y-Toado[1,i].y*Toado[2,i].x;
```

```
End;
End ;
Procedure Xuly ;
Var  D : Integer
Begin
D := 0;
For i := 1 to N-1 do
For j := i + 1 to N do
If L[i].a*L[j].b <> L[jJ].a*L[i].b then
Begin
Writeln (’Duong thang ‘,i,’ cat duong thang’ j);
Inc(D);
End ;
If D = 0 then
Writeln (’Khong co cap duong thang nao cat nhau’) ;
Readln ;
End ;
BEGIN
Nhap;
Lap;
Xuly;
END.
```

Bài 3 : Trên mặt phẳng cho N hình tròn. Hãy tính diện tích phần mặt phẳng bị phủ bởi các hình tròn trên

Dữ liệu:Cho trong file INP.BL3 đầu là số lượng hình tròn, từ dòng thứ 2 trở đi mỗi dòng chứa 3 số nguyên dương là tọa độ x, y cùa tâm và bán kính cùa từng hình trên (các số trên cùng một dòng ghi cách nhau ít nhất một dấu cách).

Kết quả: xuất ra màn hlnh.

Yêu cầu kỹ thuật: sai số cho phép không vượt quá 1 đơn

HƯỚNG DẪN GIẢI THUẬT:

— Tìm hình chữ nhật nhỏ nhất có các cạnh song song trục tọa độ và chứa toàn bộ N hình tròn.

— Chia hình chữ nhật này thành lưới các ô vuông có cạnh 0,1 đơn vị.

— Với mỗi ô thuộc hình chữ nhật, kiểm tra xem ô này có thuộc hình tròn nào hay không. Nếu có thì tăng diện tích cần tính lên 0.01 đơn vị

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_B AI_7_3;
Type Circle = Record x,y,r : LongInt; End;
Const MaxN = 20;
Inf = MaxInt;
Var  N,i,j,k,iljl : LongInt;
     C : Array[1..MaxN] of Circle ;
     S : Real;
     x1, y1, x2, y2 : LongInt;
Procedure Nhap ;
Var  f : Text;
Begin
Assign(f,'INP.BL3');
Reset(f) ;
Readln(f,n) ;
FillChar(C,SizeOf(C),0);
For i := 1 to N do
Readln(f,C[i].x, C[i].y, C[i].r);
Close(f);
End;
Function Kiemtra(x,y : Read) : Boolean;
Begin
Kiemtra := True;
For k := 1 to N do
If Sqr(x-C[k].x)+Sqr(y-C[k].y) <= Sqr(C [k].r) then Exit;
Kiemtra := False
End ;
Procedure Xuly ;
Begin
s := 0 ;
x1 := +Inf ; y1 := x1; x2 := -Inf ; y2 := x2 ;
For i := 1 to N do
With C [i] do
Begin
If x1 < x1+r then x1 := x-r ;
If y1 < x1+r then y1 := y-r ;
If x2 < x+r then x2 := x+r ;
```

```
If y2 < y+r then y2 := y+r ;
End ;
For i := x1 to x2 do
For i1 := 0 to 9 do
For j := y1 to y2 do
For j1 := 0 to 9 do
If Kiemtra(i+i1*0.1, j+j1*0.1) then S :=S+0.01;
Writeln('Dien tich : `,S:0:2);
Readln;
End;
BEGIN
Nhap ;
Xuly ;
END.
```

Bài 4 : Cho một đa giác khống tự cát cỏ N đỉnh A1,A2>.có tọa độ nguyên

Hãy đánh số lại tất cả các đỉnh của đa giác sao cho chiều đánh số này theo chiều kim đồng hồ.

Dữ liệu : cho trong file INP.BL4 gồm 2 dòng

+ Dòng 1 : Chứa số nguyên dương N

+ Dòng 2 : Chứa 2xN số nguyên dương x1 y1 x2 y2 ... xN yN là tọa độ các đỉnh của đa giác. Mỗi số ghi cách nhau một dấu cách.

Kết quả : Xuất ra màn hình tọa độ các đỉnh của đa giác theo thứ tự mới

HƯỚNG DẪN GIẢI THUẬT:

- Tính diện tích đa giác
- Nếu diện tích này dương thì đổi ngược chiều đánh số
- Nếu diện tích này âm thì giữ nguyên chiều đánh số

CHƯƠNG TRÌNH MẪU

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_4 ;
Uses CRT ;
Const MaxN = 100;
Type xy = Record x,y : Integer; End;
Var Toado : Array[ 1..MaxN+1] of xy;
N,i : Integer;
A : Real;
Procedure Nhap;
Var f : Text;
```

```
Begin
Assign (f,'INP.BL4');
Reset (f);
Readln (f,N);
For i := 1 to N do
Read (f, Toado [i].x,Toado [i].y) ;
Close (f);
End ;
Procedure Xuly ;
Begin
Clrscr;
A := 0;
For i := 1 to N do
A := A+(Toado[i+1].x-Toado[i].x)*(Toado[i+1].y+Toado[i].y)/2;
If A > 0 then For i :=N downto 1 do Writeln(Toado[i].x,' ', Toado[i].y)
Else For i := 1 to N do Writeln(Toado[i].x,' ',Toado[i].y);
Readln ;
End ;
BEGIN
Nhap;
Xuly;
END.
```

Bài 5 : Cho N điểm a1,a2,...,aN trên mặt phẳng. Các điểm đều có tọa độ nguyên và không có 3 điểm bất kỳ trong chúng thẳng hàng. Hãy viết chương trình thực hiện công việc sau đây : Xác định một đa giác không tự cắt có đỉnh là một số điểm trong các điểm đã cho và chứa tát cả các điểm còn lại va có chu vi nhỏ nhất. Hãy tính diện tích đa giác này.

Dữ liệu : cho trong file INP.BL5 gồm N + 1 dòng

+ Dòng 1 : chứa số N

+ Dòng i+1(1 =< i =< N) : chứa 2 số xi và yi là tọa độ ai.

Các số trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

Kết quả : xuất ra file OUT.BL5:

+ Dòng 1 ghi 3 số K, V, S với K là số đỉnh đa giác tìm được, V là chu vi và s là diện tích của nó.

+ Dòng i+1(1 =< i =< N) ghi tọa độ của đỉnh đa giác.

HƯỚNG DẪN GIẢI THUẬT:

— Tìm điểm có tung độ nhỏ nhất. Điểm đó sẽ là đỉnh đa giác.

- Giả sử ta đã chọn được điểm PM . Ta tìm điểm Pi sao cho góc hợp bởi đường thẳng PMPi và trục hoành là nhỏ nhất và đồng thời góc này phải lớn hơn góc hợp bởi $P_{M-1}P_M$ và trục hoành. Điểm Pi sẽ là một đỉnh của đa giác

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG TRINH MAU BAI_7_5 ;
UsesCRT ;
Const MaxN = 100;
Type Point = Record x,y : Interger; End ;
Var  n,i : Integer ;
P : Array[1..MaxN] of Point ;
L : Array[1..MaxN] of Byte ;
Procedure Nhap ;
Var  f : Text ;
Begin
Assign (f,'INP.BL5') ;
Reset (f) ;
Readln (f,N) ;
For i := 1 to N do
Readln (f,P[i].x,P[i].y);
Close (f);
FillCharCL,SizeOf(L))0);
End ;
Function Goc(P1,P2:Point):Real;
Var  dx,dy,ax,ay:Integer;
t:Real;
Begin
dx := p2.x-pljc; ax := Abs(dx);
dy := p2.y-pl.y; ay := Abs(dy);
If (dx = 0) and (dy = 0) then t := 0
Else t := dy/(ax+ay);
If dx < 0 then t := 2-t
Else If dy < 0 then t := 4+t;
Goc := 90*t;
End ;
Var  Min.M : Integer;
```

```
      Minangle.v : Real;
      t : Point;
Function   Thuchien : Integer;
Begin
Min := 1;
For i := 2 to N do
If P[i].y < P[Min].y then Min :=i;
M := 0; P[N+1] := P[Min]; Minangle := 0.0;
Repeat
M : = M+1; t := P[M] ; P[M] := P[Min] ; P[Min] := t;
Min := N+1; v := Minangle; Minangle := 360.0;
For i := M+1 to N+1 do
If Goc (P[M],P[i]) > v then
If Goc(P[M],P[i]) < Minangle then
Begin
Min := i;
Minangle : = Goc(P[M],P[Min]);
L[M] := i;
End;
Until Min = N+1;
Thuchien := M;
End;
Procedure Xuat;
Var  f : Text;
v,s : Real;
Begin
Assign (f,'OUT.BL5');
Rewrite (f) ;
Write (f, Thu chien,' ');
L[M+1] := L[1];
V := 0; P[L[N+1]].x := P[L[1]].x;
P[L[N+1]].y := P[L[1]].y;
For i := 1 to M do
V := V+Sqrt(Sqr(P[L[i]].x-P[L[i+1]].x)+Sqr(P[L[i]].y-P[L[i+1]].y));
S := 0;
For i := 1 to M do
```

```
S := S + (P[L[i+1]].x-P[L[i]].x)*(P[L[i+1].y+P[L[i]].y)/2;
S := Abs(S);
Writeln (f,V:0:2,' ',S:0:2);
For i := 1 to M do
Writeln(f,P[L[i]].x,' ',P[L[i]].y);
Close(f);
End;
BEGIN
Nhap ;
Xuat ;
END.
```

Bài 6 : Trong mặt phẳng tọa độ trực chuẩn cho N hình chữ nhật có các cạnh song song với trục tọa độ. Mỗi hình chữ nhật được xác định bởi tọa độ đỉnh dưới bên trái và đỉnh trên bên phải của nó. Hãy tính diện tích phần mặt phẳng bị phủ bởi các hình chữ nhật trên.

Dữ liệu : cho trong file INP.BL6 gồm N + 1 dòng

+ Dòng 1 : Chứa số N

+ Dòng i+1(1 =< i =< N) ; ghi 4 số nguyên x1,y1x2,y2 lần lượt là tọa độ đỉnh dưới bên trái và đỉnh trên bên phải của hình chữ nhật i

Các số ghi trên cũng một dòng cách nhau ít nhất một dấu cách.

Kết quả : Thông báo ra màn hình diện tích phần mặt phẳng bị phủ bởi các hình chữ nhật trên.

