# Ràng buộc Tên file: rangbuoc.cpp

Cho một danh sách các biến với các ràng buộc thứ tự dạng *"x <y"*, bạn phải viết một chương trình in ra tất cả các dãy thứ tự của các đại lượng mà vẫn thỏa mãn các ràng buộc.

Ví dụ, với các ràng buộc *"x < y"* và *"x < z"* thì có hai dãy thứ tự của các biến *x, y, z* phù hợp với các ràng buộc này là: "*xyz"* và *"xzy"*. Có nghĩa là biến đứng trước trong dãy nhỏ hơn biến đứng sau và vẫn thỏa mãn ràng buộc đã cho với các biến khác.

**Input: rangbuoc.inp**

* Đầu vào gồm một số bộ dữ liệu, mỗi bộ dữ liệu gồm hai dòng:
* Dòng đầu chứa một danh sách các biến.
* Dòng sau chứa lần lượt một số cặp phần tử *x, y* cho biết có ràng buộc *x < y*.

**Giới hạn**

* Mỗi bộ dữ liệu có ít nhất hai biến và không quá 10 biến.
* Mỗi bộ dữ liệu có ít nhất một ràng buộc và không quá 50 ràng buộc.
* Có từ 1 đến 300 dãy thứ tự phù hợp với các ràng buộc trong một bộ dữ liệu.

**Output: rangbuoc.out**

* Ứng với mỗi bộ dữ liệu, in ra tất cả các dãy thứ tự phù hợp với các ràng buộc đã cho. Các dãy thứ tự được in theo thứ tự từ điển (theo thứ tự chữ cái), một dãy thứ tự trên mỗi dòng.
* Giữa các bộ dữ liệu liên tiếp có một dòng trống phân cách.

Ví dụ

|  |  |
| --- | --- |
| Rangbuoc.inp | Rangbuoc.out |
| a b f g  a b b f  v w x y z  v y x v z v w v | abfg  abgf  agbf  gabf  wxzvy  wzxvy  xwzvy  xzwvy  zwxvy  zxwvy |

**Giải thích ví dụ:** Bộ dữ liệu đầu tiên có 4 biến *a, b, f, g* với 2 cặp quan hệ *a < b* và *b < f* và ta có 4 dãy thứ tự thỏa mãn: *(a, b, f, g); (a, b, g, f); (a, g, b, f)* và *(g, a, b, f)*.

# Biểu thức tên file: expression.cpp

Nhiệm vụ của bạn là viết một chương trình có thể kiểm tra xem có thể tìm thấy một biểu thức số học bao gồm năm số cho trước sẽ mang lại giá trị 23 hay không?

Trong bài toán này, chúng ta sẽ chỉ xem xét các biểu thức số học sau đây:

Trong đó: và đôi một khác nhau, và ? thuộc tập hợp {+, - , \*}

**INPUT: EXPRESSION.INP**

- Gồm nhiều dòng, mỗi dòng chứa 5 số nguyên dương không vượt quá 20

- Kết thúc input là 5 số 0.

**OUTPUT: EXPRESSION.OUT**

- Hiện ra thông báo “Possible” nếu việc kiểm tra cho kết quả đúng và ngược lại hiện “Impossible”

Ví dụ:

|  |  |
| --- | --- |
| EXPRESSION.INP | EXPRESSION.OUT |
| 1 1 1 1 1  1 2 3 4 5  2 3 5 7 11  0 0 0 0 0 | Impossible  Possible  Possible |

# Đếm vùng Tên file: DEMVUNG.CPP

Cho ma trận kích thước các giá trị của ma trận là số nguyên dương hãy đếm các vùng là số nguyên tố của ma trận *(các số nguyên tố được coi là cùng một vùng nếu chúng nằm cạnh nhau theo hướng ngang hoặc dọc)*?

**Dữ liệu vào file DEMVUNG.INP**

* dòng đầu là 2 số và cách nhau một dấu cách
* dòng tiếp theo là các giá trị của ma trận.

