

# Universal cup stage 31 : Wroclaw

Created: 4/24/2025 14:34

Updated: 4/24/2025 14:50

Exported: 4/24/2025 14:53

## Dịch đề bài:

### Bài A. Quốc ca

**Giới hạn thời gian:** 1 giây

**Giới hạn bộ nhớ:** 1024 MB

Dwarf the Piper (người lùn thổi sáo) đã chơi *Quốc ca của người lùn* (DNA) trên chiếc đàn organ ống khổng lồ của mình mỗi sáng suốt nhiều thế kỷ. Nhưng bây giờ, ông ấy phải chuyển đến một hang động mới với không gian hạn chế, nơi mà chiếc đàn organ cũ không còn vừa nữa.

Quốc ca DNA là một chuỗi các nốt nhạc (các chữ cái từ 'a' đến 'z'), và mỗi ống organ có thể chơi một nốt nhạc nhất định.

Dwarf the Piper cần chế tạo một cây organ nhỏ gọn nhất, với số lượng ống ít nhất, sao cho vẫn có thể chơi toàn bộ bản quốc ca DNA.

Các ống organ được xếp thành một hàng ngang. Để chơi bản quốc ca, Dwarf có thể bắt đầu thổi từ bất kỳ ống nào. Sau khi thổi ống tạo ra nốt nhạc cần thiết, ông có thể đứng yên tại chỗ, di chuyển sang ống bên trái, hoặc di chuyển sang ống bên phải. Ông có thể kết thúc việc thổi tại bất kỳ vị trí nào.

Hãy giúp Dwarf chế tạo cây organ ngắn nhất (ít ống nhất) có thể chơi được bản quốc ca DNA.

## Input:

- Dòng đầu tiên chứa số nguyên N — số lượng nốt nhạc trong bản quốc ca DNA.
- Dòng thứ hai chứa chuỗi DNA gồm N chữ cái thường.

## Giới hạn:

- $1 \leq N \leq 500000$ .

### Output:

- Dòng đầu tiên in ra số nguyên K — số lượng ống ít nhất cần có.
- Dòng thứ hai in ra một chuỗi gồm K chữ cái thường — mô tả các nốt nhạc của cây organ (theo thứ tự từ trái sang phải). Nếu có nhiều cách sắp xếp thỏa mãn, in ra bất kỳ cách nào.

### Ví dụ:

Input:

14

ecerrcwrwcwror

Output:

7

cercwro

### Tóm tắt đề bài:

- Bạn được cho một chuỗi các nốt nhạc (chữ cái a-z).
- Cần tìm ra chuỗi ngắn nhất mô tả các nốt của cây organ (các ống xếp thành hàng ngang).
- Bắt đầu từ bất kỳ ống nào, có thể di chuyển trái/phải/đứng yên để thổi các nốt theo đúng thứ tự chuỗi ban đầu.
- Yêu cầu: In ra số lượng ống ít nhất và chuỗi nốt của cây organ.

### Dịch đề bài:

## Bài B. Các chữ số bị cấm

**Giới hạn thời gian:** 1 giây

**Giới hạn bộ nhớ:** 1024 MB

Khi cuối năm đến gần, quan tòa của làng người lùn quyết định rằng ngôi làng của họ sẽ chuyển sang sử dụng một hệ thống số theo vị trí mới với cơ số  $N$ .

Đối với các cơ số lớn hơn 10, người lùn sử dụng các chữ số từ 0 đến 9, sau đó là các chữ cái in hoa từ A đến Z để biểu diễn các chữ số tiếp theo. Tuy nhiên, vấn đề không nằm ở quan tòa, mà là ở các bô lão trong làng. Sau khi tham khảo các cuốn sách cổ, họ quyết định rằng một số chữ số sẽ bị cấm. Đáng chú ý là chữ số 0 không bị cấm — vì người lùn coi nó là biểu tượng thiêng liêng của sự cân bằng.

Hệ thống đánh số là nền tảng của xã hội trật tự người lùn: các ngôi nhà trong làng được đánh số bằng các số nguyên dương nhỏ nhất khác nhau (dùng hệ cơ số  $N$  và không chứa các chữ số bị cấm).

Quan tòa muốn biết số nhà lớn nhất sẽ được sử dụng trong làng.

### Input:

- Dòng đầu tiên chứa số nguyên  $T$  — số lượng test case (số làng).
- Mỗi test case gồm hai dòng:
  - Dòng đầu tiên chứa ba số nguyên  $N, K, M$ :
    - $N$ : cơ số của hệ thống số ( $2 \leq N \leq 36$ ).
    - $K$ : số lượng nhà trong làng ( $1 \leq K \leq 10^{18}$ ).
    - $M$ : số lượng chữ số bị cấm ( $0 \leq M \leq N - 2$ ).
  - Dòng thứ hai chứa  $M$  chữ số bị cấm (theo hệ cơ số  $N$ ), phân tách bởi dấu cách. Nếu  $M = 0$  thì dòng này trống.