HƯỚNG DẪN GIẢI THUẬT

— Gọi mảng X[1..2N] là mảng chứa hoành độ các đỉnh của N hình chữ nhật.

— Gọi mảng Y[1..2N] là mảng chứa tung độ các đỉnh của N hình chữ nhật.

— Sắp xếp mảng X, mảng Y theo thứ tự tăng dần

— Lần lượt xét các hình chữ nhật con có tọa độ đỉnh trên bên phải là (xi+1,yi+1) và tọa độ đỉnh dưới bên trái là (xi, yi) với 1 < i < n —1. Nếu hình chữ nhật này thuộc một trong các hình chữ nhật ban đầu thì cộng thêm vào phần diện tích đang cần tìm diện tích của hình chữ nhật con này.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_6;
UsesCRT;
Const MaxN = 100;
Type HCN = Record x1,x2,y1,y2 : Integer; End;
Mang = Array[1..2*MaxN] of Integer,
```

```
Var N,i,j,k : Integer;
A : Array [L.MaxN] of HCN;
X,Y : mang;
Procedure Nhap;
Var f : Text;
Begin
Assign (f,'INP.BL6');
Reset (f);
Readln (f,n);
For i : = to N do
Readln (f, A[i]. xl, A[i].yl, A[i].x2, A[i].y2);
Close(f);
End;
Var
S : LongInt;
Procedure Sort(Var B:Mang);
Var  i,j,k : Integer;
Begin
For i := 2 to 2*N-1 do
For j:*= 2*N down to i do
If B[j] < B[j - 1] then
Begin
k := B[j];
Blj] := B[j-1];
B[j-1] := k;
End;
End;
Function KiemTra(i1,j1 : Integer) : Boolean;
Begin
KiemTra := True;
For k := 1 to N do
If (A[k].x1 <= X[i1-1]) and (X[i1] <= A[k].x2)
and (A[k].y1 <= Y[j1-1]) and (Y[j1] <= A[k].y2) then Exit.
KiemTra := False;
End;
Procedure Xuly;
```

```
Begin
Clrscr;
s := 0;
For i :=1 to N do
Begin
X[i*2-1] := A[i].x1;
X[i*2] := A[i].x2;
Y[i*2-1] := A[i] y1;
Y[i*2] := A[i].y2;
End;
Sort(X);
Sort(Y);
For i := 2 to 2*N do
For j := 2 to 2*N do
If KiemTra (i, j) then
S := S + (X[i] – X[i-1]) * (Y[j] – Y[j – 1]);
Write (’Dien tich: ‘, S);
Readln;
End;
BEGIN
Nhap;
Xuly;
END.
```

Bài 7 : Cho 2 đoạn thẳng A1B1 và A2B2 với Ai và Bi có tọa độ thực. Hãy kiểm tra xem hai đoạn thẳng có cắt nhau hay không.

Dữ liệu : tọa độ Ai và Bi được nhập từ bàn phím.

Kết quả: xuất ra màn hình.

VÍ DỤ:

A1 (0, 0), B1 (1, 1)

A2, (0, 1), B2 (1, 0)

Kết quả

Hai đoạn thẳng cắt nhau

A1 (0, 0), B1 (1, 1)

A2 (0,-1), B2 (-1, 0)

Kết quả

Hai đoạn thẳng không cắt nhau

HƯỚNG DẪN GIẢI THUẬT:

— Kiểm tra xem hai đường thẳng A1B1 và A2B2 có cắt nhau không.

— Nếu có thì kiểm tra xem giao điểm có thuộc đoạn thẳng A1B1 và A2B2 không. Nếu có thì chúng cắt nhau.

— Trong trường hợp đường thẳng A1B1 và A2B2 trùng nhau thì kiểm tra xem chúng có đoạn chung hay điểm chung không, (so sánh hoành độ)

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_7;
UsesCRT;
Const MaxN = 10;
Type        xy = Record X, y : Real; End;
Var  n : Integer;
T : Array [1..4] of xy;
Procedure Nhap;
Begin
Clrscr;
Write (‘Nhap vao toa do A1:’);
Readln (T[1].x, T[1].y);
Write (‘Nhap vao toa do B1:’);
Readln (T[2].x, T[2].y);
Write (‘Nhap vao toa do A2:’);
Readln (T[3].x, T[3].y);
Write (’Nhap vao toa do B2 : ‘);
Readln (T[4].x, T[4].y);
End;
Var a1, a2, b1, b2, c1, c2 : Real;
x1, x2, y1, y2 : Real;
D, Dx, Dy : Real;
Procedure Xuly;
Begin
a1 := T[1].y - T[2].y;
b1 := T[2].x - T[1].x;
c1 := T[1].x*T[2].y - T[2].x* T[1].y;
a2 := T[3].y - T[4].y;
b2 := T[4].x - T[3].x;
c2 := T[3].x* T[4].y - T[4].x* T[3].y;
```

```
D := a1* b2 - a2* b1;
Dx := b1* c2 - b2* c1;
Dy := c1* a2 - c2* a1;
If D < > 0 then
If (((Dx/D >= T[1].x) and (Dx/D <= T[2]JC))
or ((Dx/D >= T[2].x) and (Dx/D <= T[1].x)))
and (((Dy/D >= T[1].y) and (Dy/D <= T[2].y))
or ((Dy/D >= T[2].y) and (Dy/D <= T[1].y))) then
If (((Dx/D >= T[3].x) and (Dx/D <= T[4].x))
or ((Dx/D >= T[4].x) and (Dx/D <= T[3].x)))
and (((Dy/D >= T[3].y) and (Dy/D <= T[4].y))
or ((Dy/D >= T[4].y) and (Dy/D <= T[3].y))) then
Begin
Writeln (’Hai doan thang cat nhau ’);
Readln;
Halt;
End;
If (d = 0) and (Dx = 0) and (Dy = 0) then
If ((T[1].x <= T[3].x) and (T[1].x >= T[4].x))
or ((T[1].x <= T[4].x) and (T[1].x >= T[3].x)) then
Begin
Writeln (’ Hai doan thang cat nhau ’);
Readln;
Halt;
End;
Writeln (’Hai doan thang khong cat nhau’);
Readln;
End;
BEGIN
Nhap;
Xuly;
END.
```

Bài 8 : Cho trước hệ các đường tròn bằng nhau có bán kính R tâm là O1, O2,…, On. Hệ trên được gọi là tốt nếu các vòng tròn trên không chồng lên nhau đôi một (hai đường tròn có thể tiếp xúc nhau)

Kiểm tra xem hệ có tốt hay không ?

Dữ liệu: Cho trong file INP.BL8

+ Dòng 1: ghi số n là số lượng các đường tròn và R

+ n dòng tiếp theo mỗi dòng ghi hai số nguyên xi và yi là tọa độ Oi

Dữ liệu là các số nguyên. Các số ghi trên cùng một dòng cách nhau ít nhất một dấu cách.

Kết quả:Xuất ra màn hình

HƯỚNG DẪN GIẢI THUẬT:

- Với mỗi cặp đường tròn, ta kiểm tra xem khoảng cách giữa 2 tâm có lớn hơn 2 lần bán kính hay không.

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_8;
UsesCRT;
Var  n : Byte;
r,i,j : Integer;
C : Array[1..2,1..100] of Integer;
Procedure Nhap;
Var f : Text;
Begin
Assign (f, 'INP.BL8');
Reset (f);
Readln (f, n, r);
FillChar (C, SizeOf (C), 0);
For i := to n do
Readln (f, C[1, i], C[2, i]);
Close (f);
End;
Function KiemTra : Boolean;
Begin
Clrscr;
KiemTra := False;
If Sqr(C[1,i] - C[2,i]) + Sqr(C[1,j] - C[2,j])<4*r*r then Exit;
KiemTra := True;
End;
BEGIN
Nhap;
If KiemTra then Writein ('He duong tron tot')
```

```
Else Writeln('He duong tron khong tot');
Readln
END.
```

Bài 9 : Cho đa giác không tự cắt A1A2 ... AN với các đỉnh Ai(xi,yi) nguyên. Với điểm A(xA,yA) cho trước, hãy xác định xem A có nằm trong đa giác đã cho hay không (Trường hợp A nằm trên cạnh đa giác xem như A nằm trong đa giác).

Dữ liệu : cho trong file INP.BL9

+ Dòng đầu là số N (số lượng đỉnh của đa giác)

+ N dòng sau mỗi dòng chứa hai số nguyên xi và yi là tọa độ Ai

+ Dòng N + 2 ghi hai số xA và yA

Dữ liệu là các số nguyên.

HƯỚNG DẪN GIẢI THUẬT:

— Kiểm tra xem điểm A có trùng với đỉnh đa giác.

— Kiểm tra xem điểm A có thuộc cạnh đa giác.

— Tìm giao điểm, nếu có, của tia Ax (Ax // Ox và Ax hướng theo phần dương trục hoành) với các cạnh đa giác. Trường hợp tia Ax chứa phần đoạn thẳng cạnh đa giác ta xem như tia Ax có 1 điểm chung với cạnh này.

— Đếm số giao điểm, nếu là số lẻ thì điểm A thuộc đa giác

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG_TRINH_MAU_BAI_7_9;
UsesCRT;
Const MaxN = 10;
Typexy = Record x,y : Real; End;
Var  n : Byte;
A : Array [l..MaxN+2] of xy;
B : Array[1..2*MaxN] of xy; i,j : Integer;
a1, b1, c1, a2, b2, c2, x1, x2, y1, y2 : Real;
D, Dx, Dy : Real;
k, dem, dem1 : Integer;
Procedure Nhap;
Var  f : Text;
Begin
Assign (f,'INP.BL9');
Reset(f);
Readln (f,N);
FiliChar (A, SizeOf (A), 0);
```

```
Fill Char (B, SizeOf (B), 0);
For i := 1 to N do
Readln (f, A[i].x, A[i].y);
Readln (f, A[N + 2].x, A [N + 2].y);
A[N+1] := A[1];
Close(f);
Clrscr;
End;
Function KiemTraDinh : Boolean;
Begin
KiemTraDinh := True;
For i := 1 to N do
If (A[i].x = A[N+2].x) and (A[i].y=A[N+2].y) then Exit;
KiemTraDinh := False;
End;
Function KiemTraCanh : Boolean;
Begin
KiemTniCanh := True;
For i := 1 to N do
If A[n+2].x*(A[i].y-A[i+1].y)+A[N+2].y*(A[i+1].x^A[i].x)+A[i].x*A[i+1].y-
A[i].y*A[i+1].x=0 then Exit;
KiemTraCanh := False;
End;
Procedure Ghi (x, y : Heal);
Begin
For k := 1 to dem do
If (B[k].x = x) and (B[k].y = y) then Exit;
Inc (dem);
B[dem].x := x;
B[dem].y := y;
End;
Function KiemTra : Boolean;
Begin
KiemTra := True;
If KiemTraDinh then Exit;
If KiemTraCanh then Exit;
```

```
dem := 0; deml := 0;
For i := 1 to N do Begin
a1 := A[i].y - A[i + 1].y;
b1 := A[i + 1].x — A[i].x;
c1 := A[i].x* A[i + 1].y — A[i].y* A[i + 1].x;
a2 := 0;
b2 := 1;
c2 := — A(N + 2).y;
If A[i].y > A[i + 1].y then
Begin
y1 := A[i + 1].y; y2 := A[i].y;
End
Begin
y2 := A[i + 1].y;
y1 := A[i].y;
End;
If A[i].x > A[i + 1].x then
Begin
x1 := A[i + 1].x;
x2 := A[i].x;
End
Else
Begin
x2 := A[i + 1].x; x1 := A[i].x;
End;
D := a1* b2 - a2* b1;
Dx := b1* c2 - b2* c1;
Dy := c1* a2 — c2* a1;
If D < > 0 then
If (Dx/D >= A[N + 2].x) and
(Dx/D >= x1) and (Dx/D <= x2) and
(Dy/D >= y1) and (Dy/D <= y2) then
Ghi (Dx/D, Dy/D);
If ((A[N+2].x <= A[i].x) or (A[N+2].x <= A[i+1].x))
and (A[N+2].y = A[i].y) and (A[N+2].y = A[i+1].y) then
Inc(Dem1);
```

```
End;
If Odd(dem+dem1) then Exit;
KiemTra := False;
End;
BEGIN
Nhap;
If KiemTra then
Write ('Diem A(', A[N + 2].x : 0 : 2,',', A[N + 2].y : 0 : 2,') nam trong da giac')
Else
Write ('Diem A(', A[N + 2].x : 0 : 2,',', A[N + 2].y : 0 : 2,') khong nam trong da giác');
Readln;
END.
```

