|  |  |
| --- | --- |
| **Summary** | |
| Program | fgift.\* |
| Input | fgift.inp |
| Output | fgift.out |
| Time/test | 1 giây |

# TẶNG HOA

Nhân dịp ngày Quốc tế phụ nữ (8-3), các bạn nam trong lớp quyết định mua hoa tặng các bạn nữ trong lớp mình. Tuy nhiên, đây là một kế hoạch tự phát, mỗi bạn nam tự mình đi mua hoa không bàn bạc với bạn khác. Chính vì vậy cuối cùng có M loại hoa khác nhau được đem đến lớp (các loại hoa đánh số từ 1 đến M), loại hoa thứ i có *ai* bông hoa.

Một vấn đề đau đầu được đặt ra cho lớp trưởng - một bạn nam đẹp trai nhất trong lớp- là làm thế nào chia các bông hoa này cho các bạn nữ trong lớp để **số bông hoa của bạn nữ nhận được nhiều hoa nhất là nhỏ nhất**. Biết rằng mỗi bạn nữ chỉ nhận các bông hoa cùng một loại (hoặc không nhận được bông hoa nào).

Viết chương trình tính số lượng hoa của bạn nữ nhận được nhiều hoa nhất trong phương án trên.

**Input:**

* Dòng đầu tiên ghi hai số nguyên dương N () là số lượng bạn nữ trong lớp và M () là số lượng loại hoa khác nhau
* M dòng tiếp theo, dòng thứ *i* ghi số *ai* là số lượng hoa của loại hoa thứ *i* ()

**Output:** Một số nguyên duy nhất là số bông hoa của bạn nữ nhận được nhiều hoa nhất trong phương án tối ưu (là phương án mà số hoa của bạn nữ có nhiều hoa nhất là nhỏ nhất)

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5 2  7  4 | 3 |  |
| 7 5  7  1  7  4  4 | 4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | familiar.\* |
| Input | familiar.inp |
| Output | familiar.out |
| Time/test | 1 giây |

# GƯƠNG MẶT THÂN QUEN

"*Gương mặt thân quen"* là một chương trình giải trí khá nổi tiếng trên VTV3. Trong chương trình này, mỗi thí sinh sẽ bắt chước giọng hát của một ca sỹ nổi tiếng nào đó và trên cơ sở đó bạn giám khảo sẽ cho điểm từng thí sinh.

Có tất cả N thí sinh tham gia thi. Cuộc thi được diễn ra trong nhiều vòng thi khác nhau. Mỗi vòng thi, thí sinh tốt nhất sẽ được N điểm, thí sinh tốt thứ nhì được N-1 điểm, thí sinh tốt thứ ba được N-2 điểm, ..., thí sinh tốt thứ N được 1 điểm. Điểm của mỗi vòng thi của từng thí sinh được cộng lại, sau vòng thi cuối cùng thí sinh nào được nhiều điểm nhất sẽ giành chức vô địch. Tất nhiên, nếu có nhiều thí sinh cùng đạt nhiều điểm nhất thì tất cả họ đều giành chức vô địch.

Chỉ còn một vòng thi nữa là cuộc thi kết thúc. Hiện tại điểm tổng của các thí sinh là . Hỏi rằng có bao nhiêu thí sinh có quyền hy vọng rằng mình sẽ đạt chức vô địch sau vòng thi cuối cùng?

**Input:**

* Dòng đầu tiên chứa số nguyên N () là số lượng thí sinh tham gia thi
* N dòng tiếp theo, mỗi dòng ghi một số nguyên *ai* () là số điểm của các thí sinh trước vòng thi cuối cùng

**Output:** Một số nguyên duy nhất là số lượng thí sinh có thể đạt được chức vô địch sau vòng thi cuối cùng

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5  15  14  15  12  14 | 4 |  |
| 3  8  10  9 | 3 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | strenght.\* |
| Input | strenght.inp |
| Output | strenght.out |
| Time/test | 1 giây |

# TỔNG DÃY SỐ

Với số nguyên dương *x* ta định nghĩa *f*(*x*) là số nguyên dương nhỏ nhất mà *x* không chia hết (ví dụ f(6)=4).

Với mỗi số nguyên dương *n* ta xây dựng dãy:



Ví dụ với *n*=6 ta có:



Số *k* được định nghĩa là **strenght(n)**. Chẳng hạn như trên thì strenght(6)=4.

Cho hai số nguyên dương **A<B** tính tổng:

strenght(A)+strengh(A+1)+...+strength(B)

**Input:** Một dòng duy nhất chứa hai số nguyên dương A và B ()

**Output:** Chứa một số nguyên duy nhất là tổng tìm được

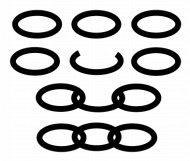
**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3 6 | 11 |  |
| 100 200 | 262 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | chairs.\* |
| Input | chairs.inp |
| Output | chairs.out |
| Time/test | 1 giây |

# CHUỖI VÒNG

Alice tìm thấy N chuỗi trên gác nhà mình. Mỗi chuỗi bao gồm một dãy các vòng tròn nối với nhau.



Alice muốn nối các chuỗi trên lại thành một chuỗi dài bằng cách mở một số vòng tròn trên các chuối sau đó luồn nó nối với vòng tròn khác và hàn lại. Hỏi rằng Alice cần mở ít nhất bao nhiêu vòng tròn để có thể làm được điều này?.

**Input:**

* Dòng đầu tiên ghi N là số lượng chuỗi tìm được ()
* Dòng thứ hai chứa N số nguyên dương Li () là độ dài (số lượng vòng tròn) của các chuỗi.

**Output:** Một số nguyên duy nhất là số lượng vòng tròn phải mở

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 2  3 3 | 1 |  |
| 3  1 1 1 | 1 |  |
| 5  4 3 5 7 9 | 3 |  |

Giải thích:

* Ở ví dụ đầu tiên Alice mở vòng cuối cùng của chuỗi 1 và nối nó với vòng đầu tiên của chuỗi hai
* Ở ví dụ thứ hai: Alice mở vòng của chuỗi 1, dùng nó nối hai vòng của chuỗi 2 và 3
* Ở ví dụ thứ ba: Alice mở cả 3 vòng của chuỗi độ dài 3. Dùng 3 vòng này để nối các chuỗi 4, 5, 7, 9 lại với nhau (4-5, 5-7 và 7-9)

|  |  |
| --- | --- |
| **Summary** | |
| Program | callfood.\* |
| Input | callfood.inp |
| Output | callfood.out |
| Time/test | 1 giây |

# GỌI MÓN ĂN

Alice đang rất đói và cô quyết định ăn trưa tại một nhà hàng gần cơ quan. Nhà hàng có N món ăn khác nhau và thật đặc biệt: món ăn thứ i có hai giá Ai và Bi (i=1,2,...,N) trong đó khách hàng phải trả Ai nếu i là món ăn đầu tiên được gọi trong bữa, các trường hợp còn lại món i có giá Bi.

Vì rất đói nên Alice không thể quyết định là chọn những món nào để ăn. Cô ta quyết định hỏi bạn (là chủ nhà hàng) rằng nếu ăn đúng k món (1≤k≤N) thì phải trả số tiền ít nhất là bao nhiêu?

**Input:**

* Dòng đầu tiên ghi số nguyên dương N () là số lượng các món ăn
* N dòng tiếp theo, dòng thứ i chứa hai số nguyên dương Ai và Bi () là giá của món thứ i theo mô tả ở trên

**Output:** Ghi N dòng, dòng thứ k ghi số tiền tối thiểu phải trả khi ăn đúng k món ăn trong số N món ăn của nhà hàng

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3  10 5  9 3  10 5 | 9  13  18 |  |
| 2  100 1  1 100 | 1  2 |  |
| 5  1000000000 1000000000  1000000000 1000000000  1000000000 1000000000  1000000000 1000000000  1000000000 1000000000 | 1000000000  2000000000  3000000000  4000000000  5000000000 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | restore.\* |
| Input | restore.inp |
| Output | restore.out |
| Time/test | 1 giây |

# KHÔI PHỤC HOÁN VỊ

Ngồi buồn chán vì công việc quá nhàn rỗi, Bờm lấy ra một mảnh giấy và viết lên đó dãy số A gồm N số là các số nguyên dương nằm trong phạm vi từ 1 đến N mỗi số xuất hiện đúng một lần. Sau đó anh ta lấy ra một mảnh giấy khác và viết lên đó các nhận xét về dãy số đã viết trên mảnh giấy trước.

Tất cả các nhận xét đều có 1 trong 2 dạng sau:

* 1 x y v: Giá trị lớn nhất của các số nằm từ vị trí x đến vị trí y là v
* 2 x y v: Giá trị nhỏ nhất của các số nằm từ vị trí x đến vị trí y là v

Phú Ông thấy Bờm viết một dãy số dài trên tờ thứ nhất lại tưởng đó là mã khóa của kho báu nên nổi lòng tham và lấy đi.

Bờm tự hỏi, nếu chỉ dựa trên các nhận xét ở tờ giấy thứ hai có thể đưa ra một dãy số gồm N số trên tờ thứ nhất (gồm các số từ 1 đến N mỗi số xuất hiện 1 lần) được không? Tất nhiên, không nhất thiết dãy số này phải giống như dãy số trên tờ giấy mà Phú Ông đã lấy đi.

**Input:**

* Dòng đầu tiên ghi hai số nguyên N () là độ dài của dãy số trên tờ thứ nhất và M () là số lượng nhận xét
* M dòng tiếp theo, mỗi dòng mô tả một nhận xét theo qui cách trên

**Output:** Một dòng chứa N số nguyên là dãy số tìm được. Nếu có nhiều dãy thỏa mãn thì chỉ cần in ra một dãy. Nếu không tồn tại dãy thỏa mãn nhận xét thì in ra -1

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3 2  1 1 1 1  2 2 2 2 | 1 2 3 |  |
| 4 2  1 1 1 1  2 3 4 1 | -1 |  |
| 5 2  1 2 3 3  2 4 5 4 | 1 2 3 4 5 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | diffuse.\* |
| Input | diffuse.inp |
| Output | diffuse.out |
| Time/test | 1 giây |

# TRUYỀN TIN

Có một con đường thẳng nối hai làng. Trên con đường này có N người truyền tin đứng tại N vị trí. Khi truyền tin họ có thể chạy lại gần nhau đồng thời hét to tin mà họ muốn truyền cho người kia.

Người truyền tin đầu tiên (luôn ở làng 1) có một máy thu radio và anh ta dùng để theo dõi tin tức trên cả nước. Khi nhận được một tin thú vị và quan trọng, anh ta bắt đầu chạy và hét to tin tức mình nhận được cho những người tiếp sau, những người này lại làm thế cho đến khi tất cả những người truyền tin đều biết thông tin. Có một vài quy tắc cần chú ý như sau:

* Tất cả những người truyền tin đều có thể chạy về hai phía với vận tốc 1 đơn vị/giây. Họ cũng có thể quyết định không chạy
* Mọi người đều nhận được toàn bộ thông tin cần tuyền tại 1 thời điểm. Ngoài ra một người có thể nghe thấy một người khác nếu như khoảng cách giữa họ không vượt quá K đơn vị dộ dài.

Viết chương trình cho biết vị trí của tất cả những người truyền tin. Hãy xác định thời gian nhỏ nhất để tất cả mọi người đều nhận được tin từ người 1.