### Output:

- Với mỗi test case, in ra số nhà lớn nhất (theo hệ cơ số  $N$ , không có chữ số bị cấm).

## Ví dụ:

### Input:

```
6
3 5 1
1
9 7 1
7
2 100 0
7 123456789 3
3 6 1
16 123456789 3
1 A F
36 123456789 5
A X Z 2 4
```

### Output:

```
202
8
1100100
25224550520222
2E878582
6C0503
```

## Tóm tắt đề bài:

- Có **T** làng, mỗi làng có:
  - Hệ thống đánh số nhà theo cơ số **N**.
  - Tổng cộng **K** ngôi nhà (đánh số bằng các số nguyên dương nhỏ nhất, không lặp lại).
  - Một số chữ số bị cấm (trừ số 0).
- Yêu cầu: Tìm **số nhà lớn nhất** (dạng chữ số cơ số N) trong danh sách **K** số hợp lệ.

## Ý chính:

- Dãy số hợp lệ là các số nguyên dương nhỏ nhất theo thứ tự, không chứa chữ số bị cấm.
- In ra số lớn nhất trong các số đó, viết dưới dạng cơ số N.

## Dịch đề bài:

### Bài C. Những tấm bảng tên CERC

**Giới hạn thời gian:** 1 giây

**Giới hạn bộ nhớ:** 1024 MB

Sự kiện CERC 2024 tại Wroclaw đang đến gần và tất cả mọi người đều đang góp sức chuẩn bị, kể cả những người lùn ở Wroclaw! Là thành viên ban tổ chức, mỗi người lùn được phát một tấm bảng tên để người khác dễ dàng nhận diện họ.

Người lùn rất sáng tạo, mỗi người đều tự nghĩ ra một **biệt danh độc đáo** để ghi trên bảng tên của mình. Một biệt danh được coi là **hợp lệ** nếu **bốn ký tự đầu tiên của biệt danh trùng chính xác (phân biệt hoa thường)** với **bốn ký tự đầu tiên của tên người lùn đó**.

Ví dụ:

- Người lùn tên **Mathew** có thể dùng bảng tên **Mathy**,
- Nhưng không thể dùng **Matty** hoặc **MATHY** (do phân biệt hoa thường).

Tuy nhiên, **Sloppy the Dwarf** lại là người in bảng tên, và vì anh ấy rất cầu thả nên đã không ghi chú lại bảng tên nào thuộc về ai. Tệ hơn, có thể một số bảng tên còn in sai biệt danh!

Nhiệm vụ của bạn là giúp Sloppy xác định bảng tên nào thuộc về ai:

- Bạn sẽ được cung cấp danh sách **tên của tất cả người lùn** và danh sách **biệt danh in trên bảng tên**.

- Hãy xác định xem có thể gán mỗi bảng tên cho một người lùn sao cho biệt danh hợp lệ với tên người lùn đó hay không.
- Nếu có, hãy in ra cách gán.

### Input:

- Dòng đầu tiên là số nguyên **N** — số lượng người lùn.
- **N dòng tiếp theo** chứa tên các người lùn (chữ cái hoa và thường, độ dài từ 4 đến 400000).
- **N dòng tiếp theo** chứa biệt danh in trên bảng tên (chữ cái hoa và thường, độ dài từ 4 đến 400000).
- Giới hạn:
  - $1 \leq N \leq 100000$ .
  - Tổng độ dài tất cả các tên và biệt danh không vượt quá 400000.
  - Không có tên hoặc biệt danh nào bị lặp lại.

### Output:

- Nếu không thể gán đúng bảng tên, in **NO**.
- Nếu có thể, in **YES** và N dòng tiếp theo, mỗi dòng gồm **tên người lùn** và **biệt danh** tương ứng, cách nhau bởi dấu cách. Nếu có nhiều cách, in ra bất kỳ cách nào.

### Ví dụ:

Input :

4

Slopy

Mathy

Thinky  
Cody  
Thinky  
Math  
Slopppy  
Codythesecond

Output:

YES

Cody Codythesecond  
Mathy Math  
Slopy Slopppy  
Thinky Thinky

Input:

3

Writy  
Buggy  
Solvvy  
Bogg  
Write  
Solvvy

Output:

NO

## Tóm tắt đề bài:

- Có **N người lùn**, mỗi người có **tên** và một **biệt danh in trên bảng tên**.
- Một **biệt danh hợp lệ** nếu **4 ký tự đầu của biệt danh** trùng chính xác với **4 ký tự đầu của tên người lùn** (phân biệt chữ hoa chữ thường).
- Hãy kiểm tra xem có thể gán mỗi biệt danh cho đúng người lùn hay không. Nếu có, in ra cách gán.

- Nếu không thể, in **NO**.