Bài 10 : Cho một bảng hình chữ nhật có m dòng và n cột các ô vuông (kích thước x) và m,n<20. Mỗi ô vuông có một trong 4 màu xanh, đỏ, tím, vàng. Hai ô được gọi là liền kề nhau nối tiếp nếu chúng có chung một cạnh cùng màu. Hai ô được gọi là liền kề nhau nếu chúng liền kề nhau trực tiếp hay liền kề nhau với một ô trung gian. Hình là một tập hợp các ô cùng màu liền kề nhau sao cho với mọi ô cùng mau không thuộc hình thì không thể liền kề với một ô nào trong hình. Diện tích của hình là số lượng ô thuộc hình. Hai hình (có thể màu khác nhau) được gọi là bùng nhau nếu chúng đặt trùng khít lên nhau (sau một số phép tính tiến hay quay 90 độ. Tất cả các hình bùng nhau tạo thành một nhóm hình. Cho m, n và một bảng chữ nhật như trên, hãy lập trình giải bài toán:

Câu 1 : Tìm số lượng các hình có trong bảng. Hiện bảng thông tin gồm m dòng, n cột các số nguyên dương mà ô thuộc hình i sẽ có giá trị i.

Câu 2 : Có bao nhiêu nhóm hình thuộc bảng. Hiện bảng thông tin gồm rn dòng, n cột các số nguyên dương mà ô thuộc nhóm hình i sẽ có giá trị i.

Dữ liệu : cho trong file INP.BL0 có dạng

+ Dòng 1 có hai số m và n

+ Dòng i + 1 (1 =< i =< m) có một chuỗi n ký tự chứa các ký tự X (chỉ màu xanh), D (chỉ màu đỏ), T (chỉ màu tím), V (chỉ màu vàng).

Kết quả : xuất ra file OUT.BLO có dạng

+ Dòng đầu tiên chứa số lượng hình tìm được.

+ m dòng tiếp theo hiện bảng thông tin về các hình tìm được:

Mỗi dòng chứa n số theo quy cách 4 cột cho một số.

+ Dòng m + 2 chứa số lượng nhóm hình

+ m dòng tiếp theo hiện bảng thông tin về các nhóm hình tìm được mỗi dòng chứa n số theo quy cách 4 cột cho một số.

+ Dòng cuối cùng ghi diện tích các hình tìm được theo thứ tự trong câu 1

HƯỚNG DẪN GIẢI THUẬT:

— Với mỗi ô (i, j) chưa được đánh dấu, ta loang ra bốn ô xung quanh nếu chúng chưa được đánh dấu và có cùng màu với ô (i, j). Cứ tiếp tục cho đến khi không thực hiện được nữa, ta sẽ có được một hình.

— Với hai hình có cùng kích thước (chiều dài và chiều rộng), ta xét xem chúng có giống nhau hay không (sau khi tịnh tiến về cùng một vị trí hay sau một số phép quay với góc quay 90°)

CHƯƠNG TRÌNH MẪU:

```
PROGRAM CHUONG _TRINH_MAU_BAI_7_10;
Uses Crt;
Const Max = 20;
Di : array [1..4] of Integer = (— 1, 0, 1, 0);
Dj : array [1..4] of Integer = (0, 1, 0, — 1);
Var
A, B, S, T : array [1..Max, 1.. Max] of Byte;
Li, Lj : array [1..Max* 10] of Byte;
Ni, Nj : array [1..Max* 10] of Byte;
D : array [1..Max*10] of Byte;
G : array [1..Max*10] of set of Byte;
X : array [1..Max*10] of Byte;
n, m : Byte;
c, z : Integer;
f1, f2 : String;
Procedure Chuanbi;
Begin
Clrscr,
f1 := 'INP.BL0';
f2 := 'OUT.BL0';
c := 0;
z := 0;
Fillchar (B, SizeOf(B), 0);
Fillchar (Li, SizeOf(Li), 0);
```

```
Fillchar (Lj, SizeOf(Lj), 0);
Fillchar (Ni, SizeOf(Ni), $FF);
Fillchar (Nj, SizeOf(Nj), $FF);
Fillchar (D, SizeOf(D), 0);
Fillchar (X, SizeOf(X), 0);
End;
Procedure Nhap;
Var  f : Text;
i, j : Byte; t : Char;
Begin
Assign (f, F1);
Reset (f);
Readln (f, m, n);
For i := 1 to m do
Begin
For j := 1 to n do
Begin
Repeat
Read(f, t);
Until t <> #32;
Case t of
'X' : A[i, j] := 1;
'D' : A[i, j] := 2;
'T' : A[i, j] := 3;
'V' : A[i, j] := 4;
End;
End;
Readln(f);
End;
Close(f);
End;
Procedure Loang(i, j : Byte);
Var  k : Byte;
Begin
B[i, j] := c;
Inc(D[c]);
```

```
If i < Ni[c] then Ni[c] := i;
If i > Li[e] then Li[e] := i;
If j < Nj[c] then Nj[c] := j;
If j > Lj[c] then Lj[c] := j;
For k := 1 to 4 do
If (i + Di[k] > 0) and (j + Dj[k] > 0) then
If (B [i + Di[k], j + Dj[k]] = 0) and
(A[i + Di[k], j -f Dj[k]] = A[i, j]) then
Loang (i + Di[k], j + Dj[k]);
End;
Procedure Cau1;
Var  i, j : Byte;
f : Text;
Begin
For i := 1 to m do
For j : = 1 to n do
If B[i, j] = 0 then
Begin Inc (c);
Loang (i, j);
End,
Assign (f,f2);
Rewrite (f);
Writeln (f,c);
For i := 1 to m do
Begin
For j := 1 to n do
Write (f, B[i, j] : 4);
Writeln(f);
End;
Close(f);
End;
Procedure Thu(k, h : Integer);
Var i, j, q : Byte;
mk, mh, nk, nh : Byte;
Function OK : boolean;
Var i, j : Byte;
```

```
Begin
For i := 1 to mk do
For j := 1 to nk do If (All, j] Xor S[i, j]> > 0 then Begin
OK : false;
Exit;
End;
OK:= true;
End;
Procedure Quay;
Var  i, j, y : Byte;
Begin
Fillchar (T, SizeOf (T), 0);
For i := 1 to mh do
For j := 1 to nh do
If S[i, j] > 0 then
T[nh - j + 1, i] := 1
S := T;
y := mh;
mh := nh;
nh := y;
End;
Begin
If D[k] < > D[h] then exit;
Fillchar (A, SizeOf(A), 0);
Fillchar (S, SizeOf(S), 0);
For i := Ni[k] to Li[k] do
For j := Nj[k] to Lj[k] do
If B[i, j] = k then A[i - Ni[k] + 1, j - Nj[k] + 1] ;
For i := Ni[h] to Li[h] do
For j := Nj[h] to Lj[h] do
If B[i, j] = h then S[i - Ni[h] + 1, j - Nj[h] + 1] := 1;
mk := Li[k] - Ni[k] + 1;
nk := Lj[k] - Nj[k] + 1;
mh := Li[h] - Ni[h] + 1;
nh := Lj[h] - Nj[h] + 1;
If (mk = mh) and (nk = nh) and OK then
```

```
Begin
G[z] := G[z] + [h];
X[h] := z;
Exit;
End;
For q := 1 to 3 do
Begin
Quay,
If (mk = mh) and (nk = nh) and OK then
Begin
G[z] := G[z] + [h];
X[h] := z;
Exit;
End;
End;
End;
Procedure Cau2;
Var  k, h, i, j : Integer;
f : Text;
Begin
For k :=1 to c— 1 do
If X[k] = 0 then
Begin
Inc(z);
X[k] := z;
G[z] := [k];
For h := k + 1 to c do
If X[h] = 0 then
Thu(k, h);
End;
If X[h] = 0 then
Begin
Inc(z);
X[c] := z;
G[z] := [c];
End;
```

```
Assign(f, f2);
Append(f);
Writeln(f, z);
For i := 1 to m do
Begin
For j := 1 to n do
Write (f, X[B[i, j]] : 4);
Writeln(f);
End;
For k := 1 to c do
Writeln(f,D[k]);
Close(f);
End;
BEGIN
Chuanbi;
Nhap;
Cau1;
Cau2;
END.
```

CHƯƠNG 3 CÁC ĐỀ THI TIN HỌC QUỐC TẾ

§1. ĐỀ THI TIN HỌC QUỐC TẾ LẦN V

(Năm 1993)

Bài 1 : NECKLACE

Cho n chuỗi hạt (n =< 180) . Trong chuỗi có một số hạt màu đỏ, một số hạt khác màu xanh, những hạt còn lại màu trắng. Các hạt trong chuỗi được sắp xếp một cách ngẫu nhiên. Các hạt được biểu diễn bằng xâu các kí tự b và r, trong đó b kí hiệu màu xanh, còn r — màu đỏ, vậy xâu đó có dạng như sau : brbrrrbbbrrrbrrbbrbbbrrrb. Giả sử bạn cắt chuỗi hạt, trải thẳng ra và sau đó chọn các hạt cùng màu từ một đầu cho đến khi gặp hạt màu khác và cũng làm tương tự như vậy ở đầu kia (màu của hạt đầu tiên ở phía ấy có thể không cùng màu với các hạt đã được chọn trước đó ở đầu kia). Hãy xác định điểm nơi cần phải cắt chuỗi hạt sao cho số lượng các hạt được chọn là nhiều nhất.