**Input:**

* Dòng đầu tiên ghi số thực K (0≤K≤106) là khoảng cách lớn nhất có thể nghe thấy nhau.
* Dòng thứ hai ghi số nguyên N là số người truyền tin (1≤N≤105)
* N dòng tiếp theo, mỗi dòng ghi một số thực D (0≤D≤109) là khoảng cách của người truyền tin đến vị trí của làng thứ nhất. Các giá trị này được liệt kê tăng dần. Có thể có nhiều người ở cùng một vị trí.

**Output:**

Một số thực duy nhất là thời gian nhỏ nhất để N người nhận được thông tin. Kết quả có sai số 0.001

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 2.000  4  0.000  4.000  4.000  8.000 | 1.000 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | squirr.\* |
| Input | squirr.inp |
| Output | squirr.out |
| Time/test | 1 giây |

# SÓC VÀ HẠT DẺ

Có *n*chú sóc đứng chời dưới gốc của *m* cây hạt dẻ. Cây hạt dẻ thứ *i* sẽ bắt đầu rụng quả đầu tiên sau *Ti* giây nữa và cứ sau *Pi* giây nó lại rụng thêm một quả. Sóc mẹ muốn các cậu con trai của mình đem về tổ không ít hơn *k* quả hạt dẻ để chuẩn bị tránh cơn sóng thần dữ dội đến từ biển Đông xa xôi, nhưng cũng phải thật nhanh chóng trước khi cơn sóng ập đển chứ chẳng thể nhởn nhơ! Vậy nên các chú sóc đang bàn nhau xem đứng ở những gốc cây nào để có thể hững đủ số quả cần thiết trong thời gian nhanh nhất. Thời gian để chú sóc đi về vị trí cần đứng không đáng kể, các chú sóc cũng không di chuyển sang gốc cây nào khác trong lúc hứng hạt dẻ.

**Yêu cầu:** Hãy tìm thời gian sớm nhất (sau bao nhiêu giây nữa) các chú sóc có thể hứng đủ số quả cần thiết.

**Input:**

* Dòng đầu tiên chứa các số nguyên *m*, *n*, *k* ()
* Dòng thứ hai chứa lần lượt các số nguyên 
* Dòng thứ ba chứa lần lượt các số nguyên 

**Output:** Ghi ra một số nguyên duy nhất là thời gian sớm nhất tìm được

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3 2 5  5 1 2  1 2 1 | 4  (hai chú sóc đứng ở cây 2 và 3 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | coconut.\* |
| Input | coconut.inp |
| Output | coconut.out |
| Time/test | 1 giây |

# KHỈ HÁI DỪA

Có một khu vườn rộng nằm ngay bên cạnh một sở thú, ở đó có rất nhiều dừa.

Sở thú đã huấn luyện được 2 loại khỉ đặc biệt: Loại thứ nhất có khả năng hái dừa và loại thứ hai có khả năng bổ những quả dừa để chế biến thành sản phẩm (nước, cùi,...).

Có N con khỉ thuộc loại thứ nhất, con khỉ thứ k của loại này sẽ mất Ak thời gian để leo lên cây và bắt đầu hái quả dừa đầu tiên. Cứ sau mỗi Bk giây nó hái được một quả.Có M con khỉ thuộc loại thứ hai, con khỉ thứ k của loại này sẽ mất Ck thời gian để tìm công cụ và bắt đầu chế biến quả dừa đầu tiên, cứ sau Dk giây nó chế biến xong một quả dừa.

Loại khỉ thứ hai là loại khỉ rất hung hãn nên cúng sẽ đuổi đánh loại khỉ thứ nhất nếu như cùng ở trong vườn tại một thời điểm. Chính vì vậy, đầu tiên người ta cho các con khỉ loại thứ nhất vào hái dừa. Sau khi hái hết những quả dừa, người ta mới đưa con khỉ loại thứ hai vào để chế biến. Thời gian đưa các con khỉ vào/ra khỏi vườn xem như bằng 0.

Chúng ta không biết trong vườn có bao nhiêu quả dừa. Tuy nhiên biết rằng T giây là thời gian kể từ khi đưa những con khỉ thứ nhất vào vườn đến khi số dừa được chế biến hoàn toàn.

Hãy xác định sau thời gian bao nhiêu lâu, kể từ khi đưa những con khỉ thứ nhất vào hái dừa đến khi những con khỉ thứ hai được đưa vào vườn.

**Input:**

* Dòng đầu tiên ghi T là tổng thời gian hai loại khỉ ở trong vườn (1≤T≤109)
* Dòng thứ hai ghi số nguyên dương N (1≤N≤100)
* N dòng tiếp theo, mỗi dòng ghi hai số Ak, Bk (1≤Ak,Bk≤109)
* Dòng tiếp theo ghi số nguyên dương M (1≤M≤100)
* M dòng cuối, mỗi dòng ghi hai số Ck, Dk( 1≤Ck,Dk≤109)

**Output:**Một số nguyên là khoảng thời gian từ khi đưa con khỉ thứ nhất đến vườn đến khi đưa con khỉ thứ hai đến vườn

**Example:**

|  |  |  |
| --- | --- | --- |
|  |  | Giải thích: |
| 12  1  3 1  1  5 1 | 5 | Giải thích: Trong vườn có 3 quả dừa   * Ở giây thứ 3 bắt đầu hái quả dừa đầu tiên * Ở giây 4, 5 hái các quả dừa thứ 2 và thứ 3 * Sau 5 giây đưa loại khỉ thứ hai vào vườn * Ở giây thứ 10 bắt đầu chế biến quả dừa đầu tiên * Ở giây 11, 12 bắt đầu chế biến quả dừa thứ 2 và thứ 4 * Sau 12 giây đưa con khỉ thứ hai ra khỏi vườn. |

|  |  |
| --- | --- |
| **Summary** | |
| Program | wlan.\* |
| Input | wlan.inp |
| Output | wlan.out |
| Time/test | 1 giây |

# PHỦ SÓNG

Để mở rộng địa bàn phục vụ, Công ty dịch vụ mạng không dây WLAN quyết định đặt K trung tâm phủ sóng để phục vụ các địa điểm dân cư dọc theo một đường xa lộ. Các khảo sát địa bàn của Công ty cho thấy có N địa điểm dân cư cần được phục vụ (được đánh số từ 1 đến N) dọc theo xa lộ này. Công suất của thiết bị đặt tại mỗi trung tâm là như nhau và được đặc trưng bởi bán kính phủ R. Nghĩa là, khi đưa vào hoạt động, thiết bị đặt tại trung tâm sẽ phục vụ được tất cả các điểm dân cư nằm trong phạm vi bán kính R. Giá thành của thiết bị càng tăng nếu bán kính phủ sóng của nó càng lớn.

Thông tin đầu vào được mô tả trên mặt phẳng toạ độ Đềcác như sau. Xa lộ được xác định bởi đường thẳng đi qua hai điểm A(xA, yA), B(xB, yB) cho trước, và giả thiết là không bị giới hạn về cả hai phía. Điểm dân cư thứ i được cho bởi toạ độ (xi, yi), i = 1, 2, ..., N. Mỗi trung tâm phủ sóng phải được đặt tại một điểm nằm trên đường thẳng đi qua A và B.

Yêu cầu: Hãy xác định số R nhỏ nhất sao cho có thể tìm được cách đặt K trung tâm phủ sóng dọc theo xa lộ sao cho mỗi một trong số N địa điểm dân cư đều được phủ sóng bởi ít nhất một trong K trung tâm này.

**Input:**

* Dòng đầu tiên chứa số nguyên *N* (1 ≤ *N* ≤ 10000) là số địa điểm dân cư và số nguyên *K*
* (1 ≤ *K* ≤ *N*) là số trung tâm phủ sóng.
* Dòng thứ hai chứa các số nguyên *xA, yA*, *xB, yB*.
* Dòng thứ *i* trong số *N* dòng tiếp theo chứa hai số nguyên *xi, yi*, *i* = 1, 2, ..., *N*.

Giả thiết rằng các toạ độ là các số nguyên có trị tuyệt đối không vượt quá 105. Hai số liên tiếp trên cùng một dòng được ghi cách nhau bởi 1 dấu cách.

**Output:**Số thực *R* là công suất tìm được với 4 chữ số sau dấu phảy.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output | Minh họa |
| 2 2  0 5 5 0  2 5  4 3 | 1.4142 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | oaktree.\* |
| Input | oaktree.inp |
| Output | oaktree.out |
| Time/test | 1 giây |

# LỐI ĐI BỘ CHO KHÁCH DU LỊCH

Chính phủ có kế hoạch xây dựng một lối đi bộ dành cho khách du lịch ở giữa một khu rừng sồi. Khu rừng có thể được mô tả như là mặt phẳng có N điểm đặc biệt mô tả các các cây sồi.

Lối đi cho khách du lịch là một đường vòng có dạng hình chữ nhật có các cạnh song song với các trục tọa độ. Nếu như cạnh của hình chữ nhật này đi qua cây sồi nào thì cây sồi đó phải được đốn bỏ. Các cây sồi không nằm trên các cạnh của hình chữ nhật thì không cần phải đốn hạ.

Một dạnh sách P lối đi bộ cho khách du lịch được đệ trình lên chính phủ. Ljuko, một quan chức cao cấp của Bộ Lâm nghiệp - người rất yêu cây cối quan tâm đến việc với kế hoạch xây dựng được đệ trình thì có bao nhiêu cây sồi bị đốn hạ trong quá trình xây dựng các lối đi bộ.

Viết chương trình xác định với mỗi lối đi bộ được xây dựng thì có bao nhiêu cây sồi bị đốn hạ.

**Input:**

* Dòng đầu tiên ghi số N là số lượng cây sồi (N≤300000)
* N dòng tiếp theo, dòng thứ i ghi hai số nguyên Xi, Yi là tọa độ của một cây sồi (1 ≤ Xi, Yi≤109)
* Dòng kế tiếp ghi P là số lối đi bộ trên kế hoạch (P≤100000)
* P dòng tiếp theo, mỗi dòng ghi 4 số nguyên X1, Y1, X2, Y2 với ý nghĩa (X1,Y1) là tọa độ của góc trái dưới và (X2, Y2) là tọa độ của góc phải trên của 1 lối đi bộ lần lượt được xây dựng (1≤X1≤X2≤109, 1≤Y1≤Y2≤109)

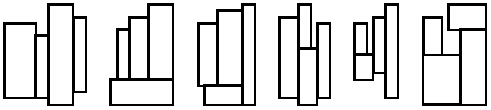
**Output:** Gồm P dòng, dòng thứ i ghi số lượng cây sồi bị đốn hạ khi xây dựng lối đi bộ thứ i (theo trình tự trong file input)

*Example:*

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6  1 2  3 2  2 3  2 5  4 4  6 3  4  2 2 4 4  2 2 6 5  3 3 5 6  5 1 6 6 | 3  4  0  1 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | packrec.\* |
| Input | packrec.inp |
| Output | packrec.out |
| Time/test | 1 giây |

# ĐÓNG GÓI HÌNH CHŨ NHẬT

  
Sáu cách sắp xếp cơ bản của 4 hình chữ nhật

Cho 4 hình chữ nhật. Tìm một hình chữ nhật bao (mới) nhỏ nhất mà ta có thể đặt 4 hình chữ nhật cho trước vào mà không có hình nào chồng lên hình nào. Hình chữ nhật nhỏ nhất là hình có diện tích nhỏ nhất.