**Kết quả ra file DEMVUNG.OUT**

* Số vùng nguyên tố của ma trận trên.

Ví dụ:

|  |  |
| --- | --- |
| DEMVUNG.INP | DEMVUNG.OUT |
| 5 5  2 3 4 5 3  5 6 8 7 2  3 7 6 5 3  3 4 5 7 4  2 7 6 9 3 | 3 |

# Trò chơi trên ma trận Tên file: GAMES.CPP

Ngày 1 – 6, mẹ dẫn hai anh em Hiếu Nghĩa vào siêu thị mua đồ chơi. Nhân dịp ngày tết thiếu nhi, siêu thị mở đợt khuyến mại đặc biệt cho người vào mua hàng. Để nhận được một phần quà khuyến mại của siêu thị khách mua hàng phải thắng trong một trò chơi, được gọi là “trò chơi trên ma trận số”.

Trò chơi như sau:

Trên ma trận N x N (3 <= N <= 10), mỗi ô ghi một số nguyên có giá trị tuyệt đối <= 103. Người chơi phải chọn ra trên mỗi hàng, mỗi cột chỉ có 1 số được chọn, đồng thời tổng các số được chọn là lớn nhất.

Nếu ai chọn được thỏa mãn yêu cầu trên sẽ được một món bộ đồ chơi “Siêu nhân” của siêu thị.

Anh em Hiếu Nghĩa rất thích bộ đồ chơi đó, vì vậy nhờ bạn lập trình để giúp cho họ dành chiến thắng.

Input: đọc từ file **GAMES.INP**:

* Dòng 1 chứa số N
* N dòng tiếp theo, mỗi dòng chứa N số nguyên cách nhau 1 dấu cách.

Output: Ghi ra file **GAMES.OUT**:

* Dòng đầu là giá trị tổng lớn nhất
* N dòng tiếp theo: dòng thứ i ghi 1 số nguyên là chỉ số cột của cần chọn của dòng i.

Ví dụ:

|  |  |
| --- | --- |
| GAMES.INP | GAMES.OUT |
| 3  7 9 6  3 5 8  8 6 9 | 25 |

Giải thích: Các ô (1,2) (2,3) và (3,1) sẽ được chọn

# Cung điện Tên file: CUNGDIEN.CPP

Ở vương quốc nọ có một vị vua và N quí phi. Trên miếng đất hình vuông kích thước N x N, nhà vua muốn xây cho các quí phi, mỗi người một cung điện (giả sử mỗi cung điện đều nằm trên một ô đất kích thước 1 x 1). Vấn đề là các quí phi đều rất ghen ghét nhau nên nhà vua không muốn các cung điện nhìn thấy nhau từ các hướng (ngang, dọc, chéo). Chi phí xây dựng các cung điện trên mỗi ô đất có thể có các giá trị khác nhau, nhà vua muốn xây dựng N cung điện với chi phí thấp nhất.

**Yêu cầu**: Bạn hãy giúp nhà vua thực hiện công việc đó.

Dữ liệu vào: Từ file CUNGDIEN.INP gồm:

* Dòng đầu chứa số
* dòng tiếp theo, mỗi dòng chứa N số là chi phí xây dựng của ô tương ứng (chi phí xây dựng cung điện trong ô có giá trị từ 1 đến 1000). Mỗi số cách nhau 1 khoảng trắng.

Dữ liệu ra: Ghi ra file CUNGDIEN.OUT: Dòng 1 chứa chi phí nhỏ nhất xây dựng N cung điện

|  |  |
| --- | --- |
| CUNGDIEN.INP | CUNGDIEN.OUT |
| 4  3 4 12 3  6 1 7 1  2 4 1 5  12 3 8 7 | 15 |

**Ví dụ**

# Đổ nước Tên file: WATER.CPP

Cho một thùng dung tích có thể xem là vô hạn và bình có dung tích lít. Liệu có thể dùng (mỗi bình chỉ dùng 1 lần) bình nước này để đổ vào thùng đựng đúng lít nước không? Khi dùng mỗi bình để đổ nước vào thùng, bình đó phải chức đầy nước.