Viết chương trình để thực hiện:

1) Đọc một hình dung từ file dữ liệu NECKLACE.DAT, mỗi dòng là một hình trạng. Ghi dữ liệu đó ra file NECKLACE.SOL.

2) Với mỗi hình trạng hãy xác định số lớn nhất M của số hạt được chọn tương ứng với cách cắt (Ví dụ : 8 between 9 and 10).

3) Ghi ra file NECKLACE.SOL số M và điểm cắt (Ví dụ : 8 between 9 and 10). Các kết quả của các hình trạng khác nhau phải được cách nhau bởi một bản ghi trắng.

Bài 2 : COMPANY

Một số hãng nào đó có một số cổ phần ở một số hãng khác. Ví dụ hãng Ford chiếm 12 % cổ phần của hãng Mazda. Ta nói rằng hãng A kiểm soát hãng B, nếu ít nhất một trong các điều kiện sau được thỏa mãn:

a) A = B

b) A chiếm nhiều hơn 50% cổ phần của B hoặc A kiểm soát các hãng C(1), C(2), ..., C(k) sao cho C(i) chiếm x(i) % cổ phần và x(1) +...+ x(k) > 50.

Bài toán, phải giải là như sau:

Cho một danh sách bộ ba (i, j, p) với nghĩa là hãng i chiếm p % cổ phần của hãng j. Hãy tìm tất cả các cặp (k, s) sao cho hãng k kiểm soát hãng s.

Viết chương trình thực hiện:

1) Đọc từ file COMPANY.DAT chứa danh sách các bộ ba (i, j, p) cần phải xét với từng trường hợp (i, j, p là số nguyên dương). Các trường hợp khác nhau được cách nhau bởi một bản ghi trắng.

2) Tìm tất cả các cặp (h, s) sao cho háng h kiểm soát hãng s.

3) Ghi ra file COMP ANY.SOL tít cả các cặp (h, 8) tìm được. Các cặp (h, s) phải được ghi thành các bản ghi liên tiếp và theo thứ tự tăng dần của h. Các kết quả với trường hợp khác nhau phải được cách nhau bằng một bảng ghi trống.

Bài 3 : RECTANGLE

N hình chữ nhật màu khác nhau được xếp chổng lên một tờ giấy trắng. Kích thước tờ giấy đó là : a cm chiều rộng và b cm chiều dài. Các hình chữ nhật được đặt sao cho các cạnh của nó song song với các lề của tờ giấy, tất cả các hình chữ nhật đều nằm bên trong tờ giấy và các hình khác nhau của các màu khác nhau sẽ được nhìn thấy. Hai miền cùng màu được xem như là một phần của một hình nếu như chúng có ít nhất một điểm chung; ngược lại chúng được xét như là các hình khác nhau. Yêu cầu tính diện tích của mỗi hình. Các giá trị a, b đều là những số nguyên dương không lớn hơn 30.

Hệ tọa độ dùng có tâm là điểm ở giữa tờ giấy và các trục song song với lề của tờ giấy.

Các tập dữ liệu khác nhau được ghi ở file dữ liệu RECTANG.DAT:

+ a, b, N được ghi ở dòng đầu tiên của file dữ liệu và được cách nhau bằng dấu cách.

+ Mỗi dòng trong nhóm N dòng tiếp theo : Tọa độ đỉnh trái dưới, phải trên của hình chữ nhật và màu của hình chữ nhật (được biểu hiện bằng một số nguyên dương trong khoảng từ 1 đến 64). Màu trắng được biểu diễn bằng số 1.

Thứ tự các bản ghi là tương ứng với thứ tự đã được sử dụng khi đạt các hình chữ nhật lên trên tờ giấy. Các tập dữ liệu khác nhau được cách nhau bởi một bản ghi trắng.

Viết chương trình thực hiện:

1) Đọc tập dữ liệu từ file RECTANG.DAT.

2) Tính diện tích mỗi hình.

3) Ghi ra file ASCII có tèn RECTANG.SOL màu và diện tích của mỗi hình (xem ví dụ dưới đây) các bản ghi được sắp xếp theo thứ tự tăng dần của màu. Các kết quả ứng với các tập dữ liệu khác nhau phải cách nhau bởi một bản ghi trắng.

Bài 4

Bạn là người đoạt giải trong một cuộc thi do Hãng hàng không CANADA tổ chức. Giải thưởng là một chuyến du lịch không mất tiền vòng quanh Canada bằng máy bay của hãng. Hành trình chỉ được đi từ Tây sang Đông cho đến khi gặp thành phố ở phía Đông nhất rồi quay trả lại theo hướng từ Đông sang Tây cho đến khi gặp lại thành phố xuất phát. Không được thăm một thành phố nào quá một lần ngoại trừ thành phố xuất phát mà bạn phải thăm đúng hai lần (một lần khi xuất phát và một lần khi kết thúc hành trình). Bạn không được sử dụng máy bay của hãng khác hay bất cứ phương tiện giao thông nào khác.

Bài toán phải giải là : Cho một danh sách các thành phố và một danh sách các tuyến bay trực tiếp giữa hai thành phố. Hãy vạch ra hành trình thỏa mãn các điều kiện nêu trên và cho phép bạn thăm được càng nhiều thành phố càng tốt.

Dữ liệu : cho trong file ITIN.DAT

+ Dòng đầu : chứa số N là số lượng thành phố và V là số lượng các tuyến bay trực tiếp.

+ Mỗi dòng trong N dòng tiếp theo : Tên thành phố. Tên thành phố được sắp theo thứ tự từ Tây sang Đông nghĩa là thành phố thứ i sẽ là thành phố phía Đông của thành phố j khi và chỉ khi i > j không có hai thành phố nào cũng ở trên một kinh tuyến). Tên của mỗi thành phố là một xâu nhiều nhất là 15 kí tự, mỗi kí tự là chữ số hoặc chữ cái La tinh, ví dụ : AGR34 hoặc BEL4.

+ Mỗi dòng trong V dòng tiếp theo : tên của hai thành phố có trong danh sách tên các thành phố, tên hai thành phố đó cách nhau bằng một dấu cách. Nếu

cặp CITY1 CITY2 xuất hiện ở một dòng nào đó thì có nghĩa 1 h tồn tại tuyến bay trực tiếp từ CITY1 đến CITY2 và đồng thời củng cố tuyến bay trực tiếp từ CITY2 đến CITY1.

+ Mỗi cặp dữ liệu khác nhau được ghi cách nhau bằng một bản ghi trống

Kết quả : ghi ra file ITIN.SOL theo quy định.

+ Dòng đầu là số lượng các thành phố trong tập dữ liệu vào.

+ Dòng 2 là số lượng M các thành phố khác nhau mà hành trình đi qua.

+ M + 1 dòng tiếp theo mỗi dòng ghi tên một thành phố theo đúng trình tự đi của hành trình.

Nếu kết quả là vô nghiệm thì trong file kết quả chỉ ghi 2 dòng:

+ Dòng đầu là số lượng các thành phố trong tập dữ liệu vào.

+ Dòng 2 ghi thông báo "NO SOLUTI0N"

+ Mỗi tập dữ liệu khác nhau được ghi cách nhau băng một bản ghi trống.

§2. ĐÊ THI TIN HỌC QUỐC TẾ LẦN VI

(Năm 1994)

Bài 1 : TRIANGLE

Hình 1 là một bảng tam giác các số nguyên không âm. Hãy viết chương trình để tính tổng lớn nhất các số trên đường đi từ đỉnh tam giác và kết thúc tại một điểm nào đó ở đãy tam giác.

+ Mỗi nước đi ta được quyền đi thẳng xuống bên trái hay bên phải của số đố.

+ Số hàng trong tam giác lớn hơn 1 và < 100 + Các số trong tam giác đều là số nguyên không âm và nhỏ hơn 100.

Dữ liệu: cho trong file INPUT.TXT

+ Dòng đầu ghi số lượng các dòng trong tam giác (N)

+ Dòng i+1(1 =< i =< N) ghi i số

Ví dụ :

5

7

3 8

8 1 0

2 7 4 4

4 5 2 6 5

Kết quả: Xuất ra file OUTPUT.TXT tổng lớn nhất tìm được

Bài 2: CASTLE

Hình 1 biểu diễn bản đồ một lâu đài (Mũi tên chỉ bức tường cần loại bỏ)

Hãy viết phương trình tính:

1) Lâu đài có bao nhiêu phòng.

2) Phòng lớn nhất có diện tích bao nhiêu.

3) Bức tường nào cần loại bỏ để phòng càng rộng càng tốt.

Lâu dài dược chia thành mxn(m =< 50,n =< 50) modul vuông. Mỗi modul vuông có từ 0 đến 4 bức tường.