Bốn hình chữ nhật phải có các cạnh song song với các cạnh của hình chữ nhật bao ngoài. Hình 1 cho thấy 6 cách đặt 4 hình chữ nhật với nhau. Sáu cách này là các cách sắp xếp cơ bản duy nhất, bởi vì các cách sắp xếp khác có thể tạo ra từ một cách sắp xếp cơ bản bằng cách xoay hoặc lấy đối xứng. Các hình chữ nhật có thể xoay 90 độ trong lúc đóng gói.

Có thể tồn tại nhiều hình chữ nhật bao khác nhau thỏa mãn các yêu cầu, với cùng diện tích. Bạn phải xuất ra tất cả các hình chữ nhật bao đó.

**Input:**Gồm Bốn dòng, mỗi dòng chứa 2 số nguyên cách nhau bởi khoảng trắng cho biết độ dài 2 cạnh của một hình chữ nhật. Mỗi cạnh của một hình chữ nhật nhỏ nhất là 1 và lớn nhất là 50.

**Output:** Chứa số dòng nhiều hơn số đáp án là 1. Dòng đầu tiên chứa 1 số nguyên: diện tích nhỏ nhất của các hình chữ nhật bao. Mỗi dòng tiếp theo chứa 1 đáp án được mô tả bởi hai số p và q với p<=q. Các dòng này phải được sắp xếp theo chiều tăng của p, và tất cả phải khác nhau.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 1 2  2 3  3 4  4 5 | 40  4 10  5 8 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | party.\* |
| Input | party.inp |
| Output | party.out |
| Time/test | 1 giây |

# DỰ TIỆC BÀN TRÒN

Có n nhà khoa học đánh số 1, 2, ..., n và 26 lĩnh vực khoa học ký hiệu A, B, C, ..., Z. Thông tin về người thứ i được cho bởi một xâu ký tự Si gồm các chữ cái in hoa thể hiện những lĩnh vực khoa học mà người đó biết.

*Ví dụ: S2 = 'ABCXYZ' cho biết nhà khoa học thứ 2 có hiểu biết về các lĩnh vực A, B, C, X, Y, Z.*

Một lần cả n nhà khoa học đến dự một bữa tiệc. Chủ nhân của bữa tiệc định xếp n nhà khoa học ngồi quanh một bàn tròn, nhưng một vấn đề khiến chủ nhân rất khó xử là các nhà khoa học của chúng ta có hiểu biết xã hội tương đối kém, nên nếu như phải ngồi cạnh một ai đó không hiểu biết gì về các lĩnh vực của mình thì rất khó nói chuyện.

Vậy hãy giúp chủ nhân xếp n nhà khoa học ngồi quanh bàn tròn sao cho hai người bất kỳ ngồi cạnh nhau phải có ít nhất một lĩnh vực hiểu biết chung, để các nhà khoa học của chúng ta không những ăn ngon mà còn có thể trò chuyện rôm rả.

**Dữ liệu:**

* Dòng 1: Ghi số n
* n dòng tiếp theo, dòng thứ i ghi xâu ký tự Si

**Kết quả:**Gồm n dòng. Dòng thứ i ghi nhà khoa học ngồi tại vị trí i của bàn (Các vị trí trên bàn tròn được đánh số từ 1 đến n theo chiều kim đồng hồ)

**Lưu ý:**

* n ≤ 20
* Nếu có nhiều cách xếp thì chỉ cần chỉ ra một cách
* Nếu không có cách xếp thì ghi vào file PARTY.OUT một dòng: NO SOLUTION

**Ví dụ:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | Output |  | Input | Output |  | Input | Output |
| 6  AV  DIQR  DV  CQ  AC  DR | 1  3  6  2  4  5 |  | 10  AX  BI  ABTX  AS  IK  KS  BE  AB  EK  AK | 1  3  2  5  6  4  8  7  9  10 |  | 6  AB  BC  CD DE  EF  FG | NO SOLUTION |

|  |  |
| --- | --- |
| **Summary** | |
| Program | poklon.\* |
| Input | poklon.inp |
| Output | poklon.out |
| Time/test | 1 giây |

# QUÀ CỦA BÀ NGOẠI

Bà ngoại có hai đứa cháu mà bà rất rất thích tặng quà cho. Một hôm, bà mua N món quà tặng có giá khác nhau và để cho hai đứa cháu tự chia nhau. Bắt đầu từ đứa lớn hơn, hai cháu luân phiên chọn một món quà. Do tính tham lam của trẻ con nên chúng luôn chọn món quà đắt nhất trong số các món quà còn lại.

Bà ngoại biết rằng đứa cháu lớn sẽ không hài lòng nếu tổng giá trị quà của nó so với đứa em ít hơn A. Còn đứa em sẽ không hài lòng nếu như tổng giá trị quà của đứa lớn hơn nó B. Nói cách khác hiệu giữa tổng giá trị quà của đứa lớn và đứa bé phải nằm giữa A và B thì cả hai mới hài lòng.

Giá của mỗi món quà là số nguyên dương. Giá này được in trên mỗi món quà trừ ra một món quà không in giá. Bà ngoại quyết định sẽ in 1 giá cho món quà này sao cho sau khi hai đứa cháu chọn quà với giá bà in lên chúng đều càm thấy hài lòng. Tất nhiên, giá mà bà in ra cũng phải là số nguyên dương

Viết chương trình xác định xem có bao nhiêu cách khác nhau để bà ngoại chọn giá cho món quà còn lại.

**Input:**

* Dòng đầu tiên ghi ba số nguyên dương A, B và N (1≤A≤B≤109, 1≤N≤105)
* N-1 dòng tiếp theo, mỗi dòng ghi giá của một món quà. Nó là số nguyên dương nhỏ hơn 109. Giá được liệt kê theo giá trị giảm dần. Tất nhiên giá món quà mà bà in ra không có ở đây.

**Output:**Một số nguyên duy nhất là số lượng các giá khác nhau mà bà có thể chọn.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6 15 7  17  13  10  10  6  2 | 23 |  |

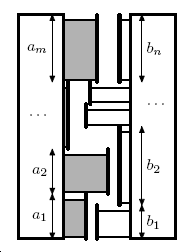
Ghi chú: Có 80% số test có giá nhỏ hơn 10000, trong số này có 50% số test có N nhỏ hơn 100.

|  |  |
| --- | --- |
| **Summary** | |
| Program | shelves.\* |
| Input | shelves.inp |
| Output | shelves.out |
| Time/test | 1 giây |

# TỦ TÀI LIỆU

Chi nhánh ngân hàng thành phố mua 2 tủ chống cháy lưu thông tin của khách hàng. Mỗi tủ có một số lượng ngăn kéo khác nhau với mỗi ngăn có độ cao khác nhau. Tủ thứ nhất có ***m*** ngăn tính từ dưới lên có độ cao là ***a***1, ***a***2, . . ., ***am***, tủ thứ 2 có ***n*** ngăn kéo, tính từ dưới lên có độ cao là ***b***1, ***b***2, . . ., ***bn***.

Các tủ được đặt quay mặt vào nhau trong một hành lang hẹp, vì vậy không thể mở đồng thời các ngăn kéo đối diện nhau. Để tiện làm việc, các nhân viên muốn mở đồng thời càng nhiều ngăn kéo càng tốt và giữ chúng ở trạng thái mở cả ngày.

***Yêu cầu***: Cho ***m***, ***n***, ***ai***, ***bji*** =1 ÷***m***. ***j*** =1 ÷ ***n*** (1 ≤ ***m***, ***n*** ≤ 100 000, 1 ≤ ***ai***, ***bj*** ≤ 109). Hãy xác định số năng kéo nhiều nhất có thể mở đồng thời và chỉ ra các ngăn kéo có thể để mở.

***Dữ liệu***:

* Dòng đầu tiên chứa 2 số nguyên ***m*** và ***n***,
* Dòng thứ 2 chứa ***m*** số nguyên ***a***1, ***a***2, . . ., ***am***,
* Dòng thứ 3 chứa ***n*** số nguyên ***b***1, ***b***2, . . ., ***bn***.

***Kết quả***:

* Dòng thứ nhất đưa ra 2 số nguyên ***k*** và ***l*** – số ngăn kéo để mở được ở tủ thứ nhất và thứ 2,
* Dòng thứ 2 chứa ***k*** số nguyên: các ngăn kéo tủ thứ I có thể để mở,
* Dòng thứ 3 chứa ***l*** số nguyên: các ngăn kéo tủ thứ II có thể để mở.

***Ví dụ***:

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5 5  1 2 3 4 5  6 4 3 2 1 | 3 4  1 2 3  2 3 4 5 |  |

Ghi chú: Có 60% số test có M, N ≤100

|  |  |
| --- | --- |
| **Summary** | |
| Program | visit.\* |
| Input | visit.inp |
| Output | visit.out |
| Time/test | 1 giây |

# DUYỆT ĐIỂM

Xét lưới ô vuông tạo thành từ n\*n đường, các đường của lưới được đánh số từ 1 đến n từ trái qua phải và từ trên xuống dưới (1≤n≤20000). Ở mỗi hàng thứ i, người ta cho đoạn thẳng xác định bởi hai điểm li và ri (1≤li≤ri≤n, i=1…n).

**Yêu cầu:** Xác định độ dài của đường đi ngắn nhất dọc theo các cạnh của lưới từ điểm (1,1) đến điểm (n,n) và thoả mãn các điều kiện:

* Chỉ đi sang phải, sang trái hoặc xuống dưới
* Đi qua tất cả các điểm thuộc các đoạn thẳng đã cho

**Input:**

* Dòng đầu tiên chứa số nguyên dương n
* Dòng thứ i trong n dòng sau chứa hai số nguyên dương li, ri

**Output:**Một số nguyên duy nhất là độ dài của đường đi ngắn nhất tìm được.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6  2 6  3 4  1 3  1 2  3 6  4 5 | 24 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | coins.\* |
| Input | coins.inp |
| Output | coins.out |
| Time/test | 1 giây |

# BỘ SƯU TẬP TIỀN KIM LOẠI

Một bạn học sinh rất yêu thích việc sưu tập các đồng tiền kim loại của các nước. Trong một lần kiểm kê, bạn đã sắp xếp n đống tiền (hình tròn) bộ sưu tập của mình lên mặt bàn quanh một cái vỉ có dạng tròn để thu được hình ảnh của một bông cúc. Các đồng tiền tạo thành một chuỗi khép kín quanh cái vỉ sao cho mỗi đồng tiền tiếp xúc với đúng hai đồng tiền bên cạnh nó như trong hình vẽ chỉ ra dưới đây:

C¸i vØ

Các đồng tiền theo thứ tự ngược chiều kim đồng hồ có bán kính lần lượt là r1, r1, .. rn. Bạn học sinh biết bán kính của các đồng tiền nhưng không biết bán kính của cái vỉ. Hãy giúp bạn học sinh tính bán kính của cái vỉ.

**Input:**

* Dòng đầu tiên chứa số nguyên N (3≤N≤100)
* Dòng thứ hai chứa n số thực r1, r2, .. rn. là bán kính của các đồng tiền được xác định chính xác đến hai chữ số sau phần thập phân.

**Output:** Bán kính của cái vỉ được tính chính xác đến hai số sau phần thập phân.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4  2 2 2 2 | 0.83 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | vitri.\* |
| Input | vitri.inp |
| Output | vitri.out |
| Time/test | 1 giây |

# TÌM VỊ TRÍ

Một đại lý kinh doanh xăng dầu có n trạm bán xăng dầu (gọi tắt là cây xăng) đánh số từ 1 đến n trên một đường cao tốc muốn tìm vị trí đặt k bể chứa xăng để cung cấp xăng cho các cây xăng. Trên đường cao tốc người ta đặt các cột mốc cây số, bắt đầu từ cột mốc số 0. Biết vị trí của cây xăng thứ i là ở cột cây số di (i=1,2,…,n), d1<d2<…<dn.