Dữ liệu vào được cho bởi file **water.inp**:

* Dòng thứ nhất ghi số nguyên dương và số nguyên dương .
* Trong dòng tiếp theo, ghi các số nguyên dương , mỗi số cách nhau một dấu cách.

Kết quả ghi ra file **water.out** như sau:

* Nếu không được, ghi ra -1
* Nếu có thể được, ghi dòng chữ số K, là số lượng cách đổ nước vào, các dòng tiếp theo ghi thứ tự các bình nước.

VD:

|  |  |
| --- | --- |
| Nuoc.inp | Nuoc.out |
| 10 50  24 25 16 28 21 27 13 19 12 40 | 2  2 7 9  3 5 7 |

# Những bức ảnh Tên file: images.cpp

Một bức ảnh đen trắng kích thước gồm điểm ảnh, các điểm ảnh được mô tả như các số 0 hoặc 1. Người ta muốn biết có bao nhiêu vùng trắng rời nhau trong bức ảnh. Một vùng trắng là một miền liên thông gồm các điểm ảnh biểu diễn bằng các chữ số 1 kề cạnh hoặc chung đỉnh trong bức ảnh.

Em hãy lập trình đếm số vùng trắng trong một bức ảnh cho trước.

*Dữ liệu*: vào từ file **images.inp:** Gồm một số test mẫu, mỗi test mẫu gồm:

* Dòng đầu chứa số nguyên dương , mô tả kích thước bức ảnh
* dòng sau, mỗi dòng chứa một chuỗi nhị phân có độ dài , mô tả bức ảnh.
* là dấu hiệu kết thúc input

*Kết quả*: ghi ra file **imagest.out**: Ứng với mỗi test mẫu, in ra một dòng gồm hai số nguyên S và C, trong đó S là số thứ tự của test mẫu trong tệp dữ liệu, C là số vùng trắng đếm được trong bức ảnh tương ứng. Các số in cách nhau một dấu cách.

*Ví dụ*:

|  |  |
| --- | --- |
| **imagest.inp** | **imagest.out** |
| 6  100100  001010  000000  110000  111000  010100  8  01100101  01000001  00011000  00000010  11000011  10100010  10000001  01100000  0 | 1 3  2 6 |

# Vòng số nguyên tố Tên file: PRIMERING.CPP

Một vòng tròn lớn gồm các ***n*** vòng tròn nhỏ (***n*** là số chẵn) như hình dưới.

A picture containing sketch, circle, line art, white

Description automatically generated

Hãy điền các số tự nhiên 1, 2, 3, …, ***n*** vào các vòng tròn nhỏ sao cho tổng hai số tự nhiên ở hai vòng tròn liên tiếp là một số nguyên tố.

Biết rằng: Vòng tròn đầu tiên luôn điền số 1.

INPUT: PRIMERING.INP

* Gồm nhiều dòng, mỗi dòng chứa một số nguyên dương

OUTPUT: PRIMERING.OUT

* Đưa ra tất cả các cách sắp xếp các số tự nhiên từ 1 đến n vào các vòng tròn nhỏ.
* Các cách xếp được hiện theo thứ tự từ điển.
* Các test cách nhau bởi một dấu xuống dòng.

Ví dụ:

|  |  |
| --- | --- |
| PRIMERING.INP | PRIMERING.OUT |
| 6  8 | 1 4 3 2 5 6  1 6 5 2 3 4  1 2 3 8 5 6 7 4  1 2 5 8 3 4 7 6  1 4 7 6 5 8 3 2  1 6 7 4 3 8 5 2 |