INPUT DATA

Bản đồ được lưu trữ trong file INPUT.TXT ở dạng các số cho mối modul

— File bắt đầu từ số lượng các modul theo hướng Bắc — Nam và số lượng các modul theo hướng Đông — Tây.

— Trong các dòng tiếp theo, mỗi modul được mô tả bởi một số p(0 =< p =< 15) . Số đó là tổng cùa : 1 (= tường phía Tây), 2 (= tường phía Bắc), 4 (= tường phía Đông), 8 (= tường phía Nam).

— Các bức tường ở bên trong được xác dịnh 2 lần. Ví dụ : bức tường phía Nam trong modul 1, 1 đồng thời là bức tường phía Bắc trong modul 2, 1. Cho biết lâu đài có ít nhât 2 phòng.

- OUTPUT DATA

Trong file ra OUTPUT.TXT viết ra 3 dòng:

+ Dòng thứ nhất viết số lượng phòng, tiếp theo là diện tích của phòng lớn nhất (tính theo số modul) và bức tường cần loại bỏ (trước tiên là hàng, sau đó là cột của modul có tường đó) và hướng của bức tường cần loại bỏ. (Chỉ cần chỉ ra một lời giải).

Hình 1 biểu diễn một hình vuông. Mỗi dòng, mỗi cột và hai đường chéo có thể đọc thành một số nguyên tố có 5 chữ số. Các dòng đọc từ trái sang phải. Các cột đọc từ trên xuống dưới. Cả hai đường chéo được đọc từ trái sang phải. Sử dụng dữ liệu trong file INPUT.TXT để viết chương trình xây dựng các hình vuông như vậy:

+ Các số nguyên tố phải có cùng tông các chữ số (11 trong ví dụ)

+ Chữ số ở đỉnh trái trên của hình vuông được xóc định trước (1 trong ví dụ).

+ Một số nguyên tố có thể được sử dụng nhiều lần trong một hình vuông.

+ Nếu có nhiều nghiệm thì tất cả các nghiệm đó phải được chỉ ra.

+ Số nguyên tố có 5 chữ số không được bắt đầu bằng chđ số 0.

INPUT DATA

Chương trình đọc dữ liệu từ file INPUT.TXT : trước tiên là tổng các chữ số của số nguyên tố, tiếp theo là chữ số ở góc trên bên trái.

OUTPUT DATA

Trong file OUTPUT.TXT ghi 5 dòng cho mỗi nghiệm tìm được. Mỗi dòng là một số nguyên tố có 5 chữ số.

Bài 4. CLOCK

Có 9 chiếc đồng hồ đặt trong một mảng 3*3. Mục tiêu là biến đổi tất cả các đồng hồ về vị trí 12 giờ. Có 9 cách biến đổi khác nhau các vị tri của các đồng hồ. Mỗi cách gọi là một biến đổi được đánh số từ 1 đến 9. Mỗi phép biến đổi làm quay tất cả các đồng hồ (trong phần được đánh dấu trong hình) một góc 90 độ theo chiều kim đồng hồ.

Dữ liệu : trong file INPUT.TXT

(quy ước 0=12 giờ, 1 = 3 giờ, 2 = 6 giờ, 3 = 9 giờ). Ví dụ:

3 3 0

2 2 2

2 1 2

Kết quả : ghi ra file OUTPUT.TXT một dãy ngắn nhất các biến đổi (các số) làm quay tất cả các kim đồng hồ về vị trí 12 giờ. Chỉ cần một nghiệm.

Bài 5 : BUS LINE

Một người đứng ở một bến xe buýt từ 12 giờ và ở đó từ 12 giờ 00 đến 12 giờ 59. Bến xe này dùng cho một số tuyến. Người đó ghi lại các thời điểm đến của các xe. Các xe thuộc cùng một tuyến đến bến một cách đều đặn (cùng khoảng thời gian giữa hai xe liên tiếp) trong suốt khoảng thời gian từ 12 giờ 00 đến 12 giờ 59. Các thời điểm được tính bằng số phút nguyên từ 0 đến 59. Các xe thuộc mỗi tuyến dừng ở bến ít nhất 2 lần. Số tuyến xe không quá 17. Các xe thuộc các tuyến khác nhau có thể đến cùng một lúc. Một số tuyến xe có thể có cùng thời điểm đến ban đầu và khoảng thời gian giữa hai xe liên tiếp cùng tuyến. Nếu hai tuyến xe có cùng thời điểm đến ban đầu và khoảng thời gian giữa hai xe liền tiếp cùng tuyến thì chúng là khác nhau và cùng phải xuất hiện như nhau. Căn cứ vào bản ghi nhận các thời điểm đến của các xe, hãy tìm một số ít nhất tuyến xe đã dừng lại bến phù hợp đúng với bản ghi nhận này. Với mỗi tuyến xe, hãy thông báo thời điểm đến đầu tiên và thời khoảng giữa hai lần đến liên tiếp.

Dữ liệu vào được cho bởi file INPUT.TXT trong đó dòng thứ nhất ghi số N(N =< 300) biểu thị bao nhiêu lần xe đến được ghi nhận, dòng thứ hai ghi các thời điểm xe đến theo thứ tự tăng dần. Ví dụ:

0 0 3 5 13 13 13 15 21 26 26 27 29 37 39 39 45 51 52 52 53

Dữ liệu ra ghi vào file OUTPUT.TXT mỗi tuyến xe một dòng gồm hai số, số thứ nhất là thời điểm đến đầu tiên và số thứ hai là thời khoảng giữa hai lần đến liên tiếp.

Với ví dụ trên của file INPUT.TXT, file OUTPUT.TXT sẽ là

0 13

0 13

3 12

5 8

Bài 6 : SECTOR

Một hình tròn được chia thành N hình quạt. N =< 6 . Giả sử trên mỗi hình quạt ta ghi một số nguyên dương nào đó. Khi đó ta có thế tạo lập một số số nguyên dương bằng cách cộng các số ghi trên một số (>= 1) các hình quạt kế nhau liên tiếp. Cho trước hai số nguyên dương M và K đều không quá 20, hãy ghi vào mỗi hình quạt một số nguyên >= K sao cho từ các số có thể tạo lập được như trên, ta có thể trích ra một dãy nhiều nhất các số nguyên liên tiếp bắt đầu từ số M.

Dữ liệu : cho trong file INPUT.TXT gồm 3 dòng:

+ Dòng thứ nhất ghi số N.

+ Dòng thứ hai ghi số M.

+ Dòng thứ ba ghi số K.

Kết quả : ghi ra file OUTPUT.TXT trong đó dòng thứ nhất là số I thể hiện dãy số nguyên liên tiếp dài nhất tạo lập từ M tới I. Từ dòng thứ hai, mỗi dòng là một cách ghi khác nhau nếu chúng không nhận được từ nhau nhờ hoán vị vòng tròn. Mối cách sắp xếp bắt đầu từ số nhỏ nhất.

§3. ĐỀ THI TIN HỌC QUỐC TẾ LẦN VII

(Năm 1995)

Bài 1 : PACK

Cho 4 hình chữ nhật. Tìm hình chữ nhật (mới) nhỏ nhất mà ta có thể đặt 4 hình chữ nhật đã cho vào trong đó sao cho các hình chữ nhật này không đè lên nhau.

Hình chữ nhật nhỏ nhất là hình chữ nhật có diện tích nhỏ nhất.

C1 4 hình chữ nhật đó phải có các cạnh song song với cạnh của hình chữ nhật chứa chúng.

Hình trên là sáu cách xếp cơ bản của 4 hình chữ nhật. Chỉ có 6 cách cơ bản này vì một cách bất kì khác có thể nhận được từ 1 trong 6 cách này bằng cách quay hay lấy đối xứng.

Hãy chỉ ra mọi hình chữ nhật nhỏ nhất thỏa mãn đề bài.

Dữ liệu : file INPUT.TXT gồm 4 dòng. Mỗi dòng mô tả một hình chữ nhật bằng 2 số nguyên dương là độ dài các cạnh của hình chữ nhật (độ dài này từ 1 đến 50).

Kết quả : file OUTPUT.TXT : Số dòng của file nhiều hơn 1 so với số lời giải. Dòng thứ nhất chứa 1 số nguyên duy nhất là diện tích của hình chữ nhật nhỏ nhất.

Mỗi dòng tiếp theo là 1 lời giải, mô tả bởi 2 số nguyên p và q, p =< q , là độ dài của 2 cạnh hình chữ nhật cần tìm. Những dòng này phải xếp theo thứ tự tăng dần của p và phải khác nhau đôi một.

Bài 2 : SHOP

Trong một cửa hàng, mỗi loại hàng có một giá. Ví dụ giá một bông hoa là 2 đồng và giá một cái bình là 5 đồng. Để thu hút nhiều khách hàng, cửa hàng đề ra một số cách bán đặc biệt.

Một cách bán đặc biệt liên quan đến việc bán một hay một số mặt hàng với giá chung được giảm.

Ví Dụ : 3 bông hoa bán với giá 5 đồng thay vì 6 đồng.

2 cái bình và 1 bông hoa bán với giá 10 đồng thay vì 12 đồng.

Viết chương trình tính giá mà một khách hàng phải trả cho một nhu cầu mua hàng để tận dụng một cách tối ưu các cách bán đặc biệt, nghĩa là phải trả ít nhất.

Ví dụ : Với các giá và các cách bán đặc biệt nêu trên, giá thấp nhất để mua 3 bông hoa và 2 cái bình là 14 đồng; 2 bình và 1 hoa là 10 đồng; 2 hoa với giá thường là 4 đồng.

Dữ liệu : cho trong 2 file INPUT.TXT và OFFER.TXT

File thứ nhất mô tả nhu cầu mua. File thứ hai mô tả các cách bán đạc biệt. Trong cả hai file, chỉ có các số nguyên dương.

File INPUT.TXT gồm b+1 dòng:

+ Dòng đầu chứa số b là số loại hàng cần mua (0 < b < 5)

+ Mỗi dòng trong b dòng tiếp theo ghi 3 số c, k, p. Giá trị C là mã của loại hàng. (1 =< C =< 999) . Giá trị k là số đơn vị hàng cần mua với mã C (1 =< k =< 5) . Giá trị p là giá bình thường của một đơn vị hàng với mã C (1 =< p =< 999) .

Chú ý : Trong một yêu cầu có không quá 5 * 5 = 25 đơn vị hàng.