**Yêu cầu:** Tìm vị trí đặt k bể chứa xăng tại k trong số n cây xăng sao cho khoảng cách lớn nhất từ cây xăng không có bể chứa xăng đến cây xăng có bể chứa xăng gần nó nhất là nhỏ nhất.

**Input:**

* Dòng đầu tiên ghi hai số nguyên dương n, k (n≤10000, k≤n)
* Dòng thứ hai ghi hai ghi các số d1, d2, …, dn (di là các số nguyên dương không quá 2.109). Các số nguyên trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

**­­Output:**Gồm k dòng, mỗi dòng ghi chỉ số của cây xăng đặt bể chứa xăng.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6 3  5 6 12 19 20 27 | 2  4  6 |  |

Ghi chú:

* Trong cách đặt bể xăng như trên khoảng cách xa nhất từ cây xăng không có bể chứa xăng đến cây xăng có bể chứa xăng gần nó nhất là 6
* Có 50% số test có N ≤ 10, 30% số test có 10< n≤200, k≤30

|  |  |
| --- | --- |
| **Summary** | |
| Program | truck.\* |
| Input | truck.inp |
| Output | truck.out |
| Time/test | 1 giây |

# VẬN CHUYỂN HÀNG

Công ty tin học XYZ quyết định thuê K xe chở hàng có tải trọng như nhau để vận chuyển các container hàng. Có *n* kiện hàng được bốc dỡ **lần lượt** theo thứ tự từ 1 đến *n*, kiện hàng thứ *i* có trọng lượng *ai*. Hỏi rằng công ty cần thuê loại xe có tải trọng nhỏ nhất là bao nhiêu để cho thể vận chuyển hết số hàng?.

**Input:**

* Dòng đầu tiên ghi hai số nguyên *n* m K (1≤*n*≤105, 1≤K≤*n*)
* Các dòng tiếp theo lần lượt ghi các số nguyên dương *a1, a2, ..., an* (*ai*≤109) Hai số liên tiếp trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

**Output:**

Một số nguyên duy nhất là tải trọng tối thiểu của K xe tải.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5 2  3 2 4 5 1 | 9 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | dqueue.\* |
| Input | dqueue.inp |
| Output | dqueue.out |
| Time/test | 1 giây |

# HÀNG ĐỢI HAI ĐẦU CƠ BẢN

**(Bài tập lý thuyết)**

Cho dãy số nguyên a1, a2, ..., an. Một truy vấn (Question) trên dãy con này là một lệnh có dạng **Q(i,j)** với ý nghĩa là tìm giá trị nhỏ nhất của các phần tử trong dãy con

*ai, ai+1, ..., aj* (i≤j).

Cho *m* truy vấn Q(i1, j1), Q(i2, j2), ...., Q(im, jm) thỏa mãn:

1. i1 ≤ i2 ≤ ... ≤ im-1 ≤im
2. j1 ≤ j2 ≤ ... ≤jm-1 ≤ jm

Hãy in ra các giá trị là câu trả lời cho các truy vấn tương ứng.

**Input:**

* Dòng đầu tiên ghi hai số nguyên dương n, m (1≤n, m ≤100000)
* Dòng thứ hai ghi n số nguyên a1, a2, ..., an
* m dòng tiếp theo, dòng thứ k ghi hai số ik, jk thể hiện cho truy vấn thứ k (dữ liệu đảm bảo thỏa mãn điều kiện 1 và 2 ở trên)

**Output:** Gồm m dòng, dòng thứ k ghi kết quả của truy vấn thứ k

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5 5  2 3 1 4 5  1 3  2 3  3 4  3 5  4 5 | 1  1  1  1  4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | stack.\* |
| Input | stack.inp |
| Output | stack.out |
| Time/test | 1 giây |

# NGĂN XẾP CƠ BẢN

**(Bài tập lý thuyết)**

Cho dãy *n* số nguyên *a1, a2, ..., an*. Người ta định nghĩa:

* **previous(k):** Là vị trị **i** nhỏ nhất thỏa mãn ***i≤k, aj ≥ ak*** với mọi ***j=i,i+1,...,k***
* **next(k):** Là vị trí***i*** lớn nhất thỏa mãn ***i ≥k, aj ≥ ak*** với mọi ***j=k,k+1,...,i***

Hãy xác định các giá trị trên với mọi *k=1,2,..., n*.

Ví dụ, nếu A=(1, 3, 2, 4, 6, 7, 3, 8, 9) thì

Previous(A) = (1, 2, 2, 4, 5, 6, 4, 8, 9)

Next(A) = (9, 2, 9, 6, 6, 6, 9, 9, 9)

**Input:**

* Dòng đầu tiên ghi *n* (n≤500000)
* Tiếp theo là một số dòng ghi dãy a1, a2, ..., an. Hai số liên tiếp trên một dòng cách nhau ít nhất bởi dấu cách.

**Output:**

Gồm 2 dòng, dòng 1 ghi n số của previous và dòng 2 ghi n số của next

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 9  1 3 2 4 6 7 3 8 9 | 1 2 2 4 5 6 4 8 9  9 2 9 6 6 6 9 9 9 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | io.\* |
| Input | io.inp |
| Output | io.out |
| Time/test | 1 giây |

# HÀNG ĐỢI ƯU TIÊN MAX

**(Bài tập lý thuyết)**

Cho trước một danh sách rỗng. Người ta xét hai thao tác trên danh sách đó:

* Thao tác **"+V"** (ở đây V là một số tự nhiên ≤ 109): Nếu danh sách đang có ít hơn 15000 phần tử thì thao tác này bổ sung thêm phần tử V vào danh sách; Nếu không, thao tác này không có hiệu lực.
* Thao tác **"-"**: Nếu danh sách đang không rỗng thì thao tác này loại bỏ tất cả các phần tử lớn nhất của danh sách; Nếu không, thao tác này không có hiệu lực

*Ví dụ: Với danh sách ban đầu là rỗng:*

* *Nếu ta thực hiện liên tiếp các thao tác: +1, +3, +2, +3 ta sẽ được danh sách (1, 3, 2, 3)*
* *Thực hiện thao tác -, ta sẽ được danh sách (1, 2)*
* *Thực hiện hai thao tác +4, ta sẽ được danh sách (1, 2, 4, 4)*
* *Thực hiện thao tác -, ta sẽ được danh sách (1, 2)*
* *Tiếp tục với các thao tác +2, +9, +7, +8, ta sẽ được danh sách (1, 2, 2, 9, 7, 8)*
* *Cuối cùng thực hiện thao tác -, ta còn lại danh sách (1, 2, 2, 7, 8)*

*Vấn đề đặt ra là cho trước một dãy không quá 100000 thao tác, hãy xác định những giá trị số nào còn lại trong danh sách, mỗi giá trị chỉ được liệt kê một lần.*

**Input:**

Gồm nhiều dòng, mỗi dòng ghi một thao tác. Thứ tự các thao tác trên các dòng được liệt kê theo đúng thứ tự sẽ thực hiện.

**Output:**

* Dòng 1: Ghi số lượng những giá trị còn lại trong danh sách.
* Dòng 2: Liệt kê những giá trị đó theo thứ tự giảm dần

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| +1  +3  +2  +3  -  +4  +4  -  +2  +9  +7  +8  - | 4  8 7 2 1 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | foot.\* |
| Input | foot.inp |
| Output | foot.out |
| Time/test | 1 giây |

# ĐỦ CHẤT

Cũng như mọi sinh viên, Steve cố gắng đảm bảo ăn uống điều độ, đủ chất và tiết kiệm. Đã mấy năm rồi, sáng nào Steve cũng ăn hai cái bánh mỳ tròn và uống một cốc sữa đậu nành.

Sữa đậu nành đóng hộp có thể giữ khá lâu, nhưng bánh mỳ thì không để dành được quá ***k*** ngày. Giá bánh mỳ thường xuyên biến động. Nhờ tính tình vui vẻ cởi mở, Steve có quan hệ rất tốt với người bán hàng và biết được giá bánh trong ***m*** ngày tính từ hôm nay. Từ đó Steve có thể lên kế hoạch để tiết kiệm nhất trong việc mua bánh mỳ.

Ví dụ, bánh có thể giữ được trong hai ngày. Giá bánh hôm này là 3 đồng/chiếc, giá ngày mai là 1 đồng/chiếc và giá ngày kia sẽ là 2 đồng/ chiếc. Kế hoạch chi tiết kiệm của Steve se là: hôm nay mua hai chiếc bánh mỳ tròn, ngày mai – sẽ mua 4 chiếc vừa ăn vừa để dành cho ngày kia. Như vậy Steve phải chi tất cả là 3×2+2×4 = 10.

***Yêu cầu***: Cho ***m***, ***k*** và ***ci***, ***i*** =1 ÷ ***m***, trong đó ***ci*** – giá một chiếc bánh mỳ tròn bán ngày thứ ***i*** (1 ≤ ***m***, ***k***, ***ci*** ≤ 105). Hãy xác định số tiền tối thiểu cần có và số lượng bánh phải mua ở mỗi ngày.

**Input:**

* Dòng thứ nhất chứa 2 số nguyên ***m*** và ***k***,
* Dòng thứ 2 chứa ***m*** số nguyên ***c1***, ***c2***, . . ., ***cn***.

**Output:**Một số nguyên – chi phí tối thiểu,

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3 2  3 1 2 | 10 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | frogs.\* |
| Input | frogs.inp |
| Output | frogs.out |
| Time/test | 1 giây |

# ẾCH ỘT BIẾN GEN

Cuộc sống an nhàn với thức ăn đầy đủ và đa dạng tại các đống rác thành phố đã sinh ra thế hệ các chú ếch đột biến gen. Trên con đường dẫn đến bải rác thành phố có ***n*** đống rác, đánh só bắt đầu từ 0 đến ***n***-1 từ trái qua phải. Đống rác thứ ***i*** có độ cao ***hi*** (***i*** = 0 ÷ ***n***-1, 0 <***hi*** ≤ 109, 0 <***n*** ≤ 106, ***hi*** – nguyên). Trên mỗi đống rác hiện có một chú ếch sống. Đến tuổi trưởng thành, mỗi chú ếch đều muốn đi tìm một chổ sống tốt đẹp hơn bằng cách nhảy sang đống rác cao hơn gần nhất bên phải. Chú ếch ở đống rác thứ ***i*** có thể thực hiện đươc ***Ji*** bước nhảy (0 <***Ji***<***n***). Bãi rác thành phố có độ cao lớn hơn mọi đống rác trên đường. Ta ký hiệu độ cao này là -1 (vì không cần và cũng không thể biết chính xác).

Ví dụ, có 8 đống rác với độ cao tương ứng từ trái sang phải là 3, 1, 4, 5, 6, 2, 3 và 8. Số bước nhảy mỗi chú ếch có thể thực hiện là 1, 2, 1, 3, 4, 2, 1, 2. Sau khi di chuyển hết khả năng của mình, chú ếch ở đống rác 0 sẽ tới được đống rác 2 với độ cao là 4, còn chú ếch ở đống rác 3 – tới được bải rác thành phố (độ cao -1).

***Yêu cầu***: Hãy xác định độ cao nơi ở mới của mỗi chú ếch.

**Input:**

* Dòng đầu tiên chứa số nguyên ***n***,
* Dòng thứ 2 chứa ***n*** số nguyên ***h***0, ***h***1, …, ***hn***-1,
* Dòng thứ 3 chứa ***n*** số nguyên ***J***0, ***J***1, …, ***Jn***-1.