# Quét vôi Tên file: QV.CPP

Một quần thể nhà cao tầng được xây dựng trên một nền hình chữ nhật, trên đó được chia thành *M*×*N* ô vuông ðõn vị (*M* dòng, *N* cột). Các dòng được đánh số từ 1 đến *M*, các cột được đánh số từ 1 đến *N*. Người ta xem khu nhà được tạo bởi các khối có đáy là một ô vuông với những chiều cao nào đó mà người ta gọi là những đơn nguyên. Một đơn nguyên được xác định bởi tọa độ dòng, cột của ô đáy và chiều cao tương ứng. Một khối nhà được định nghĩa là một tập hợp các đơn nguyên có đáy tạo thành một miền gồm các ô kề cạnh. Ví dụ hình vẽ dưới đây mô tả một quần thể gồm 3 khối nhà: Người ta đánh số các khối nhà bằng những số nguyên liên tục bắt đầu từ 1 theo trình tự duyệt các ô đáy theo từng dòng từ 1 đến *M* và trên mỗi dòng, duyệt các ô đáy theo từng cột từ 1 đến *N*. Ví dụ, các khối nhà cho trong hình vẽ bên dýới được đánh số theo thứ tự các ô đáy (có màu xám, con số là chiều cao). Người ta muốn quét vôi các bức tường xung quanh tất cả các khối nhà (không quét vôi phía trên nóc nhà).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 0 | 2 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 2 | 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 |

**Yêu cầu:** Hãy xác định số lượng các khối nhà, khối nhà có diện tích quét vôi lớn nhất và diện tích này.

**Dữ liệu:** Vào từ file QV.INP:

- Dòng đầu tiên ghi 2 số nguyên *M*, *N* (0 < *M*, *N* ≤100)

- *M* dòng tiếp theo, mỗi dòng ghi *N* số nguyên thể hiện chiều cao của đơn nguyên (0 ≤ *H*[*i*, *j*] ≤ 10.000 với *i* = 1, 2, …, *M*; *j* = 1, 2, …, *N*), với quy ước bằng 0 khi đơn nguyên này không có. Giả thiết rằng các giá trị này đều là các số nguyên và tính theo đơn vị một cạnh của ô vuông. Các số trên cùng môt dòng ghi cách nhau ít nhất một dấu cách.

**Kết quả :** Ghi ra file QV.OUT :

- Dòng đầu tiên ghi số lượng các khối nhà ;

- Dòng thứ ba ghi số hiệu của khối nhà có diện tích quét vôi lớn nhất và giá trị diện tích của khối nhà này. Nếu có nhiều khối nhà có cùng diện tích quét vôi lớn nhất thì đưa ra khối nhà có số hiệu nhỏ nhất.

**Ví dụ:**

|  |  |
| --- | --- |
| QV.INP | QV.OUT |
| 4 6  1 2 3 0 2 1  1 0 1 0 0 1  2 1 1 0 0 1  0 0 0 1 1 0 | 3  1 30 |

# Sói và cừu Tên file: SOICUU.CPP

Có một số con cừu trong trại chăn nuôi của Mickey. Trong khi Mickey đang ngủ say, những con chó sói đã vào trại và tấn công đàn cừu. Trại có dạng hình chữ nhật gồm các ô tổ chức thành hàng và cột. Kí tự dấu ‘.’ là ô rỗng, kí tự ‘#’ là hàng rào, kí tự ‘o’ là cừu và kí tự ‘v’ là chó sói. Chúng ta coi 2 ô là cùng một miền nếu có thể chuyển ô nọ tời ô kia bằng đường đi chỉ gồm các đường đi theo chiều ngang hoặc thẳng đứng không vướng hàng rào. Các ô mà từ chúng có thể thoát ra khỏi sân được xem là một phần của một miền nào đó.

May thay, những con cừu biết tự vệ. Chúng có thể chiến đầu với những con sói trong miền (húc chết sói) nếu số lượng cừu lớn hơn số lượng sói trong cùng một miền. Ngược lại, những con sói sẽ ăn hết các con cừu trong cùng một miền.

Ban đầu các con cừu và các con sói đã được xác định trong các miền của trại.

Viết một chương trình tính số lượng cừu và số lượng sói còn lại trong sáng hôm đó.