File OFFER.TXT gồm s + 1 dòng:

+ Dòng đầu chứa số s là số cách bán đặc biệt (0 =< s =< 99)

+ Mỗi dòng trong s dòng tiếp theo mô tả một cách bán đặc biệt.

Số đầu tiên n của mối dòng như vậy là số loại hàng trong cách bán đặc biệt tương ứng với dòng đó (1 < n < 5) ; n cặp số tiếp theo (c, k) trong đó c là mã loại hàng, k là số đơn vị hàng đó (1 =< k =< 5; 1 =< c =< 999). Số p cuối cùng trong dòng là giá đã được giảm trong lô hàng này. Giá đã được giảm nhỏ hơn tổng các giá bình thường.

Các số ghi trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

Kết quả : xuất ra file OUTPUT.TXT : chi phí thấp nhất phải trả cho nhu cầu mua trong file vào.

VÍ DỤ:

INPUT.TXTOFFER.TXT

Bài 3 LGame

Các trò chơi chữ là phổ biến ở nhà và trên TV. Trong một cách chơi, mỗi chữ có một giá trị, và bạn tập hợp các chữ cái để lập thành một hay nhiều từ cho giá trị (điểm) cao nhất có thể được. Mặc dù bạn đã tìm được một cách tạo lập từ nào đó, bạn vẫn phải thử mọi từ mà bạn biết, thinh thoảng phải soát lỗi chính tả và sau đó tính giá trị (điểm). Rõ ràng điều đó có thể làm một cách chính xác hơn nhờ máy tính

Cho các giá trị trong hình trên, một danh sách các từ tiếng Anh và một dãy các chữ cái để tạo lập các từ, hãy tìm các từ hay các cặp từ với giá trị cao nhất có thể lập được.

INPUT DATA : File INPUT.TXT gồm một dòng chứa một xâu gồm các chữ cái thường (từ V đến V) : dãy các chữ cái để tạo lập các từ. Xâu này gồm ít nhất 3 và nhiều nhất 7 chữ cái theo thứ tự bất kì.

File 'từ điển' WORDS.TXT gồm nhiều nhất 40000 dòng. Cuối file là một dòng với một dấu chấm duy nhất. Mỗi một trong các dòng khác chứa một xâu gồm ít nhất 3 và nhiều nhất 7 chữ cái thường. File WORDS.TXT không chứa các từ trùng lặp. Các từ trong file được sắp xếp theo vần a, b, c.

OUTPUT DATA : File OUTPUT.TXT, chương trình bạn phải ghi giá trị cao nhất, và trên mỗi dòng tiếp theo, mọi từ hay cặp từ thuộc file WORDS.TXT có giá trị đó. Dùng các giá trị của chữ trong hình đã cho.

Khi một tổ hợp của hai từ có thể được tạo lập với xâu đã cho, các từ phải được ghi trên cùng một dòng và cách nhau bởi dấu trống. Số lần xuất hiện của mỗi chứ cái trong dòng ra không vượt quá số lần xuất hiện chữ cái đó trong dòng vào. Mỗi cặp từ có thể là hai từ như nhau.

Bài 4 : RACE

Hình dưới cho ví dụ về một bản đồ dùng cho một cuộc dạo phố. Bạn thấy một số đỉnh đánh số từ 0 đến N và một số mũi tên nối chúng.

Đỉnh 0 là đỉnh xuất phát, vh đỉnh N là đỉnh kết thúc. Các mũi tên biểu thị các phố một chiều. Người tham gia cuộc dạo đi từ đỉnh này đến đĩnh kia theo các phố chỉ theo hướng mũi tên. Tại mỗi đỉnh, người đó có thể chọn một mũi tên bất kỳ để đi theo.

Một bản đồ lập tốt có các tính chất:

1. Từ đỉnh xuất phát có thể đi đến mọi đỉnh trên bản đồ.

2. Từ một đỉnh bất kỳ có thể đi đến đỉnh kết thúc.

3. Không có mũi tên ra từ đỉnh kết thúc.

Người tham gia không nhất thiết đi qua mọi đỉnh để đến đỉnh kết thúc. Tuy nhiên có một số đỉnh không thể tránh được. Theo ví dụ trên thì đó là các đỉnh 0, 3, 6, 9. Cho một bản đồ lập tốt, hãy xác định tập hợp các đỉnh không thể tránh khỏi (ngoại trừ đỉnh 0 và N).

Giả sử rằng cuộc dạo chơi diễn ra trong 2 ngày liên tiếp. Với mục đích đó, bản đồ chia thành 3 bản đồ, mỗi ngày một bản đồ. Trong ngày 1, xuất phát từ 0 và kết thúc tại đỉnh chia cắt nào đó. Ngày 2 xuất phát từ đỉnh chia cắt và kết thúc tại N. Cho bản đô lập tốt, cần tìm tất cả các đỉnh chia cắt. Đỉnh s gọi là đỉnh chia cắt nếu s khác 0 và N và bản đồ được chia thành 2 bản đổ lập tốt và không có mũi tên nào chung và chỉ có đỉnh chung là s.

Dữ liệu : trong file INPUT.TXT cho một bản đồ lập tốt với không quá 50 đỉnh và không quá 100 mũi tên.

Trong N dòng đầu, dòng thứ i, 1 =< i =< N , ghi các đỉnh cuối của các mũi tên xuất phát từ đỉnh i — 1. Mỗi dòng kết thúc bằng số -2. Dòng cuối cùng ghi số -1.

Kết quả : viết ra 2 dòng trong file OUTPUT.TXT.

Dòng thứ nhất ghi số lượng đỉnh không tránh được trong bản đồ, tiếp theo là các đỉnh không tránh được theo thứ tự tùy ý.

Dòng thứ hai ghi các đỉnh chia cắt trong đồ thị, tiếp theo là các đỉnh chia cắt theo thứ tự tùy ý. (Nếu không có đỉnh chia cắt thì chỉ ghi một số 0).

§4. ĐỀ THI TIN HỌC QUỐC TẾ LẦN VIII

(Năm 1996)

Bài 1 : SORT 3

SẮP XẾP DÃY VỚI BA GIÁ TRỊ (SORTING A THREE - VALUED SEQUENCE)

Sắp xếp là một trong những công việc tính toán hay phải làm nhất. Xét bài toán sắp xếp cụ thể sau đây : Cần sắp xếp các bản ghi theo giá trị của khóa, trong đó các bản ghi cần sắp xếp có nhiều nhất ba giá trị khóa khác nhau. Chẳng hạn, bài toán như vậy sẽ phải giải quyết khi ta muốn sắp xếp những người được huy chương trong một cuộc thi theo giá trị của huy chương mà họ nhận được, có nghĩa là những người đạt huy chương vàng xếp trước, tiếp theo là huy chương bạc và cuối cùng là những người đạt huy chương đồng.

Trong bài này ba giá trị có thể có của khóa là các số nguyên 1, 2 và 3. Cần tiến hành sắp xếp theo thứ tự không giảm của giá trị khóa. Việc sắp xếp cần được thực hiện bằng một dãy các thao tác đổi chỗ. Một thao tác đổi chỗ được xác định bởi hai số p và q chỉ rõ cần đổi chỗ hai phần tử ở vị trí p và q cho nhau.

Bạn được cho một dãy các giá trị khoá. Hãy viết chương trình tính số ít nhất các thao tác đổi chỗ cần thực hiện để sắp xếp dãy giá trị khóa đã cho thành một dãy không giảm (Câu A). Đồng thời, hãy xây dựng dãy các thao tác đổi chỗ tương ứng với cách sắp xếp tìm được (Câu B).

Dữ liệu vào

Dòng dầu tiên của file INPUT.TXT chứa N là số lượng bản ghi (1 <= N <= 1000). Mỗi một trong số N dòng tiếp theo chứa một giá trị khóa.

Dữ liệu ra

Ghi ra dòng dầu tiên của file OUTPUT.TXT số lượng nhỏ nhất L các thao tác đổi chỗ cần thực hiện để sắp xếp dãy đã cho thành dãy không giảm (Câu A). N dòng tiếp theo ghi dãy các thao tác đổi chỗ tương ứng theo trình tự thực hiện. Mỗi dòng chứa một thao tác đổi chỗ được mô tả bởi hai số p và q là các vị trí của hai phân tử dược đổi chỗ cho nhau (Câu B). Các vị trí dược đánh số từ 1 đến N.

Hình 1 (Figure 1) cho file dữ liệu vào và file kết quả tương ứng với nó.

Bàl 2 : PREFIX

ĐOẠN ĐẦU DÀI NHẤT (LONGEST PREFIX)

Cấu trúc của một số vật thể sinh học được biểu diễn bởi một dãy các thành phần của chung. Các thành phần này được ký hiệu bởi các chữ cái hoa. Các nhà sinh học quan tâm đến việc phân rã một dãy dài thành các dãy ngắn hơn. Các dãy ngắn đó được gọi là các dãy nguyên thủy. Ta nói rằng một dãy S có thể ghép được từ một tập cho trước p các dãy nguyên thủy nếu tìm được n dãy nguyên thủy P1.., Pn thuộc P sao cho ghép của chúng Pi...Pn bằng S. Việc ghép các dãy nguyên thủy Px,...,Pn có nghĩa là đặt chúng liên tiếp liền nhau theo thứ tự đó. Một dãy nguyên thủy có thể có mặt nhiều lần trong phép ghép và không nhất thiết mọi dãy nguyên thủy phải có mặt. Chẳng hạn dãy ABABACABAAB có thể ghép được từ tập các dãy nguyên thủy.

{A, AB, BA, CA, BBC}

K ký tự đầu của một dãy S được gọi là đoạn đầu với độ dài K của S. Viết chương trình với dữ liệu vào là một tập p các dãy nguyên thủy và một dãy T các thành phần. Chương trình cần tính độ dài của đoạn đầu dài nhất của dãy T mà đoạn đầu đó có thể ghép được từ các dãy nguyên thủy trong p.

Dữ liệu vào

Dữ liệu vào được cho trong hai file INPUT.TXT mô tả tập các dãy nguyên thủy p còn file DATA.TXT chứa dãy T cần được xem xét. Dòng thứ nhất của file INPUT.TXT chứa N là số lượng dãy nguyên thủy trong P(1<=N<= 100). Mối dãy nguyên thủy được cho bởi hai dòng liên tiếp. Dòng thứ nhất chứa độ dài L của dãy

nguyên thủy (1 <= L < = 20). Dòng thứ hai chứa một xâu độ dài L gồm các chữ cái hoa (từ ’A’ đến ’Z’). N dãy nguyên thủy này từng đôi một khác nhau.