**Output:**Một dòng chứa ***n*** số nguyên – độ cao nơi ở mới của mỗi chú ếch.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 8  3 1 4 5 6 2 3 8  1 2 1 3 4 2 1 2 | 4 5 5 -1 -1 8 8 -1 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | tardy.\* |
| Input | tardy.inp |
| Output | tardy.out |
| Time/test | 1 giây |

# LẬP LỊCH

Siêu máy tính Deep Blue là máy tính mạnh nhất thế giới trong những năm cuối cùng của thế kỷ XX. Nó đã từng thắng cả vua cờ Kasparop. Vì vậy, có rất nhiều bài toán được giao cho nó thực hiện. Tuy nhiên không phải lúc nào nó cũng hoàn thành công việc được giao.

Có *n* chương trình thực hiện trên Deep Blue. Biết:

* *pi* là thời gian cần thiết để hoàn thành chương trình thứ *i*.
* *di* là thời hạn phải giao nộp kết quả của chương trình thứ *i* cho bên đặt hàng.

Máy tính bắt đầu hoạt động tại thời điểm 0. Mỗi chương trình cần phải được thực hiện liên tục kể từ khi bắt đầu cho đến khi kết thúc, không cho phép ngắt quãng. Giả sử *ci* là thời điểm hoàn thành chương trình thứ *i*. Khi đó nếu *ci*≤*di* thì ta nói chương trình này hoàn thành đúng hạn, trường hợp ngược lại là hoàn thành không đúng hạn.

**Yêu cầu:** Tìm trình tự thực hiện các chương trình sao cho số chương trình hoàn thành không đúng hạn là nhỏ nhất.

**Input:**

* Dòng đầu tiên chứa số nguyên dương *n* ()
* Dòng thứ hai chứa *n* số nguyên dương 
* Dòng thứ ba chứa *n* số nguyên dương 

**Output:**

* Dòng đầu tiên ghi số lượng chương trình hoàn thành không đúng hạn
* Dòng tiếp theo ghi *n* số nguyên dương là chỉ số các chương trình được thực hiện theo thứ tự tìm được.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6  2 4 1 2 3 1  3 5 6 6 7 8 | 2  1 3 4 6 2 5 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | riddle.\* |
| Input | riddle.inp |
| Output | riddle.out |
| Time/test | 1 giây |

# BÍ HIỂM

Bà của Ellenora thường ra cho cháu gái mình những bài toán đố mà Elly coi là bí hiểm. Buổi tối vừa rồi bà đố Elly bài toán sau:

“*Ở cửa hàng cạnh nhà ta có* ***k*** *mặt hàng với giá khác nhau từ 1 đến* ***k****. Bà có* ***n*** *đồng tiền mệnh giá* ***a1****,* ***a2****, . . .,* ***an****. Bà định sang bên đấy mua một mặt hàng nào đó, trả đúng giá của nó mà không phải nhận lại tiền thừa. Nhưng bà đã già quá rồi. Bà không muốn mang tất cả tiền của mình đi, có thể lẫn hoặc rơi mất, vì vậy bà chỉ mang theo* ***một số đồng đầu tiên****. Vậy bà phải mang theo bao nhiêu đồng tiền để mua được mặt hàng bất kỳ?*”

Chỉ mất vài giây Elly đã đưa ra được câu trả lời và nghĩ thầm trong bụng: “*Ôi, bà ơi, lại những bài toán giải thuật quá chuẩn!*”.

Bạn có thể đua tài với Elly bằng cách viết chương trình giải bài toán này được không?

**Input:**

* Dòng đầu tiên chứa số nguyên ***T*** – số lượng tests trong file,
* Mỗi test cho trên 2 dòng:
  + Dòng thứ nhất chứa 2 số nguyên ***n*** và ***k*** (1 ≤ ***n*** ≤ 105, 1 ≤ ***k*** ≤ 106),
  + Dòng thứ 2 chứa n số nguyên ***a1****,* ***a2****, . . .,* ***an***, (0 <***ai*** ≤ 105, ∀***i*** = 1 ÷ ***n***).

**Output:** Kết quả mối test đưa ra trên một dòng dưới dạng số nguyên. Nếu không có cách mang thì đưa ra số -1.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3  7 10  1 2 3 4 5 6 7  3 3  2 4 1  3 6  3 1 4 | 3  4  -1 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | doubleq.\* |
| Input | doubleq.inp |
| Output | doubleq.out |
| Time/test | 1 giây |

# HÀNG ĐỢI KÉP

**(Bài tập lý thuyết)**

Ngân hàng BIG-Bank mở một chi nhánh ở Bucharest và được trang bị một máy tính hiện đại với các công nghệ mới nhập, C2#,VC3+ ... chỉ chuối mỗi cái là không ai biết lập trình. Họ cần một phần mềm mô tả hoạt động của ngân hàng như sau: mỗi khách hàng có một mã số là số nguyên K, và khi đến ngân hàng giao dịch, họ sẽ nhận được 1 số P là thứ tự ưu tiên của họ. Các thao tác chính như sau

* **0** Kết thúc phục vụ
* **1 K P** Thêm khách hàng K vào hàng đợi với độ ưu tiên P
* **2** Phục vụ người có độ ưu tiên cao nhất và xóa khỏi danh sách hàng đợi
* **3** Phục vụ người có độ ưu tiên thấp nhất và xóa khỏi danh sách hàng đợi.

Tất nhiên là họ cần bạn giúp rồi.

**Input:**Mỗi dòng của input là 1 yêu cầu, và chỉ yêu cầu cuối cùng mới có giá trị là 0. Giả thiết là khi có yêu cầu 1 thì không có khách hàng nào khác có độ ưu tiên là P. ( K≤106, và P≤ 107.Một khách hàng có thể yêu cầu phục vụ nhiều lần và với các độ ưu tiên khác nhau).

**Output:**  Với mỗi yêu cầu 2 hoặc 3, in ra trên 1 dòng mã số của khách hàng được phục vụ tương ứng. Nếu có yêu cầu mà hàng đợi rỗng, in ra số 0.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 2  1 20 14  1 30 3  2  1 10 99  3  2  2  0 | 0  20  30  10  0 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | nasa.\* |
| Input | nasa.inp |
| Output | nasa.out |
| Time/test | 1 giây |

# KẾ HOẠCH PHÓNG TÀU VŨ TRỤ

Cơ quan hàng không vũ trụ Mĩ NASA vừa thực hiện thành công một dự án lớn. Đó là xây dựng một trạm vũ trụ trên mặt trăng. Công việc trước mắt là duy trì trạm vũ trụ này trong N ngày. Để vận hành tốt, lúc nào cũng cần có một phi hành gia ở trên trạm vũ trụ. Tuy nhiên, mỗi phi hành gia không thể ở trên trạm vũ trụ quá M ngày liên tiếp, vì vậy NASA cần lập một kế hoạch luân phiên các nhà du hành vũ trụ. Chi phí cho việc luân phiên này cũng khác nhau đối với mỗi ngày và NASA muốn tối thiểu tổng chi phí này. Nhiệm vụ của bạn là đọc các thông tin và đưa ra một kế hoạch tối ưu. Chú ý rằng ở ngày thứ 1 luôn cần có sự thay đổi.

**Input:**

* Dòng thứ nhất ghi hai số N và M (1 ≤ N ≤ 100000, 1 ≤ M ≤ 10000) .
* Dòng thứ i trong N dòng sau ghi số Ci chi phí cho việc thay đổi nhà du hành vũ trụ trong ngày thứ i (0 ≤ Ci ≤ 105).

**Output:** Gồm một dòng duy nhất ghi S là chi phí tối thiểu cho việc duy trì trạm vũ trụ.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 9 3  1  1  1  5  1  5  2  1  1 | 4  (Thay người ở các ngày thứ 1, 3, 5, 8) |  |

Ghi chú: Có 60% số test có N ≤100

|  |  |
| --- | --- |
| **Summary** | |
| Program | kmin.\* |
| Input | kmin.inp |
| Output | kmin.out |
| Time/test | 1 giây |

# K TỔNG BÉ NHẤT

Cho 2 dãy số nguyên A và B. Với mọi số *ai* thuộc A,*bj* thuộc B (1 ≤ *ai, bj* ≤ 109 ) người ta tính tổng nó. Tất cả các tổng này sau khi được sắp xếp không giảm sẽ tạo thành dãy C.

Nhiệm vụ của bạn là: Cho 2 dãy A, B. Tìm K số đầu tiên trong dãy C

**Input:**

* Dòng đầu tiên gồm 3 số: M, N, K (1≤M, N, K≤50000)
* M dòng tiếp theo gồm M số mô tả dãy A
* N dòng tiếp theo gồm N số mô tả dãy B

**Output:**Gồm K dòng tương ứng là K phần tử đầu tiên trong dãy C.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4 4 6  1  2  3  4  2  3  4  5 | 3  4  4  5  5  5 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | paint.\* |
| Input | paint.inp |
| Output | paint.out |
| Time/test | 1 giây |

# SƠN HÀNG RÀO

Mr Bean muốn sơn lại hàng rào nhà mình. Hàng rào của anh ấy được ghép bởi N tấm ván liên tiếp, mỗi tấm rộng 1 cm và có chiều cao khác nhau. Để sơn nhanh chóng và dễ dàng hơn, Mr Bean đã mua "*Super Pain Roller Deluxe*" (một cây lăn xịn). Cây lăn có chiều rộng x cm. Mỗi lần dùng, Mr Bean phải để cây lăn chạm vào bức tường hoàn toàn, nếu không sẽ không sơn được gì cả. Mặt khác, lúc nào cũng phải để cho cây lăn song song với mặt đất. Nói cách khác, Mr Bean sẽ chọn ra x tấm ván liên tiếp và sơn từ dưới lên đến độ cao của tấm ván thấp nhất trong x tấm ván đó.

Tuy nhiên, cây lăn không thể giúp Mr Bean sơn hết được hàng rào. Phần diện tích còn lại không thể sơn bằng cây lăn, anh ta sẽ sơn bằng cọ nhỏ. Hãy giúp Mr Bean tính xem phần diện tích phải sơn bằng cọ nhỏ nhất có thể là bao nhiêu?.

**Input:**

* Dòng đầu ghi số nguyên N (1 ≤ N ≤ 106) là số tấm ván cần phải sơn và số nguyên X (1 ≤ X ≤ 105).
* Dòng tiếp theo ghi N số nguyên dương (mỗi số không quá 106) mô tả chiều cao của từng tấm ván.

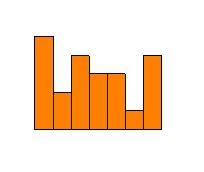
**Output:** Một dòng ghi số nguyên là diện tích nhỏ nhất phải sơn bằng cọ

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5 3  5 3 4 4 5 | 3 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | wood.\* |
| Input | wood.inp |
| Output | wood.out |
| Time/test | 1 giây |

# CẮT GỖ

Có N mảnh gỗ hình chữ nhật độ rộng đơn vị được ghép lại với nhau tạo thành một tấm ván to hơn (*hình vẽ dưới).* Mảnh ván thứ i có chiều cao là ai:

Người ta cắt tấm ván lớn này chỉ giữ lại thành một hình vuông theo nguyên tắc: *Chiều cao của hình vuông là chiều cao của một mảnh gỗ nào đó và chiều rộng là một hoặc nhiều mảnh gỗ đã được ghép cạnh nhau*.