## Dịch đề bài:

### Bài D. Các ước hoán vị

**Giới hạn thời gian:** 4 giây

**Giới hạn bộ nhớ:** 1024 MB

Ori Badpun, một người lùn "biết tuốt", thách đố bạn với một câu đố:

- **Algorist** (người giỏi giải thuật), đúng không? Am hiểu về thuật toán chia để trị (divide and conquer), hay là **conquer and divide**?

Ông ta định nghĩa một khái niệm mới:

- Một **ước hoán vị** của số nguyên dương **N** là **một ước số thực sự của N** (tức là **ước số nhỏ hơn N**), **có các chữ số là hoán vị** của các chữ số của **N** (không cho phép số có số 0 đứng đầu).

Hiểu rồi chứ? Bây giờ, ông ta sẽ kiểm tra kỹ năng của bạn qua **T** thử thách.

- Mỗi thử thách, bạn được cho một số nguyên dương **N**, và bạn cần xác định có bao nhiêu **ước hoán vị** của **N**.

### Input:

- Dòng đầu tiên là số nguyên **T** — số lượng test case.
- Mỗi dòng trong **T dòng tiếp theo** chứa một số nguyên dương **N**.

### Giới hạn:

- $1 \leq T \leq 100000$
- $1 \leq N \leq 10^{18}$

### Output:



- Với mỗi số **N**, in ra số lượng **ước hoán vị** của nó.
- 

### Ví dụ:

Input :

4  
7  
31  
90  
370521

Output :

0  
0  
0  
1

### Giải thích:

- **370521** có một ước hoán vị là **123507** (vì **370521** chia hết cho **123507**).
- 

### Tóm tắt đề bài:

- Cho **T** số nguyên dương **N**.
- Với mỗi số **N**, tìm **số lượng ước số thực sự** của **N** mà các chữ số của ước đó là **hoán vị** của **các chữ số của N** (không cho phép chữ số 0 đứng đầu).
- In ra đáp án cho từng test case.

## Dịch đề bài:

### Bài E. Biểu thức

**Giới hạn thời gian:** 1 giây

**Giới hạn bộ nhớ:** 1024 MB

Kỹ năng chế tạo máy móc huyền thoại của người lùn cho phép họ tạo ra những cỗ máy tính cơ khí **trước cả cỗ máy phân tích của Babbage**. Tuy nhiên, gần đây trào lưu **tối giản** (minimalism) trở nên phổ biến, dẫn đến đề xuất thay thế các cỗ máy phức tạp bằng những máy đơn giản hơn, với tập hợp lệnh rút gọn. Những cỗ máy mới này sẽ nhỏ gọn hơn, rẻ hơn, và có thể vận hành nhanh hơn đáng kể.

Vấn đề là người lùn **chưa xác định được tập hợp phép toán tối thiểu** nào đủ để thực hiện tất cả các phép tính quan trọng đối với họ. Để giải quyết điều này, họ quyết định thực hiện hàng loạt thử nghiệm.

**Nhiệm vụ của bạn** là giải một số bài toán như vậy:

- Với một **biểu thức** được cho (gồm các phép toán nhị phân ``min``, ``max``, ``<=``, ``<`` và các biến là 10 chữ cái đầu bảng chữ cái tiếng Anh: ``a`` đến ``j``), hãy **chuyển đổi biểu thức** thành biểu thức tương đương nhưng **chỉ dùng hai phép toán** ``min`` và ``<=``, hoặc kết luận là **không thể**.

### Định nghĩa biểu thức:

- var** := ``a`` | ``b`` | ... | ``j``
- op** := `expr`<=`expr` | `expr`<`expr` | ``min`expr expr` | ``max`expr expr`
- expr** := var | (op)

### Hai biểu thức được coi là tương đương nếu:

- Tập biến** trong hai biểu thức giống nhau (gọi là **S**).
- Với **mọi cách gán giá trị** cho các biến trong **S** (các giá trị 0 hoặc 1), hai biểu thức đánh giá ra **kết quả giống nhau**.

### Input:

- Một dòng duy nhất chứa biểu thức theo cú pháp trên, với tối đa **200 phép toán nhị phân**.

### Output:

- Nếu **tồn tại biểu thức tương đương** chỉ dùng ``min`` và ``<=``, in ra:
  - Dòng 1: **YES**
  - Dòng 2: Biểu thức tương đương (tối đa **40000 phép toán nhị phân**).
- Nếu **không tồn tại**, in ra **NO**.

(Lưu ý: Nếu có cách biểu diễn tương đương, luôn tồn tại biểu thức có tối đa 40000 phép toán).