Mỗi dòng của file DATA.TXT chứa một chữ cái hoa ở vị trí đầu tiên. File này kết thúc bằng dòng chứa một dấu chấm (‘.’) ở vị trí đầu tiên.

Độ dài của dãy từ 1 đến 500000.

Dữ liệu ra

Viết vào dòng thứ nhất của file OUTPUT.TXT độ dài của đoạn đầu dài nhất của T mà đoạn đầu có thể ghép được từ tập p.

- Hình 1 (Figure 1) cho hai file vào và một file ra tương ứng.

Bài 3 : MAGIC

CÁC Ô VUÔNG THẦN BÍ (MAGIC SQUARES)

Kế tục thành công của trò chơi với khối lập phương thần bí, ngài Rubik sáng tạo ra dạng phẳng của trò chơi này gọi là trò chơi các ô vuông thần bí. Đây là một bảng gồm 8 ô vuông bằng nhau (xem hình 1 (Figure 1)).

Trong bài này chúng ta xét bảng trong đó mỗi ô vuông có một màu khác nhau. Các màu được ký hiệu bởi 8 số nguyên dương đầu tiên (xem hình 1). Trạng thái của bảng được cho bởi dãy ký hiệu màu của các ô được viết lần lượt theo chiều kim đồng hồ bắt đầu từ ô ở góc trái trên và kết thúc tại ô ở gốc trái dưới. Ví dụ, trạng thái của bảng trong hình 1 được cho bởi dãy (1, 2, 3, 4, 5, 6, 7, 8). Trạng thái này được gọi 1A trọng thái khởi đầu.

Có thể dùng 3 phép biến đổi cơ bản đối với bảng có tên là ‘A’, ’B’ và ‘C’

‘A’ : đổi chỗ dòng trên và dòng dưới.

‘B’ : thực hiện một phép hoán vị vòng quanh sang phải.

‘C’ : quay theo chiều kim đồng hồ bốn ô giữa.

Biết rằng từ trạng thái khởi đầu luôn có thể chuyển về một trạng thái bất kỳ bằng cách dùng các phép biến đổi cơ hản nói trên.

Tác động của ba phép biến đổi cơ bản được mô tả trong hình 2 (Figure 2), trong đó các số viết bên cạnh bảng dùng để chỉ vị trí các ô vuông của bảng và ô vuông ở vị trí p chứa số i có nghĩa là, sau khi áp dụng phép biến đổi tương ứng, ô vuông mà vị trí trước khi biến đổi của nó là i được chuyển đến vị trí p.

Bạn phải viết chương trình tìm dãy các phép biến đổi cơ bản để chuyển bảng từ trạng thái khởi đầu cho trong hình 1 về một trạng thái đích cho trước (Câu A). Bạn sẽ được thêm hai điểm nếu số lượng phép biến đổi tìm được không vượt quá 300 (Câu B).

Dữ liệu vào

File INPUT.TXT chứa 8 số nguyên dương trong dòng đầu tiên mô tả trạng thái đích.

Dữ liệu ra

Trong dòng đầu tiên của file OUTPUT.TXT chương trình của bạn cần ghi L là số lượng phép biến đổi của dáy các phép biến đổi tìm được. Trong L dòng tiếp theo phải ghi dãy tên các phép biến đổi cơ bản theo trình tự thực hiện, mối tên ghi ở vị trí đầu tiên của mối dòng.

Công cụ trợ giúp

Chương trình MTOOL.EXE ghi trong thư mục bài giúp bạn luyện tập với trò chơi các ô vuông thần bí. Bằng cách thực hiên lệnh "mtool input.txt ouput.txt"

Bạn có thể xem tác động của dãy các phép biến đổi cho trong output.txt đối với bảng ở trạng thái đích cho trong input.txt.

Ví dụ vé dữ liệu vào và ra

Hình 3 (Figure 3)

INPUT.TXT

Bài 4 : JOB PROCESSING

Một nhà máy chạy một dây chuyền sản xuất. Có hai nguyên công cần phải thực hiện đối với mỗi sản phẩm theo trình tự sau : đầu tiên là nguyên công "A", sau đó tới nguyên công "B". Có một số máy để thực hiện từng nguyên công. Hình 1 chỉ ra cách tổ chức dây chuyền sản xuất hoạt động như sau:

Máy kiểu "A" lấy sản phẩm từ băng chuyền vào, thực hiện nguyên công "A" và đặt sản phẩm cho băng chuyền trung gian. Máy kiểu "B" lấy sản phẩm từ băng chuyền trung gian, thực hiện nguyên công "B" và đặt sản phẩm vào băng chuyền ra. Mọi máy đều có thể làm việc song song và độc lập với nhau. Kích thước của các băng chuyền là không giới hạn. Các máy có khả năng làm việc khác nhau, mỗi máy làm việc với thời gian xử lí cho trước. Thời gian xử lí là số đơn vị thời gian cần thiết để thực hiện nguyên công bao gồm cả thời gian lấy sản phẩm từ băng chuyền trước khi xử lí và thời gian đặt sản phẩm vào băng chuyền sau khi xử lí.

Đưa ra thời điểm sớm nhất mà nguyên công "A" được hoàn thành đối với tất cả N sản phẩm với điều kiện là các sản phẩm này đã sẵn sàng trên băng chuyền vào tại thời điểm 0 (Câu A). Đưa ra thời điểm sớm nhất mà cả hai nguyên công "A" và "B" được hoàn thành đối với tất cả N sản phẩm khi các sản phẩm này đã sẵn sàng trên băng chuyền vào tại thời điểm 0 (Câu B).

Dữ liệu vào

File INPUT.TXT gồm các số nguyên dương ghi trong 5 dòng. Dòng thứ nhất chứa N là số sản phẩm (1 < = N < = 1000). Trên dòng thứ hai ghi M1 là số lượng

các máy kiểu "A" (1 <= MI <= 30). Trên dòng thứ ba ghi M1 số nguyên là các thời gian xử lí của từng máy kiểu "A". Các dòng thứ tư và thứ năm tương ứng ghi M2 là số lượng các máy kiểu "B" (1 <= M2 <= 30) và các thời gian xử lí của từng máy kiểu "B". Thời gian xử lí là một số nguyên nằm trong khoảng từ 1 đến 20.

Dữ liệu ra

Chương trình của bạn cần ghi hai dòng ra file OUTPUT.TXT. Dòng đầu tiên chứa một số nguyên dương là lời giải của câu A. Dòng thứ hai chứa lời giải của câu B.

Ví dụ về dữ liệu vào và ra

Hình 2 (Figure 2) cho một file input có thể có và file output tương ứng với nó.

Bài 5: MẠNG CÁC TRƯỜNG HỌC

(NETWORK OF SCHOOLS)

Một số trường học được nối với nhau bởi một mạng máy tính. Có một sự thỏa thuận giữa các trường học này : mỗi trường học có một danh sách các trường học (gọi là danh sách các "trường nhận") và mỗi trường khi nhận được một phần mềm từ một trường khác trong mạng hoặc từ bên ngoài, cần phải chuyển phần mềm nhận được cho các trường trong danh sách các trường nhận của nó. Cần chú ý rằng nếu B thuộc danh sách các trường nhận của trường học A thì A không nhất thiết phải xuất hiện trong danh sách các trường nhận của trường học B.

Người ta muốn gửi một phần mềm đến tất cả các trường học trong mạng. Bạn cần viết chương trình tính số ít nhất các trường học cần gửi bản sao của phần mềm này để cho phần mềm đó có thể chuyển tới tất cả các trường học trong mạng theo thỏa thuận trên (Câu A). Ta muốn chắc chắn rằng khi bản sao phần mềm được gửi đến một trường học bất kì, phần mềm này sẽ được chuyển tới tất cả các trường học trong mạng. Để đạt được mục đích này, ta có thể mở rộng các danh sách các trường nhận bằng cách thêm vào các trường mới. Tính số ít nhất các mở rộng cần thực hiện sao cho khi ta gửi một phần mềm mới đến một trường bất kì trong mạng, phần mềm này sẽ được chuyển đến tất cả trường khác (Câu B). Ta hiểu một mở rộng là việc thêm một trường mới vào trong danh sách các trường nhận của một trường học nào đó.

§5. GỢI Ý LỜI GIẢI

ĐỂ THI TIN HỌC Quốc TẾ LẦN V NĂM 1993

BÀI 1: NECKLACE

— Lần lượt thử mọi vị trí điểm cắt. Với mỗi vị trí này, ta tính số lượng hạt chọn được và lưu trữ lại vị trí tốt nhất.

BÀI 2: COMPANY

1. Tìm tất cả các cặp i và j sao cho hãng i chiếm hơn 50 % cổ phần của hãng j=> Hãng i kiểm soát hãng j.

2. Với mỗi cặp (i, j), ta xét tất cả các hãng C1,C2,...,Ck mà hảng i kiểm soát hãng C1,C2,...Ck và hãng Ct (t = 1, 2, ..., k) có Xt (%) cổ phần của hãng j và X1+X3 + ...+ Xk > 50 (%) Hãng i kiểm soát hãng j.

3. Trở lại bước 2 cho đến khi không thực hiên được nữa.

BÀI 3: RECTANGLE

1. Với mỗi cặp hình chữ nhật i và j (i >= j) nghĩa là hình chữ nhật i được đặt lên tờ giấy sau hình chữ nhật j. Nếu hai hình chữ nhật này có phần chung và chúng khác màu nhau thì phần chung này sẽ mang màu của hình chữ nhật i.

2. Từ nội ô chưa được đánh dấu, ta loang ra các ô cùng màu chưa được đánh dấu xung quanh (tối đa là 8 ô) và cứ thế cho đến khi không thể thực hiện tiếp dược nữa. Như vậy ta được một hình.

3. Cứ tiếp tục thực hiện bước 2 cho đến khi mọi ô đều được xét đến.

4. Lưu ý tờ giấy hình chữ nhật kích thước (a, b) có màu trắng (màu 1)

BÀI 4

— Thủ tục đệ quy Solve (Count, Kind : Byte):

+ Kind 1 nếu trên đường đi từ Tây sang Đông.