**Yêu cầu:** Hãy tìm hình vuông có cạnh lớn nhất nhận được nếu cắt tấm ván theo kiểu trên.

**Input:**

* Dòng đầu tiên ghi N là số mảnh gỗ (N≤106)
* Tiếp theo là một số dòng ghi các số nguyên dương a1, a2, ..., an theo thứ tự. Hai số cạnh nhau trên cùng một dòng ghi cách nhau ít nhất một khoảng trẳng (1≤ai≤109)

**Output:** Một số nguyên duy nhất là độ dài cạnh hình vuông tìm được

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6  4 3 8 8 8 8 | 3 |  |
| 7  5 2 4 3 3 1 4 | 3 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | eva.\* |
| Input | eva.inp |
| Output | eva.out |
| Time/test | 1 giây |

# SƠ TÁN

Một Trung tâm nghiên cứu tuyệt mật (mà chúng ta không có quyền nói rõ tên ở đây) có ***n*** phòng thí nghiệm đặt ngầm trong lòng đất. Các phòng thí nghiệm được đánh số từ 1 đến ***n*** (1 ≤ ***n*** ≤ 105). Giữa một số phòng có đường hầm nối với nhau, sao cho từ một phòng bất kỳ có thể đi đến phòng bất kỳ khác (có thể phải đi qua một số phòng nào đó). Độ dài mỗi đường hầm là như nhau và thời gian đi hết một đường hầm là 1. Không có đường hầm nào nối một phòng với chính nó, nhưng có thể có nhiều đường hầm cùng nối 2 phòng với nhau và tổng cộng trong Trung tâm có tất cả ***m*** đường hầm (1 ≤ ***m*** ≤ 105). Đường hầm cho phép đi lại theo cả hai chiều. Có ***k*** phòng có lối thoát hiểm lên mặt đất (1 ≤ ***k*** ≤ ***n***). Trong trường hợp sơ tán khẩn cấp, tất cả các nhân viên phải tập trung ở những phòng có lối thoát hiểm.

***Yêu cầu***: Hãy xác định thời gian tối thiểu để nhân viên mỗi phòng tập trung về phòng có lối thoát hiểm trong trường hợp phải sơ tán khẩn cấp.

**Input:**

* Dòng đầu tiên chứa 2 số nguyên ***n*** và ***k***,
* Dòng thứ 2 chứa ***k*** số nguyên khác nhau cho biết các phòng có cửa thoát hiểm,
* Dòng thứ 3 chứa số nguyên ***m***,
* Mỗi dòng trong ***m*** dòng tiếp theo chứa 2 số nguyên xác định cặp phòng có đường hầm nối trực tiếp.

**Output:**Một dòng chứa ***n*** số nguyên, số thứ ***i*** xác định thời gian tối thiểu để nhân viên phòng i đi được tới phòng có lối thoát hiểm.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 10  2  10 8  9  6 7  7 5  5 8  8 1  1 10  10 3  3 4  4 9  9 2 | 1 4 1 2 1 3 2 0 3 0 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | castle.\* |
| Input | castle.inp |
| Output | castle.out |
| Time/test | 1 giây |

# LÂU ĐÀI

Nông dân John trúng thưởng sổ số độc đắc của Ailen và phần thưởng đó, thật ngạc nhiên, là một tòa lâu đài có nhiều phòng. John rất muốn biết tòa lâu đài của mình có bao nhiêu phòng và hơn thế nữa ông ta muốn phá một bức tường trong lâu đài để có thể có được một phòng lớn hơn (dành chỗ cho các con bò của ông ta tụ tập vui vẻ !!!).

Bạn hãy viết một chương trình giúp John xác định xem tòa lâu đài có bao nhiêu phòng và cần phải phá bức tường nào để có được một phòng lớn nhất có thể.

Sơ đồ lâu đài có thể mô tả như một lưới M cột và N hàng các modul. Mỗi một Modul như vậy có thể có 0, 1, 2, 3, 4 bức tường bao quanh nó. Tất nhiên các modul trên biên của lâu đài luôn có các bức tường ngăn nó với bên ngoài để cho modul không bị ảnh hưởng của gió và mưa:

Chẳng hạn dưới đây là một sơ đồ lâu đài:

1 2 3 4 5 6 7

#############################

1 # | # | # | | #

#####---#####---#---#####---#

2 # # | # # # # #

#---#####---#####---#####---#

3 # | | # # # # #

#---#########---#####---#---#

4 # -># | | | | # #

#############################

# = tường -,| = không tường

-> = vị trí bức tường cần phá để có một phòng mới có diện tích lớn nhất

Trong ví dụ trên, lâu đài có 4x7 modul. Nó có 5 phòng (có kích thước là 9, 7, 3, 1 và 8). Việc phá bức tường đánh dấu sẽ cho một phòng mới có diện tích lớn nhất trong số các phòng mới nhận được bằng cách phá đi chỉ một bức tường.

Tòa lâu đài có ít nhất hai phòng và luôn có một bức tường có thể bị phá

**Input:**

* Dòng đầu tiên chứa hai số M và N (1≤M,N≤50) là số cột và số hàng của tòa lâu đài
* Tiếp theo là N x M số nguyên (có thể có nhiều số trên một dòng cách nhau bằng dấu cách) lần lượt mô tả các modul của dòng 1 từ tây sang đông, các mô dul của dòng 2 từ tây sang đông,... mỗi modul được gán một giá trị nguyên theo qui tắc như sau: 1-có tường phía tây, 2-có tường phía bắc, 4-có tường phía đông, 8-có tường phía nam. Nếu có nhiều bức tường xung quanh một modul thì giá trị biểu diễn modul đó bằng tổng giá trị biểu diễn của các bức tường. Ví dụ nếu modul có tường phía tây, đông, nam thì giá trị của nó là 1+4+8 = 13 (!).

**Output:**

* Dòng 1: Ghi số lượng phòng mà lâu đài có thể có
* Dòng 2: Ghi số modul của phòng rộng nhất
* Dòng 3: Ghi số modul của phòng rộng nhất sau khi phá đi một bức tường
* Mô tả bức tường cần phá. Nếu có nhiều bức tường có thể chọn thì chọn bức tường xa nhất về hướng tây. Nếu vẫn còn nhiều bức tường cùng xa nhất về hướng tây có thể chọn thì chọn bức tường xa nhất về hướng nam. Vị trí của bức tường cần phá được xác định bởi (i,j) - chỉ số hàng và cột của modul chứa bức tường và hoặc N (nếu tường ở hướng bắc) hoặc E (nếu tường ở hướng đông)

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 7 4  11 6 11 6 3 10 6  7 9 6 13 5 15 5  1 10 12 7 13 7 5  13 11 10 8 10 12 13 | 5  9  16  4 1 E |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | circus.\* |
| Input | circus.inp |
| Output | circus.out |
| Time/test | 1 giây |

# BIỂU DIỄN XIẾC

Xe đạp một bánh là hình ảnh quen thuộc trong các tiết mục biểu diễn xiếc. Để tăng tính hấp dẫn cho tiết mục, đạo diễn quyết định "tin học hoá" nó: chia bánh xe đạp đặc (không có nan hoa) thành 5 sector, mỗi sector 720 và sơn 1 trong 5 màu khác nhau đánh số từ 1 đến 5. Sàn diễn được chia thành lưới M x N ô vuông, mỗi ô vuông có cạnh bằng 1/5 chu vi của bánh xe. Như vậy khi bánh xe tiếp xúc với sàn tại tâm của ô vuông theo một màu nào đó, thì sang ô bên cạnh - màu khác sẽ tiếp xúc với nó (Màu 5 thành 4, 4 thành 3, 3 thành 2, 2 thành 1, 1 thành 5).

Square 3

Square 3

Square 1

Diễn viên phải xuất phát từ ô ban đầu S tiến tới ô đích T sau khoảng thời gian ngắn nhất theo các qui tắc sau:

* Quay mặt về phía nào thì phải theo hướng đó
* Chỉ được phép đi qua các ô trống
* Tại tâm một ô có thể xoay 900 về phía phải hay trái
* Xuất phát ở trạng thái màu CS tiếp xúc với sàn và kết thúc ở T với màu cT tiếp xúc với sàn.
* Mỗi thao tác xoay hoặc chuyển sang ô mới mất 1 giây
* Ban đầu, diễn viên quay mặt về phía Bắc (Các dòng được dánh số từ 1 đến M theo chiều Bắc - Nam và các cột được đánh số từ 1 đến N theo chiều Tây - Đông)

**Input:**

* Dòng đầu tiên chứa 4 số nguyên M, N, cS, cT (0<M,N≤50)
* M dòng sau: mỗi dòng chứa một xâu N ký tự mô tả trạng thái của các dòng tương ứng với S - ô xuất phát, T - ô đích, # - ô có vật cản, dấu chấm (.) - ô trống.

**Output:**Thời gian hoặc -1, nếu không thể đi được.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 10 10 1 1  #S.......#  #..#.##.##  #.##.##.##  .#....##.#  ##.##..#.#  #..#.##...  #......##.  ..##.##...  #.###...#.  #.....###T | 49 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | mnet.\* |
| Input | mnet.inp |
| Output | mnet.out |
| Time/test | 1 giây |

# MẠNG MÁY TÍNH

Một hệ thống N máy tính đánh số từ 1 đến N được nối mạng bằng M kênh truyền tin một chiều giữa một số cặp máy. Mạng cho gọi là thông suốt nếu như từ một máy u bất kỳ luôn có thể truyền tin đến mỗi máy v trong số các máy còn lại hoặc theo kênh truyền tin từ u đến v hoặc thông qua một số kênh trung gian. Ta gọi một mạng con của mạng đã cho là một mạng gồm một số máy và các kênh nối chúng của mạng đã cho. Trong trường hợp mạng là không thông suốt nó sẽ phân rã thành một số mạng con thông suốt. Mạng con thông suốt được gọi là cực đại nếu như không tồn tại một mạng con thông suốt của mạng đã cho chứa nó như một mạng con. Bạn cần xác định số mạng con thông suốt cực đại của mạng dã cho.

**Input:**

* Dòng đầu tiên chứa hai số ngyên dương N, M (1≤N,M≤105)
* Dòng thứ i trong M dòng tiếp theo ghi 2 số nguyên dương di, ci cho biết kênh truyền tin thứ i cho phép truyền tin từ máy di sang máy ci.

**Output:** Dòng đầu tiên ghi K là số mạng con thông suốt cực đại của mạng đã cho

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output | Giải thích |
| 9 14  1 2  1 4  1 7  2 3  2 6  3 1  4 5  5 4  5 7  6 7  7 9  9 6  9 8  8 9 | 3 | Các mạng thông suốt cực đại là:  1 2 3  4 5  6 7 8 9 |

|  |  |
| --- | --- |
| **Summary** | |
| Program | roads.\* |
| Input | roads.inp |
| Output | roads.out |
| Time/test | 1 giây |

# NÂNG CẤP ĐƯỜNG

Bộ giao thông vận tải ở Romania quyết định tiến hành nâng cấp các đọan đường nối trực tiếp các thành phố với nhau. Mạng lưới giao thông của Romania có thể được mô tả dưới dạng một đồ thị vô hướng liên thông trong đó các đỉnh đại diện cho các thành phố và các cung đại diện cho các đọan đường trực tiếp nối các thành phố, không có hai đoạn đường nối trực tiếp cùng hai thành phố. Quá trình nâng cấp sẽ được diễn ra tuần tự giữa các đọan đường nghĩa là tại mỗi thời điểm chỉ có một đọan đường được sửa và trong suốt thời gian sửa chữa, nâng cấp đọan đường đó không được sử dụng. Do đó,trong suốt quá trình nâng cấp có thể xảy ra trường hợp tắc nghẽn giao thông giữa một số thành phố (không tồn tại tuyến đường giao thông liên lạc giữa 2 thành phố nào đó). Để đảm bảo giao thông thông suốt, bộ giao thông vận tải quyết định sẽ xây dựng thêm một số đọan đường phụ để hỗ trợ. Vì lý do kinh tế nên tổng số đọan đường phụ cần bổ sung phải là ít nhất có thể được (giả thiết chi phí xây dựng các đọan đường là như nhau).