### Ví dụ:

Input:

`h`

Output:

`YES`

`h`

Input:

`((max a a) <= b)`

Output:

`YES`

`(a <= b)`

Input:

`((max a b) < (min a b))`

Output:

NO

## Tóm tắt đề bài:

- Bạn được cho một **biểu thức logic** gồm các phép toán: ``min``, ``max``, ``<=``, ``<`` và các biến ``a`` đến ``j``.
- Mục tiêu: **Biến đổi** biểu thức thành **biểu thức tương đương** (về giá trị với mọi cách gán 0/1 cho các biến) nhưng **chỉ dùng phép** ``min`` và ``<=``, hoặc kết luận **không thể**.
- In **YES** và biểu thức mới nếu làm được, hoặc **NO** nếu không thể.

## Dịch đề bài:

### Bài F. Căn hộ

**Giới hạn thời gian:** 2 giây

**Giới hạn bộ nhớ:** 1024 MB

Thành phố ngầm của người lùn đang đối mặt với vấn đề về nhà ở. Dwarf the Builder (người lùn xây dựng) đề xuất một ý tưởng táo bạo:

- Họ sẽ tận dụng **con mương Vực Thảm Vĩ Đại** (Great Divide), một cái mương cổ xưa, có vẻ **vô tận và thẳng tắp**.
- Họ sẽ **ném các viên gạch vào mương**. Những khoảng không gian được bao quanh hoàn toàn bởi gạch và mương sẽ trở thành các **căn hộ mới cho người lùn**!

### Mô tả chi tiết mương và gạch:

- Mương có **chiều rộng cố định, rất sâu, và vô hạn** về hai phía trái/phải.
- **Mỗi viên gạch** có **chiều rộng bằng chiều rộng mương**, nhưng **chiều dài và chiều cao** có thể khác nhau.
- Khi ném gạch vào mương, viên gạch sẽ **rơi thẳng đứng** cho đến khi:
  - Chạm **đáy mương** (ban đầu phẳng).
  - Hoặc chạm vào **viên gạch đã ném trước đó**.

### Quy tắc rơi của gạch:

- **Cạnh chạm cạnh hoặc mép chạm mép** giữa các viên gạch **không cản trở** việc rơi (gạch có thể trượt dọc theo các viên trước đó).
- Gạch **không thay đổi hướng** khi rơi và sau khi dừng.

**Mương được đánh dấu** tại mỗi vị trí **nguyên dương**, tính từ **gốc tọa độ 0**.

### Khái niệm căn hộ (flat):

- Một **khoảng trống** (không có gạch) có **thể tích khác 0**, được **bao quanh hoàn toàn** bởi các viên gạch hoặc tường mương, được tính là **một căn hộ**.
- Hai căn hộ được coi là **riêng biệt** nếu **không có đường di chuyển giữa chúng** (người lùn có kích thước, nên căn hộ chỉ chạm góc thì vẫn được coi là tách biệt).

### Yêu cầu:

- Hãy giúp Dwarf the Builder tính **số lượng căn hộ** được tạo ra sau khi ném tất cả các viên gạch vào mương.

### Giản lược bài toán:

- Xem mương và gạch trong **2D** (bỏ qua độ sâu), chỉ quan tâm đến chiều ngang và chiều cao.

### Input:

- Dòng đầu tiên chứa số nguyên **N** — số lượng viên gạch.
- N dòng tiếp theo**, mỗi dòng chứa 3 số nguyên **l r h**:
  - l** và **r**: tọa độ mép trái và mép phải của viên gạch (**chiều dài viên gạch** là  $r - l$ ).
  - h**: chiều cao viên gạch.

### Giới hạn:

- $1 \leq N \leq 200000$
  - $0 \leq l < r \leq 200000$
  - $1 \leq h \leq 10^9$
- 

### Output:

- In ra **một số nguyên không âm** — số lượng căn hộ tạo ra sau khi tất cả các viên gạch đã rơi xuống.
- 

### Tóm tắt đề bài:

- Có một **cái mương vô tận** (theo chiều ngang), có thể **ném các viên gạch** (có độ dài và chiều cao khác nhau) vào đó.
- Gạch **rơi thẳng đứng, trượt dọc** qua các viên gạch khác nhưng **không thay đổi hướng**.
- Sau khi ném hết **N viên gạch**, hãy đếm **bao nhiêu căn hộ (flats)** được tạo ra.

- Một căn hộ là **khoảng trống** được **bao quanh hoàn toàn** bởi gạch và tường mương, **không di chuyển được** sang các khoảng trống khác nếu chỉ chạm góc.

## Dịch đề bài:

### Bài G. Cửa hàng tạp hóa

**Giới hạn thời gian:** 3 giây

**Giới hạn bộ nhớ:** 1024 MB

Trong thành phố ngầm của người lùn có  **$N$  ngôi nhà**, được đánh số từ **1 đến  $N$**  theo **chiều kim đồng hồ**. Các ngôi nhà được nối với nhau bằng một **con đường tròn hai chiều**. Khoảng cách giữa từng cặp **hai ngôi nhà liên tiếp** là các số nguyên dương đã biết.