Kind 2 nếu trên đường đi từ Đông sang Tây.

+ Count . Số thành phố đã qua.

+ Nếu thời gian thực hiện quá MaxTime (giây) thì xuất kết quả và dừng chương trình.

+ Nếu Count + Số thành phố chưa đến < số lượng nhiều nhất các thành phố trong kết quả trước đó thì không thực hiên.

+ Nếu Kind = 1 thì tìm đường đi từ thành phố hiện giờ đến các thành phố có số thứ tự lớn hơn chưa đến.

+ Nếu Kind = 2 thì tìm đường đi từ thành phố hiên giờ đến các thành phố có số thứ tự nhỏ hơn chưa đến. (Trừ thành phố xuất phát).

ĐỀ THI TIN HỌC QUỐC TẾ LẦN VI NĂM 1994

BÀI 1 : TRIANGLE

1. Bảng tam giác được quy ước trở thành màng A [i, j]

(1. 1)

(2. 1) (2. 2)

(3. 1) (3. 2) (3. 3)

(N . 1) (N . 2) (N . N)

2. Gọi S [i, j] là tổng lớn nhất các số trên đường đi từ (1. 1) đến (i, j) S [i. j] = Max {S [i-1, j], s [i-1. j 1]}

BÀI 2 . CASTLE

1. Cách tìm các bức tường : xét modul (i, j) có giá trị p

+ (p shr 0) and 1 = 1 thì có tường phía Tây

+ (p shr 1) and 1=1 thì có tường phía Bắc

+ (p shr 2) and 1 = 1 thì oó tường phía Đông

+ (p shr 3) and 1 = 1 thì có tường phía Nam

2. Từ một modul chim được đánh dấu, ta loang ra các modul xung quanh nếu như chứng không có tường ở giữa nhau (Tối đa 4 modul) và cứ thế cho đến khi không thể thực hiện tiếp được nữa. Như vậy ta được một phòng.

3. Cứ tiếp tục thực hiện bước 2 cho đến khi mọi modul đều được xét.

4. Với mỗi modul có giá trị p > 0 (có tường), ta thử mọi khả năng phá các bức tường này và tính tổng diện tích 2 phòng rồi so sánh với kết quả tốt nhất đã có được.

BÀI 3 : PRIME

1. Xây dựng các số nguyên tố có 5 chữ số và có tổng các chữ số bằng giá trị trong dữ liệu.

2. Lần lượt điền các số nguyên tố vào dòng đầu tiên, cột đầu tiên, sau đó đến các dòng, các cột khác.

+ Chỉ cần điền vào 4 cột, 4 hàng vì sau đó ta dễ dàng xác định các chữ số còn lại của bảng.

+ Số được điền vào cột i phải không nhỏ hơn số điền vào hàng i

+ Nếu tìm được 1 nghiệm, ta tìm được một nghiệm thứ 2 bằng cách

A11	A12	A13	A14	A15
A21	A22	A23	A24	A26
A31	A32	A33	A34	A35
A41	A42	A43	A44	A45
A51	A52	A53	A54	A55

A11	A21	A31	A41	A51
A12	A22	A32	A42	A52
A13	A23	A33	A43	A53
A14	A24	A34	A44	A54
A15	A25	A35	A45	A55

ĐỀ THI TIN HỌC Quốc TẾ LẦN VI NĂM 1994

BÀI 4 : CLOCK

— Với mỗi phép biến đổi ta lần lượt thử sử dụng 0, 1, 2, 3 lần.

— Với mỗi bộ phép biến đổi ta thử xây dựng mảng B [1.. 3, 1..3]

— Nếu sau khi áp dụng ta thu được mảng có các giá trị bằng 0 và tổng số phép biến đổi ít hơn kết quả trước đó thì lưu kết quả này.

BÀI 5 : BUS LINES

— Xét các xe đến trong khoảng thời điểm từ 0 đến 29m ta thiết lập mọi tuyến xe có thể có và phù hợp với bản ghi nhận trong đề bài.

— Tìm cách loại bỏ càng nhiều tuyến xe sao cho số tuyến xe còn lại vẫn đảm bảo thỏa mãn với bản ghi nhận.

BÀI 6 : SECTOR

— Nếu M < K thì bài toán vô nghiệm.

— Nếu K < M thì I >= M+N— 1 và ta có thể điền vào các hình quạt liên tiếp các số lần lượt từ M đến M+N— 1.

— Nhận xét I =< 50 .

— Gọi A là mảng các số cần điền.

— A [1] lần lượt lấy các giá trị K, K + 1, ... M

— Với mỗi giá trị A [1], ta tìm mọi cách điền vào mảng A và tìm giá trị I tương ứng. Sau đó so sánh với kết quả tối ưu tìm được trước đó.

ĐỀ THI TIN HỌC QUỐC TẾ LẦN VII NĂM 1995

BÀI 1 : PACK

— Ta thử mọi cách xếp 4 hình chữ nhật. (6 cách)

— Với mỗi cách xếp, ta xét tất cả các cách hoán vị của 4 hình chữ nhật (4! = 24) và các khả năng hình chữ nhật quay 90 độ.

— Vậy có tối đa 6X4!X24=2304 trường hợp.

BÀI 2 SHOP

— Gọi A[CS1,Cs2>Cs3>Cg4,CS5] là chi phí nhỏ nhất phải trả khi mua CSi đơn vị hàng i.

— Với mỗi S= A[CS1,CS2,CS3,CS4,CS5] ta thực hiện:

+ For 1 := 1 to 5 do Gán D [j] := 0

+ For j := 1 to cs [i] do

D [i] := j

Nếu A [cs [1] - d [1], cs [2] - d [2], cs [3] - d [3], cs [4] - d [4], cs [5] — d [5]] + Gt [i] * j < S

thì gán cho S giá trị này.

For i := 1 to M do

+ Nếu có thể sử dụng cách mua đặc biệt i và số hàng mỗi loại trong cách mua i đều không vượt quá các Csi thì tính lại giá chi phí theo cách mua dặc biệt này và so sánh với s.

BÀI 3 : LGAME

— Tìm mọi tập con của các kí tự đưa vào. Nếu điểm thu được khi tạo ra các từ với các kí tự thuộc tập con này lớn hơn hay bằng điểm cao nhất để có thì ta tìm mọi cách sắp xếp chúng để tạo lập các từ. Kiểm tra tính đúng đắn của từ (hay cặp từ) tìm được và so sánh kết quả thu được.

BẢI 5 : RACE

— Tìm đường đi ngắn nhất từ 0 đến N.

— Với mỗi đỉnh thuộc đường đi này, ta thử loại bỏ đi và kiểm tra xem có còn đường đi nào khác từ 0 đến N hay không. Nếu không thì đỉnh đó là đỉnh không tránh được.

— Tập hợp các đỉnh chia cắt là tập con của tập các đỉnh không tránh được. Kiểm tra tất cả các đỉnh không tránh được, ta sẽ thu được kết quả.

ĐỀ THI TIN HỌC Quốc TẾ LẦN VIII NĂM 1996

BÀI 1 : SORT 3

— Giả sử đã sắp xếp xong, gọi Pi là vị trí đầu tiên của số i (1 < i < 3)

— Tìm tất cả các số i, j sao cho j ở vị trí >= Pi và =< Pi+1 và i ở vị trí >= Pj và =< Pj+1. Ta hoán vị hai số i và j.

— Tìm tất cả các cặp i, j ở sai vị trí còn lại và hoán vị chúng.

BÀI 2 PREFIX

— Gọi lp là độ dài dài nhất của Pi

— Xét từng đoạn của dãy s có độ dài lp

— Với đoạn thứ nhất, ta tìm mọi cách sử-dụng Pi để tạo ra phần đầu của đoạn này (gọi tập hợp các dãy này là A1)

— Đến đoạn thứ k (k > 1) của s, ta tìm mọi cách điển P1 vào các dãy thuộc Ak-1 thu được trước đó. Nếu dãy A thuộc Ak -1 với mọi cách điền Pi đều có độ dài nhỏ hơn (k-1) * lp thi loại bỏ A đi. Các dãy còn lại (sau khi điền Px nào đó) sẽ là các phân tử thuộc tập Ak.

— Đến khi không còn một cách điền các Pi vào các dãy thuộc An để tạo dãy dài hơn thì dãy A thuộc An có độ dài dài nhất sẽ là kết quả.

BÀI 3 : MAGIC

— Có tất cả 8!. = 40320 dạng khác nhau của bảng gồm 8 ô vuông như trong đề bài.

— Từ dạng khởi đầu, ta loang ra 3 dạng khác (theo 3 phép biến đổi A, B, C) và đánh dấu các dạng này.

— Từ một dạng đã được đánh dấu, ta loại loang ra các dạng khác (nếu chúng chưa được đánh dấu), cho đến khi dạng kết quả được đánh dấu.

BÀI 4 JOB PROCESSING

Câu A :

— Gọi D là số sản phẩm mà các máy A tạo ra trong 1 đơn vị thời gian.

— Thời gian ít nhất để làm xong là số injin sao cho i * D >= N

Câu B :

— Tương tự câu A nhưng thời điểm tốt đầu làm của mỗi sản phẩm (sau khi được hoàn thành công việc ở máy A) không giống nhau và khác 0.

BÀI 5: NETWORK OF SCHOOLS

— Thiết lập ma trận A : A [i, j] = 1 nếu có đường đi từ i tới j (không cần trực tiếp).

Câu A :

— Số lượng các đỉnh j thỏa Max {A [i, j] | i khác j} = 0 chính là đáp số cần tỉm.

Câu B :

— Số lượng các đỉnh j thỏa Max {A [i, j] | i khác j} = 0 chính là đáp số cần tìm.

BÀI 6 : GAME

— Gọi A1,A2,...An là các số đã cho theo thứ tự.

— Gọi B1=A1+A3+…+AN-1 và B2=A2 + A4+ ...+AN

— Giả sử B1 > B2 thì ta tìm cách để luôn chọn các số Ai có chỉ số lẹ như sau : Giả sử ta đã chọn một số Ai với i lẻ thì bắt buộc máy phải chọn Aj với j chẵn. Cứ thế cho đến khi ta chọn đủ N/2 số có chỉ số lẻ.