**Input:**

* Dòng đầu tiên ghi hai số N và M là số thành phố và số đoạn đường nối trực tiếp giữa chúng (3 ≤ N ≤ 5000, 2 ≤ M ≤ 30000)
* Trong M dòng sau, mỗi dòng ghi hai số U và V thể hiện có đường nối trực tiếp giữa U và V.

**Output:**Ghi số K là số đoạn đường tối thiểu cần xây dựng thêm.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4 3  1 2  2 3  2 4 | 2 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | safenet.\* |
| Input | safenet.inp |
| Output | safenet.out |
| Time/test | 1 giây |

# MẠNG MÁY TÍNH AN TOÀN

Có ***n*** máy tính đánh số từ 1 đến ***n*** và ***m*** dây cáp mạng,giữa 2 máy tính có thể có một hoặc nhiều đường dây cáp mạng nối chúng, không có cáp mạng nối một máy với chính nó.Hai máy tính có thể truyền dữ liệu cho nhau nếu có đường cáp nối trực tiếp giữa chúng hoặc truyền qua một số máy trung gian.

Một tập **S** các máy tính được gọi là hệ thống an toàn nếu dù một máy tính bất kỳ bị tấn công (chẳng hạn bị hacker tấn công) thì trong số những máy tính còn lại,những máy tính thuộc tập **S** vẫn có thể truyền được dữ liệu cho nhau. Xác định số lượng lớn nhất có thể các máy tính của tập **S**

**Input:**

* Dòng thứ nhất chứa hai số nguyên ***n,m*** (1≤***n***≤30000,0≤*m*≤100000)
* ***m*** dòng tiếp theo ghi thông tin về các dây cáp mạng,gồm 2 chỉ số của 2 máy được dây đó nối trực tiếp

**Output:**

Ghi một số nguyên duy nhất là số lượng máy tính lớn nhất tìm được.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 8 10  1 2  2 3  3 1  1 4  4 5  5 1  1 6  6 7  7 8  8 1 | 4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | school.\* |
| Input | school.inp |
| Output | school.out |
| Time/test | 1 giây |

# ĐƯỜNG ĐẾN TRƯỜNG

Hùng đã trở thành học sinh của trường chuyên Nguyễn Trãi. Là học sinh mới, Hùng không muốn đi học muộn nên đã chuẩn bị khá kỹ càng. Chỉ còn lại một công việc khá gay go là Hùng không biết đi đường nào tới trường là nhanh nhất. Thường ngày Hùng không quan tâm tới vấn đề này lắm cho nên bây giờ Hùng không biết phải làm sao cả . Bản đồ thành phố là gồm có N nút giao thông và M con đường nối các nút giao thông này. Có 2 loại con đường là đường 1 chiều và đường 2 chiều. Độ dài của mỗi con đường là một số nguyên dương. Nhà Hùng ở nút giao thông 1 còn trường Nguyễn Trãi ở nút giao thông N. Vì một lộ trình đường đi từ nhà Hùng tới trường có thể gặp nhiều yếu tố khác như là gặp nhiều đèn đỏ , đi qua công trường xây dựng, ... phải giảm tốc độ cho nên Hùng muốn biết là có tất cả bao nhiêu lộ trình ngắn nhất đi từ nhà tới trường. Bạn hãy lập trình giúp Hùng giải quyết bài toán khó này.

**Input:**

* Dòng thứ nhất ghi hai số nguyên N và M.
* M dòng tiếp theo, mỗi dòng ghi 4 số nguyên dương K, U, V, L. Trong đó
  + K = 1 có nghĩa là có đường đi một chiều từ U đến V với độ dài L.
  + K = 2 có nghìa là có đường đi hai chiều giữa U và V với độ dài L.

**Output:**

* Ghi hai số là độ dài đường đi ngắn nhấn và số lượng đường đi ngắn nhất. Biết rằng số lượng đường đi ngắn nhất không vượt quá phạm vì ***int64*** trong ***pascal*** hay ***long long*** trong ***C++***.

Giới hạn:

* 1 ≤ N ≤ 5000
* 1 ≤ M ≤ 20000
* Độ dài các con đường ≤ 32000
* Thời gian: 1 s/test
* Bộ nhớ: 1MB

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 3 2  1 1 2 3  2 2 3 1 | 4 1 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | robots.\* |
| Input | robots.inp |
| Output | robots.out |
| Time/test | 1 giây |

# KIỂM TRA ĐƯỜNG TRỤC

Trong mạng lưới đường ống dẫn dầu có n trạm điều áp, đánh số từ 1 đến ***n*** và có ***m*** đoạn đường ống, mỗi đoạn nối 2 trạm điều áp (2 ≤ ***n***, ***m*** ≤ 105). Mạng có tính liên thông, tức là giữa hai trạm điều áp bao giờ cũng có đường ống nối với nhau (trực tiếp hoặc qua các trạm khác). Một đoạn đường ống được gọi là trục nếu nó hỏng thì hệ thống mất liên thông. Trong hệ thống mà chúng ta đang xét có ít nhất một đoạn đường trục.

Do tính chất quan trọng của đường trục nên chúng được ưu tiên trong công tác duy tu bảo dưỡng. Người ta chế tạo 2 rô bốt phục vụ kiểm tra đường trục. Khi được lệnh kiểm tra 2 rô bốt (có thể đang ở những trạm khác nhau) sẽ lựa chọn một đoạn đường trục và đồng thời chuyển động tập kết tới hai đầu của đoạn đường trục này, mỗi rô bốt tới một đầu của đoạn trục. Rô bốt chuyển động theo đường ống, mỗi đơn vị thời gian đi được một đơn vị độ dài. Thời gian tập kết là thời gian cần thiết để rô bốt đến sau tới được vị trí tập kết của mình. Rô bốt luôn lựa chọn đoạn đường trục cho thời gian tập kết là nhỏ nhất.

***Yêu cầu:*** Cho cấu hình của mạng, các trạm ***u***, ***v*** đang giữ rô bốt. Hãy xác định thời gian tập kết.

**Input:**

* Dòng đầu tiên chứa 2 số nguyên ***n*** và ***m***,
* Mỗi dòng trong ***m*** dòng sau chứa 3 số nguyên ***x***, ***y*** và ***d*** xác định đoạn đường ống độ dài ***d*** nối 2 trạm ***x*** và ***y***,
* Dòng cuối cùng chứa 2 số nguyên ***u*** và ***v***.

**Output:**Một số nguyên – thời gian tập kết.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 8 11  1 2 3  1 3 5  1 4 8  2 4 3  3 4 4  4 5 2  5 6 9  5 7 3  6 7 4  6 8 5  7 8 6  3 6 | 7 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | silkroad.\* |
| Input | silkroad.inp |
| Output | soilkroad.out |
| Time/test | 1 giây |

# CON ĐƯỜNG TƠ LỤA

Con đường tơ lụa bắt đầu từ Phúc Châu, Hàng Châu, Bắc Kinh (Trung Quốc) qua Mông Cổ, Ấn Độ, Afghanistan, Kazakhstan, Iran, Iraq, Thổ Nhĩ Kỳ, Hy Lạp, xung quanh vùng Địa Trung Hải và đến tận châu Âu. Con đường cũng đi đến cả Hàn Quốc, Nhật Bản.

Mạng lưới giao thông của hệ thống đường này bao gồm *n* trạm dừng chân cho ngựa và khách buôn bán (các trạm đánh số từ 1 đến *n*). Giữa một số trạm có các con đường hiểm trở hai chiều (thủy hoặc bộ) nối trực tiếp và tùy theo địa hình, thời gian đi hết mỗi con đường này là khác nhau.

Nữ hoàng Ai Cập Cleopatra rất thích diện đồ tơ luạ ở Bắc Kinh và luôn lập các đoàn thương nhân đến Bắc Kinh để tìm mua loại tơ lụa nổi tiếng này .Đoàn thương nhân của nữ hoàng luôn xuất phát từ trạm dừng chân số 1 và đích đến là trạm *n* thông qua các đường nối khác nhau của con đường tơ lụa. Để đảm bào sức khỏe cho một cuộc hành trình dài các thương nhân cần phải chọn hành trình sao cho thời gian đi giữa hai trạm dừng chân càng ngắn càng tốt.

Bạn hãy giúp họ tìm con đường đi từ 1 đến *n* sao cho thời gian đi dài nhất giữa hai trạm dừng chân liên tiếp là ngắn nhất.

**Input:**

* Dòng đầu tiên ghi *n*, *m* là số trạm và số tuyến đường (*n*≤105, *m* ≤106)
* *m* dòng tiếp theo mỗi dòng ghi ba số nguyên u, v, w thể hiện có một đường nối trực tiếp giữa u và v có thời gian đi là w (1≤w≤109)

**Output:**Một số nguyên duy nhất là thời gian của con đường đi lâu nhất.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4 4  1 3 2  3 4 3  1 2 5  2 4 3 | 3 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | billboards.\* |
| Input | billboards.inp |
| Output | billboards.out |
| Time/test | 1 giây |

# BIỂN QUẢNG CÁO

Hàng rào của nhà Mr Bean gồm một dãy N tấm gỗ có chiều rộng bằng 1 đơn vị và chiều cao lần lượt là .

Mặt tiền nhà của Mr Bean là nơi rất nhiều người qua lại do đó Mr Bean quyết định cho thuê quảng cáo. Tất nhiên các tấm quảng cáo có dạng hình chữ nhật, chiều rộng phủ lên một số nguyên lần các tấm gỗ của hàng rào và không có phần nào của tấm quảng cáo mà phía sau không có tấm gỗ chắn.

Hãy tính diện tích lớn nhất của một tấm quảng cáo như vậy

**Input:**

* Dòng đầu tiên ghi số nguyên dương N ()
* Dòng thứ hai ghi N số nguyên dương )

**Output:** Một dòng duy nhất ghi diện tích lớn nhất của một tấm quảng cáo

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4  3 4 3 1 | 9 |  |
| 4  1 2 1 3 | 4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | maxrec.\* |
| Input | maxrec.inp |
| Output | maxrec.out |
| Time/test | 1 giây |

# HÌNH CHỮ NHẬT

Cho một lươi hình chữ nhật kích thước M x N được chia thành M hàng và N cột. Mỗi một ô vuông của lưới ghi hoặc số 1 hoặc số 0.

Hãy tìm hình chữ nhật con trên lưới trên gồm một số nguyên lần các ô vuông, có các cạnh song song với các cạnh của lưới hình chữ nhật, chứa toàn số 1 và có diện tích lớn nhất.