Người lùn rất thích **thăm hàng xóm**:

- Một người lùn sống ở **nhà  $i$**  sẽ thăm người lùn ở:
  - **Nhà  $i+1$**  (hoặc **nhà 1** nếu  **$i = N$** )
  - **Nhà  $i-1$**  (hoặc **nhà  $N$**  nếu  **$i = 1$** ).

Tuy nhiên, người lùn **không thích đến tay không**:

- Trước khi đến nhà hàng xóm, họ **phải ghé một cửa hàng tạp hóa** để mua đồ.
- Cửa hàng tạp hóa có thể **nằm xa hơn** hoặc **ngược hướng** so với nhà hàng xóm, và người lùn có thể **quay đầu lại** tại bất cứ điểm nào trên đường.

**Dwarf the Planner** (người quy hoạch) muốn **hiện đại hóa thành phố** bằng cách:

- **Phá bỏ toàn bộ cửa hàng tạp hóa cũ.**
- **Xây mới  $K$  cửa hàng tạp hóa**, sao cho **khoảng cách lớn nhất** mà một người lùn phải đi khi **thăm hàng xóm** và **ghé cửa hàng tạp hóa trên đường** là **nhỏ nhất có thể**.

### Lưu ý:

- Cửa hàng mới có thể được **xây ở bất kỳ vị trí nào trên con đường tròn**, không cần trùng với vị trí nhà hay khoảng cách nguyên.

### Nhiệm vụ:

- Tìm cách bố trí **K cửa hàng tạp hóa** sao cho **khoảng cách tối đa** mà **bất kỳ người lùn nào** phải đi khi **thăm hàng xóm** và **ghé cửa hàng** là **nhỏ nhất có thể**.
  - In ra **khoảng cách tối đa tối thiểu** đó.
- 

### Input:

- Dòng đầu tiên chứa hai số nguyên **N** và **K** — số lượng nhà và số cửa hàng tạp hóa cần xây.
- Dòng thứ hai chứa **N số nguyên d1, d2, ..., dN** — khoảng cách theo chiều kim đồng hồ giữa các nhà liên tiếp.

### Giới hạn:

- $2 \leq N \leq 500000$
  - $1 \leq K \leq N$
  - $1 \leq d_i \leq 10^9$
- 

### Output:

- In ra **một số nguyên** — khoảng cách tối đa tối thiểu mà người lùn phải đi (tối ưu).



### Ví dụ:

Input:

3 1

2 4 5

Output:

6

Input:

5 3

2 6 4 1 5

Output:

6

### Tóm tắt đề bài:

- Có **N ngôi nhà** xếp thành **vòng tròn**, khoảng cách giữa từng cặp nhà liên tiếp đã biết.
- Mỗi người lùn thăm **2 nhà hàng xóm** (một bên trái, một bên phải) và **phải ghé một cửa hàng tạp hóa** trên đường đi.
- **Xây K cửa hàng tạp hóa** sao cho **khoảng cách xa nhất** mà **bất kỳ người lùn nào** phải đi là **nhỏ nhất có thể**.
- Tìm và in ra **khoảng cách tối đa tối thiểu** đó.

### Dịch đề bài:

## Bài H. Sắc độ

**Giới hạn thời gian:** 1 giây

**Giới hạn bộ nhớ:** 1024 MB

**Dwarf the Painter** (người lùn họa sĩ), bậc thầy về màu sắc ma thuật, có **N thùng sơn**, mỗi thùng chứa **một màu khác nhau**.

- Mỗi khi một **tập con** các màu được **phối lên cùng một vùng** (diện tích khác 0) của tấm vải trắng, nó sẽ tạo ra **một sắc độ (hue)** nhất định.
- Mỗi **tập con màu khác nhau** sẽ tạo ra **một sắc độ khác nhau**.

**Người lùn họa sĩ** được **thử thách** bởi các bô lão trong làng:

- Phải tạo ra **một bức tranh chứa tất cả các sắc độ có thể có** (tổng cộng  $2^N - 1$  sắc độ — loại trừ sắc độ trắng khi **không có màu nào** được sơn lên).

Lười biếng như **Dwarf the Coach Potato**, họa sĩ chỉ vẽ **N hình tròn**, mỗi hình tròn chứa **một màu**.

- Khi trình bày bức tranh, các bô lão **bị choáng ngợp** bởi vẻ đẹp, nhưng họ **không chắc rằng tất cả  $2^N - 1$  sắc độ** có được tạo ra hay không.

**Nhiệm vụ của bạn:**

- Kiểm tra xem **bức tranh** có chứa **đầy đủ tất cả sắc độ** không.

**Input:**

- Dòng đầu tiên chứa số nguyên **T** — số lượng test case.
- Mỗi test case gồm:
  - Dòng đầu tiên chứa số nguyên **N** — số lượng hình tròn (và màu).
  - N dòng tiếp theo**, mỗi dòng chứa ba số nguyên **xi yi ri** — tọa độ tâm và bán kính của hình tròn thứ i.