**Dữ liệu vào:** *Lấy từ file văn bản SOICUU.INP:*

* Dòng đầu tiên chứa hai số nguyên M và N (M,N <= 250) là số hàng và số cột của trại.
* Mỗi dòng trong M dòng sau gồm N kí tự. Tất cả các kí này biểu diễn các vị trí có hàng rào, cừu và chó sói trong trại.

**Kết quả:** *ghi ra file văn bản SOICUU.OUT:*

* Chỉ một dòng gồm 2 con số: số cừu và số sói còn lại trong trại.

|  |  |
| --- | --- |
| SOICUU.INP | SOICUU.INP |
| 6 6   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **.** | **.** | **.** | **#** | **.** | **.** | | **.** | **#** | **#** | **v** | **#** | **.** | | **#** | **v** | **.** | **#** | **.** | **#** | | **#** | **.** | **o** | **#** | **.** | **#** | | **.** | **#** | **#** | **#** | **.** | **#** | | **.** | **.** | **.** | **#** | **#** | **#** | | 8 8   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | **.** | **#** | **#** | **#** | **#** | **#** | **#** | **.** | | **#** | **.** | **.** | **o** | **.** | **.** | **.** | **#** | | **#** | **.** | **#** | **#** | **#** | **#** | **.** | **#** | | **#** | **.** | **#** | **v** | **.** | **#** | **.** | **#** | | **#** | **.** | **#** | **.** | **o** | **#** | **o** | **#** | | **#** | **o** | **.** | **#** | **#** | **.** | **.** | **#** | | **#** | **.** | **v** | **.** | **.** | **v** | **.** | **#** | | **.** | **#** | **#** | **#** | **#** | **#** | **#** | **.** | |
| SOICUU.OUT | SOICUU.OUT |
| 0 2 | 3 1 |

Ví dụ:

# Cái tui Tên file: CAITUI.CPP

Cây khế nhà Khánh rất sai quả nên có một con chim to đến ăn. Ăn xong, chim chở Khánh ra đảo để trả công bằng vàng. Đảo có cục vàng. Anh ấy muốn chuyển hết cả N cục vàng của mình về nhà. Nhưng khổ nổi các cục vàng này lại có trọng lượng và kích thước khổng lồ. Khánh đem theo một cái túi ba trăm gang to đùng nhưng vẫn chưa chắc chứa hết đống vàng này. Khổ quá đi! Lấy cục nào, bỏ cục nào bây giờ! Các bạn giúp anh ấy tìm ra một cách chọn vàng để thu được giá trị lớn nhất mà vẫn không làm rách túi.

Input: Từ file CAITUI.INP:

* Dòng 1: Chứa 2 số nguyên: số cục vàng và tải trọng tối đa của túi .
* dòng sau: Mỗi dòng chứa 2 số nguyên: trọng lượng và giá trị  của cục vàng thứ .

Output: ghi ra file CAITUI.OUT

* Một số nguyên duy nhất là giá trị lớn nhất thu được.

Ví dụ

|  |  |
| --- | --- |
| CAITUI.INP | CAITUI.OUT |
| 3 4 1 4 2 5 3 6 | 10 |

# Ngoặc Tên file: NGOAC.INP

Có thể định nghĩa khái niệm dãy ngoặc đúng theo dạng đệ quy như sau:

* là một dãy ngoặc đúng.
* là dãy ngoặc đúng nếu hay với là các dãy ngoặc đúng.

Ví dụ dãy ngoặc đúng là:

Ví dụ dãy ngoặc sai là:

Hãy liệt kê tất cả các dãy ngoặc đúng có độ dài ( chẵn).

Dữ liệu vào từ file NGOAC.INP gồm 1 số nguyên dương chẵn và

Kết quả ghi ra file NGOAC.OUT:

* Gồm nhiều dòng, mỗi dòng là một dãy ngoặc đúng.
* Dòng cuối cùng là số – số lượng dãy ngoặc đúng.

Ví dụ:

|  |  |
| --- | --- |
| NGOAC.INP | NGOAC.OUT |
| 4 | (())  ()()  2 |