**Input:**

* Dòng đầu tiên ghi hai số nguyên M, N ()
* M dòng tiếp theo, mỗi dòng ghi N số 0/1 mô ta lưới chữ nhật

**Output:** Một số nguyên duy nhất là diện tích (số ô vuông) lớn nhất của hình chữ nhật chứa toàn số 1

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4 5  0 0 1 1 1  0 1 1 1 1  1 1 1 1 1  0 0 0 0 0 | 9 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | seqarek.\* |
| Input | seqarek.inp |
| Output | seqarek.out |
| Time/test | 1 giây |

# DÃY CON ĐỀU BẬC K

Cho dãy số nguyên . Một dãy con của dãy đã cho là dãy các phần tử liên tiếp nhau.

Dãy con  được gọi là dãy con đều bậc *k* nếu như chênh lệch giữa hai số bất kỳ trong dãy này không vượt quá *k*.

Hãy tìm dãy con đều bậc *k* có độ dài lớn nhất.

**Input:**

* Dòng đầu tiên ghi hai số nguyên dương *n*, *k* ()
* Dòng thứ hai ghi *n* số nguyên ()

**Output:** Một số nguyên duy nhất là độ dài của dãy con tìm được

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 5 3  2 1 3 4 5 | 4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | track.\* |
| Input | track.inp |
| Output | track.out |
| Time/test | 1 giây |

# ĐƯỜNG NHÁNH

Nhà ga xe lửa ở cuối tuyến có ***k*** đường nhánh cho các đoàn tàu đến bảo dưỡng chờ đợi khởi hành. Các đường nhánh được đánh số từ 1 đến ***k***. Khi có một đoàn tàu tới, nó sẽ được đưa vào đường nhánh còn trống có số hiệu nhỏ nhất. Nếu một đoàn tàu xuất phát từ đường nhánh ở thời điểm x, thì đoàn tàu tới ga vào thời điểm ***x*** không thể vào đường nhánh này, nhưng tàu tới ga vào thời điểm ***x***+1 thì có thể.

***Yêu cầu***: Cho biết ***ai***, ***bi*** – thời điểm đến và đi của mỗi đoàn tàu (***i*** = 1 ÷ ***n***, 0 ≤ ***ai***<***bi*** ≤ 109, 1 ≤ ***n*** ≤ 5×105), số đường nhánh ***k*** (1 ≤ ***k*** ≤ 5×105). Các đoàn tàu được đánh số từ 1 đến ***n***. Không có hai đoàn tàu nào đến cùng một thời điểm, nhưng có thể rời ga cùng một lúc. Thông tin về các đoàn tàu được cho theo trình tự tăng dần của thời điểm đến. Hãy xác định số đường nhánh cần thông báo cho mỗi đoàn tàu. Nếu không có đường nhánh thì đưa ra số 0 và số hiệu đoàn tàu đầu tiên không thể tới ga vì thiếu đường nhánh.

**Input:** Vào từ file văn bản TRACK.INP:

* Dòng đầu tiên chứa 2 số nguyên ***k*** và ***n***,
* Dòng thứ ***i*** trong ***n*** dòng sau chứa 2 số nguyên ***ai*** và ***bi***.

**Output:** Đưa ra file văn bản TRACK.OUT. Trong trường hợp bố trí được – đưa ra ***n*** số nguyên, số thứ ***i*** là đường nhánh cho đoàn tàu thứ ***i***. Nếu không bố trí được – đưa ra 2 số nguyên theo yêu cầu.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 2 3  1 3  2 6  4 5 | 1  2  1 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | substr.\* |
| Input | substr.inp |
| Output | substr.out |
| Time/test | 1 giây |

# XÂU CON

Cho hai xâu ký tự A và B chỉ gồm các chữ cái tiếng Anh in thường. Xâu A có *n* ký tự và xâu B có *m* ký tự.

Ta nói rằng xâu B xuất hiện tại vị trí *i* của xâu A nếu như:



Ở đây *ai* chỉ ký tự thứ *i* của xâu A còn *bj* chỉ ký tự thứ *j* của xâu B.

Hãy tìm tất cả các vị trí xuất hiện xâu B trong xâu A

**Input:**

* Dòng 1: ghi xâu A
* Dòng 2: ghi xâu B

Số lượng ký tự trong mỗi xâu không vượt quá 106.

**Output:** Ghi ra các vị trí xuất hiện xâu B trong xâu A (theo thứ tự tăng dần). Nếu xâu B không xuất hiện trong xâu A thì ghi số 0.

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| aaaaa  aa | 1 2 3 4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | maxstr.\* |
| Input | maxstr.inp |
| Output | maxstr.out |
| Time/test | 1 giây |

# DÃY CON DÀI NHẤT

Cho dãy ký tự S chỉ gồm các chữ cái in thường tiếng Anh. Hãy tìm độ dài của dãy con dài nhất (gồm các ký tự liên tiếp của S) xuất hiện trong S ít nhất hai lần

**Input:**

* Dòng đầu tiên ghi số nguyên dương L (1≤L≤200000) là độ dài xâu S
* Dòng thứ hai ghi L ký tự mô tả xâu S

**Output:**Ghi một dòng duy nhất số nguyên là độ dài của dãy con dài nhất tìm được. Nếu không tìm được dãy con nào thì ghi số 0

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 11  sabcabcfabc | 3 |  |
| 18  trutrutiktiktappop | 4 |  |
| 6  abcdef | 0 |  |

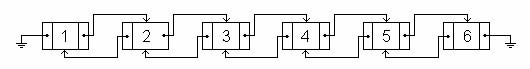
|  |  |
| --- | --- |
| **Summary** | |
| Program | list.\* |
| Input | list.inp |
| Output | list.out |
| Time/test | 1 giây |

# KHÔI PHỤC LẠI

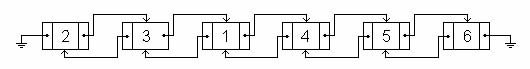
Trong ngày sinh nhật Mirko nhận được từ người cô ở Mỹ một món quà đặc biệt - một đồ chơi có cấu trúc như một danh sách liên kết đôi (xem hình dưới đây). Danh sách chứa N nút đánh số từ 1 đến N. Có hai thao tác sau được thực hiện trên danh sách:

A) Di chuyển nút X đến trước nút Y

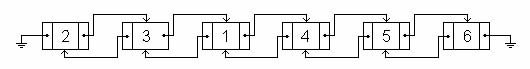
B) Di chuyển nút X đến sau nút Y



*Ví dụ về danh sách đôi có 6 nút*



*Các nút sau khi thực hiệnl ệnh A 1 4*



*Các nút sau khi thực hiện tiếp lệnh B 3 5*

Mirko chơi trò chơi này hàng giờ liền, mỗi lần viết thao tác di chuyển lên mảnh giấy. Cuối buổi chơi anh ta quyết định khôi phục lại danh sách ban đầu cũng bằng hai thao tác trên (chsu ý rằng khởi đầu các nút đánh số từ 1 đến N từ trái sang phải)

Viết chương trình xác định dãy các di chuyển để khôi phục lại danh sách ban đầu với số thao tác là ít nhất.

**Input:**

* Dòng đầu tiên ghi hai số nguyên dương N, M (1≤N≤5.105, 1≤M≤105) là số lượng nút và số lượng thao tác.
* M dòng tiếp theo, mỗi dòng ghi một thao tác mà Mirko thực hiện theo thứ tự. bắt đầu bằng ký tự 'A' hoặc 'B' và tiếp theo là hai số X, Y

**Output:**

* Dòng đầu tiên ghi K là số lượng thao tác tối thiểu cần thực hiện.
* K dòng tiếp theo, mỗi dòng ghi một thao tác thực hiện theo trình tự với qui cách giống như input.

Chú ý rằng kết quả không phải là duy nhất

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4 3  B 1 2  A 4 3  B 1 4 | 2  A 1 2  B 4 3 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | qmax.\* |
| Input | qmax.inp |
| Output | qmax.out |
| Time/test | 1 giây |

# GIÁ TRỊ LỚN NHẤT

Cho dãy *n* số nguyên *a1, a2, ..., an*. Người ta thực hiện *m* thao tác trên dãy này. Mỗi thao tác có một trong hai dạng:

1. **A u v k**: Tăng các phần tử có chỉ số trong đoạn [u,v] lên một lượng k
2. **Q u v**: In ra giá trị lớn nhất của *au*, *au+1, ..., av*

Hãy viết chương trình in ra kết quả các thao tác loại 2.

**Input:**

* Dòng đầu tiên ghi số nguyên dương *n* (1≤*n*≤105)
* Dòng thứ hai ghi *n* số nguyên *a1, a2, ..., an*
* Dòng thứ ba ghi số nguyên dương *m* (1≤*m*≤105)
* *m* dòng tiếp theo mỗi dòng ghi một thao tác theo qui cách nêu trên

Các kết quả trung gian có giá trị tuyệt đối không vượt quá 109, có ít nhất môt thao tác loại 2

**Output:** Kết quả các thao tác loại 2 theo thứ tự, mỗi kết quả được in trên một dòng

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 6  0 0 0 0 0 0  3  A 1 3 3  A 4 6 4  Q 1 6 | 4 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | inverse.\* |
| Input | inverse.inp |
| Output | inverse.out |
| Time/test | 1 giây |

# SỐ NGHỊCH THẾ

Cho {*x1, x2, ..., xn*} là một hoán vị của {1,2,...,*n*}. Ta gọi một nghịch thế là một cặp (*xi, xj*) thỏa mãn *i*<*j* và *xi*>*xj*.

Hãy đếm số nghịch thế của hoán vị đã cho

**Input:**

* Dòng đầu tiên ghi số nguyên dương *n* (1≤*n*≤105)
* Dòng thứ hai ghi *x1, x2, ..., xn* là một hoán vị của {1,2,...,*n*}

**Output:** Một số nguyên duy nhất là số nghịch thế của hoán vị

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| 4  2 1 4 3 | 2 |  |

|  |  |
| --- | --- |
| **Summary** | |
| Program | keylogger.\* |
| Input | keylogger.inp |
| Output | keylogger.out |
| Time/test | 1 giây |

# PHÁ MẬT KHẨU

Kim Oanh đang tìm hiểu về công việc của một Hacker và cô ta khá quan tâm đến việc tại sao các Hacker có thể lấy được mật khẩu của người dùng. Để tìm hiểu về quá trình này Kim Oanh đã tự mình viết một chương trình tương tự và nó được gọi là KeyLogger. Chương trình này hoạt động khá đơn giản. Nó ghi lại toàn bộ hoạt động bàn phím của người dùng và trên cơ sở đó có thể phân tích để tìm ra mật khẩu.

Thật không may, một số người dùng khá tinh quái đã sử dụng thêm các phím thừa để mật khẩu khó được tìm ra bằng các phím sang trái, sang phải và xóa một ký tự.

Hãy viết chương trình tìm mật khẩu của người dùng để cùng thi với Kim Oanh.

**Input:** Gồm một dòng duy nhất có không quá 106 ký tự thể hiện các phím mà người dùng đã ấn:

* '-': thể hiện xóa một ký tự đằng trước (nếu có)
* '>' và '<': thể hiện di chuyển con trỏ sang trái hoặc sang phải nếu có thể
* Chữ cái hoặc chữ số: nó sẽ là một phần mật khẩu nếu không bị xóa. Nếu có một ký tự được chèn vào một vị trí nào đó thì toàn bộ ký tự sau vị trí này sẽ dịch sang phải.

Dữ liệu đảm bảo mật khẩu được giải mã có độ dài lớn hơn 0.

**Output:** Một dòng duy nhất ghi mật khẩu tìm được

**Example:**

|  |  |  |
| --- | --- | --- |
| Input | Output |  |
| <<BP<A>>Cd- | BAPC |  |
| ThIsIsS3Cr3t | ThIsIsS3Cr3t |  |