**Giới hạn:**

- $1 \leq T \leq 200$
- $1 \leq N \leq 200$
- $-1000 \leq x_i, y_i \leq 1000$
- $1 \leq r_i \leq 1000$
- **Tổng N của tất cả các test case** không vượt quá 200.
- **Không có ba hình tròn nào giao nhau tại cùng một điểm.**

### Output:

- Nếu **bức tranh chứa đủ tất cả các sắc độ ( $2^N - 1$ )**, in ra **YES**.
- Nếu **thiếu sắc độ**, in ra:
  - Dòng đầu tiên: **NO**
  - Dòng thứ hai: Một sắc độ bị thiếu, được mô tả bằng **N số 0 hoặc 1** (1 nếu màu đó có mặt trong sắc độ, 0 nếu không).

### Ghi chú:

- Nếu thiếu nhiều sắc độ, bạn chỉ cần in ra **một sắc độ bất kỳ bị thiếu**.
- Một sắc độ được tính là tồn tại nếu **có diện tích dương** (không phải giao nhau tại một điểm).

### Ví dụ:

Input:

5

2

0 0 1

1 0 1

```
2
0 0 1
2 0 1
3
-1 0 2
1 0 2
0 1 2
5
0 0 4
5 -4 4
10 0 4
15 -4 4
20 0 4
5
0 0 7
0 3 4
3 0 4
0 -3 4
-3 0 4

Output:
YES
NO
1 1
YES
NO
1 0 1
NO
0 1 0 0 0
```

## Tóm tắt đề bài:

- Có **N hình tròn**, mỗi hình tròn chứa **một màu khác nhau**.
- Khi **các màu giao nhau**, chúng tạo ra **sắc độ** (một tập con màu).

- Mỗi **tập con màu** ứng với một **sắc độ duy nhất**, tổng cộng có  $2^N - 1$  sắc độ cần xuất hiện.
- Yêu cầu: Kiểm tra xem **bức tranh có đủ tất cả sắc độ** hay không.
- Nếu thiếu sắc độ, in ra **NO** và một **sắc độ bị thiếu** (dạng nhị phân). Nếu đủ, in ra **YES**.

## Dịch đề bài:

### Bài 1. Illuminati

**Giới hạn thời gian:** 1 giây

**Giới hạn bộ nhớ:** 1024 MB

**Dwarf the Illuminati** đang chuẩn bị cho **màn trình diễn ánh sáng đêm giao thừa**.

- Sẽ có **N cây nến** được đặt thành **một hàng thẳng**.
- Mỗi cây nến có thể **bật sáng (1)** hoặc **tắt (0)**.

Buổi trình diễn sẽ gồm **K vòng**.

- **Giữa các vòng, Illuminati sẽ di chuyển vị trí các cây nến.**
- Để dễ nhớ, **Illuminati** muốn **di chuyển các cây nến theo cùng một cách** sau mỗi vòng.

### Cụ thể:

- Illuminati sẽ tạo ra một **hoán vị P** gồm **N phần tử**.
- Sau mỗi vòng, cây nến ở vị trí **i** sẽ được chuyển đến vị trí **P(i)**.

### Nhiệm vụ:

- Illuminati đã thiết kế xong toàn bộ buổi diễn (tức là biết trạng thái sáng/tắt của các cây nến ở từng vòng).

- Bạn cần kiểm tra xem **có tồn tại hoán vị P** nào thỏa mãn việc **sắp xếp lại vị trí các cây nến** theo đúng thiết kế không.
  - Nếu có nhiều hoán vị, hãy in ra **hoán vị nhỏ nhất theo thứ tự từ điển**.
- 

### Input:

- Dòng đầu tiên chứa hai số nguyên **N** và **K** — số lượng cây nến và số vòng.
- **K dòng tiếp theo**, mỗi dòng là một **chuỗi nhị phân** dài **N**:
  - **1**: nến sáng.
  - **0**: nến tắt.
  - Dòng thứ *i* là trạng thái nến trong **vòng thứ i**.

### Giới hạn:

- $1 \leq N \cdot K \leq 10^6$
  - $1 \leq N, K \leq 10^5$
- 

### Output:

- Nếu tồn tại **hoán vị P**, in ra:
    - Dòng 1: **YES**
    - Dòng 2: Hoán vị P (gồm **N số nguyên**, đánh số từ **1**).
  - Nếu **không tồn tại**, in ra **NO**.
-

## Ví dụ:

Input:

3 3

100

010

001

Output:

YES

2 3 1

Input:

4 2

0011

0110

Output:

YES

1 4 2 3

## Tóm tắt đề bài:

- Có **N** cây nến và **K** vòng trình diễn.
- Ở mỗi vòng, biết được **trạng thái sáng/tắt** của các cây nến.
- Sau mỗi vòng, **di chuyển nến** theo **một hoán vị P** duy nhất (áp dụng lặp lại giữa các vòng).
- Nhiệm vụ: Tìm xem **có tồn tại hoán vị P** nào thỏa mãn việc **sắp xếp lại vị trí nến** để khớp với thiết kế không.
- Nếu có, in ra **hoán vị nhỏ nhất từ điển**. Nếu không, in **NO**.

## Dịch đề bài:

### Bài J. Chỉ là đào vàng thôi mà

**Giới hạn thời gian:** 1.5 giây

**Giới hạn bộ nhớ:** 1024 MB

Người lùn đã phát hiện ra **hiều mỏ vàng** mới ở **tầng đáy của mỏ Deep Mine**.

- Vị trí các mỏ vàng (trên mặt phẳng) được cho dưới dạng một **đa tập điểm**  $P = \{p_1, \dots, p_m\}$ ,
- Mỗi điểm có một **giá trị ước tính** tương ứng là **vi**.

Tuy nhiên, việc khai thác **không dễ dàng** vì **quá nhiều khoan đào** ở một số khu vực có thể gây **mất ổn định** và **sụp đổ toàn bộ mỏ**.

- Các **hiệp hội thợ mỏ, địa chất**, và **kỹ sư** đã xác định các **khu vực nguy hiểm**:
  - Chúng tạo thành một **đa tập hình tròn** trên mặt phẳng  $C = \{c_1, \dots, c_n\}$ .
  - Mỗi vòng tròn **ci** có **sức chứa tối đa fi** — tức là **số lượng mỏ vàng tối đa** có thể khai thác trong vòng tròn đó mà **không gây nguy hiểm**.

### Nhiệm vụ của bạn:

- Chọn ra **một tập con A** từ các điểm  $P$  sao cho:
  - **Tổng giá trị** các điểm trong  $A$  là **lớn nhất có thể**.
  - Với mỗi vòng tròn **ci**, **số lượng điểm** trong  $A$  nằm trong  $c_i$  **không vượt quá fi**.

### Lưu ý đặc biệt:

- Các vòng tròn **C** được chia thành **hai nhóm rời nhau C1 và C2**.
- Mỗi nhóm là một **gia đình laminar**:



- Tức là trong cùng một nhóm, **hoặc hai vòng tròn không giao nhau, hoặc một vòng tròn nằm hoàn toàn trong vòng tròn kia.**
  - **Không có hai vòng tròn trong cùng nhóm nào giao nhau.**
- 

### Input:

- Dòng đầu tiên chứa hai số nguyên **N** và **M** — số lượng vòng tròn và số lượng điểm.
- **N dòng tiếp theo**, mỗi dòng chứa 4 số nguyên **xi yi ri fi**:
  - **(xi, yi)**: tâm vòng tròn thứ i.
  - **ri**: bán kính vòng tròn.
  - **fi**: sức chứa tối đa của vòng tròn.
- **M dòng tiếp theo**, mỗi dòng chứa 3 số nguyên **xi yi vi**:
  - **(xi, yi)**: tọa độ điểm thứ i (mỏ vàng).
  - **vi**: giá trị ước tính của mỏ vàng.

### Giới hạn:

- $1 \leq N, M \leq 300$
  - $-10^9 \leq xi, yi \leq 10^9$
  - $1 \leq ri, vi \leq 10^9$
  - $1 \leq fi \leq 300$
- 

### Output:

- Dòng 1: Tổng **giá trị lớn nhất** có thể thu được từ các điểm chọn.

- Dòng 2: **Số lượng điểm** được chọn.
- Dòng 3: **Chỉ số các điểm** được chọn (theo thứ tự xuất hiện trong input, bắt đầu từ 1), có thể in theo bất kỳ thứ tự nào.

Nếu có nhiều cách chọn, in ra **bất kỳ một cách**.

### Ví dụ:

#### Input:

```
5 5
3 5 2 2
4 10 4 2
5 10 2 3
5 5 5 2
14 4 2 3
6 11 3
3 8 5
4 6 20
9 5 4
14 5 1
```

#### Output:

```
28
4
1 3 4 5
```

#### Input:

```
3 2
4 7 2 2
4 8 1 1
8 7 1 1
4 7 5
4 8 4
```

#### Output:

5  
1  
1

## Tóm tắt đề bài:

- Có **M mỏ vàng** (tọa độ và giá trị ước tính).
- Có **N khu vực nguy hiểm** (vòng tròn với sức chứa tối đa).
- Mỗi khu vực giới hạn **số lượng mỏ vàng tối đa** được khai thác trong đó.
- Chọn **tập con các mỏ vàng** để **tối đa hóa tổng giá trị** mà **không vượt quá sức chứa** của bất kỳ vòng tròn nào.
- In ra **tổng giá trị tối đa**, **số lượng mỏ vàng được chọn**, và **chỉ số các mỏ vàng** đó.

## Dịch đề bài:

### Bài K. Lễ hội của Nhà Vua

**Giới hạn thời gian:** 1.5 giây

**Giới hạn bộ nhớ:** 1024 MB

Cho lễ hội sắp tới, **Nhà Vua của người lùn** muốn **thắp sáng toàn bộ tòa thị chính** của vương quốc.

- Tòa thị chính là một **lưới vuông kích thước  $N \times N$** , gồm các **ô vuông nhỏ**.
- **Tất cả các ô** phải được **chiếu sáng**.

Tuy nhiên, để **tiết kiệm năng lượng** và **bảo tồn ma thuật** của các **đèn lồng**, nhà vua muốn **sử dụng ít đèn nhất có thể**.

- Nhà vua quyết định **chỉ đặt đèn trên đường chéo chính** (từ **trên trái** đến **dưới phải** của lưới).

### Đặc điểm của đèn lồng ma thuật:

- Khi đặt lên một ô, **đèn lồng** sẽ chiếu sáng:
  - **Toàn bộ hàng** chứa nó.
  - **Toàn bộ cột** chứa nó.
  - **Hai đường chéo** đi qua ô đó (gồm cả đường chéo chính và phụ).

Một số **đèn đã được đặt sẵn** bởi các kỹ sư người lùn, **không thể di chuyển**.

---

### Input:

- Một dòng duy nhất là một **chuỗi ký tự** mô tả các **đèn đã đặt sẵn** trên **đường chéo chính**:
  - **"."**: ô trống (chưa có đèn).
  - **"#"**: ô đã có đèn cố định.

### Giới hạn:

- Chuỗi có độ dài từ **1 đến 64 ký tự** (tương ứng với lưới **N x N**).
- 

### Output:

- Dòng 1: **Số lượng đèn tối thiểu** cần đặt thêm để chiếu sáng toàn bộ lưới.
- Dòng 2: **Chuỗi mô tả vị trí đèn** trên đường chéo chính sau khi đã đặt thêm (bao gồm cả đèn cố định ban đầu).

- Nếu có nhiều cách đặt, in ra **bất kỳ cách nào**.

### Ví dụ:

Input :

.....

Output :

4

###.#.

Input :

.#.#...

Output :

4

.###.#.

### Tóm tắt đề bài:

- Cho **một lưới  $N \times N$** , chỉ được **đặt đèn trên đường chéo chính**.
- **Mỗi đèn** chiếu sáng **hàng, cột**, và **hai đường chéo** đi qua nó.
- Một số **đèn đã được đặt sẵn** (không thể di chuyển).
- Yêu cầu: **Đặt thêm ít đèn nhất để chiếu sáng toàn bộ lưới**, và in ra **vị trí các đèn** sau khi đặt xong.

### Dịch đề bài:

## Bài L. Mê cung

**Giới hạn thời gian:** 2 giây

**Giới hạn bộ nhớ:** 1024 MB

**Dwarf the Puzzler** là người thiết kế các **câu đố mê cung** cho các người lùn khác.

- Một **mê cung** là một **lưới vuông  $N \times N$** , mỗi ô là **đường đi** hoặc **tường**.
- Trong một ô đường đi có đặt một **con cò**, con cò này có thể **quay mặt** về **một trong bốn hướng**: phải (R), lên (U), trái (L), hoặc xuống (D).

### Quy tắc di chuyển:

- Mỗi bước, người lùn **quay con cò** về một hướng mới, sau đó **di chuyển con cò** theo hướng đó.
- Con cò **không thể đi vào tường** hoặc **vượt qua tường**.
- Nếu con cò **ra khỏi mê cung**, câu đố được **giải xong**.

**Chiến thuật giải mê cung của Dwarf the Riddler** (đối thủ cũ của Puzzler):

- Ở mỗi bước:
  - Xem xét 4 ô **bên phải, trước mặt, bên trái, phía sau** (tương ứng với **hướng hiện tại** của con cò).
  - Ưu tiên chọn ô đầu tiên là đường đi** (theo thứ tự: phải, trước, trái, sau).
  - Quay con cò về hướng đó và di chuyển.**

### Nhiệm vụ:

- Kiểm tra xem với **chiến thuật này**, con cò **có thoát ra khỏi mê cung** hay **bị mắc kẹt**.
- Nếu **thoát ra**, in ra **số bước di chuyển** để thoát.
- Nếu **mãi không thoát ra**, in ra **-1**.

---

### Input:

- Dòng đầu tiên chứa hai số nguyên **N** và **Q** — kích thước mê cung và số truy vấn.
- **N dòng tiếp theo**: mô tả mê cung:
  - ".": đường đi.
  - "#": tường.
- **Q dòng tiếp theo**: mỗi truy vấn gồm:
  - **r, c, d** — vị trí bắt đầu (hàng r, cột c) và hướng ban đầu của con cờ:
    - **U**: lên.
    - **D**: xuống.
    - **L**: trái.
    - **R**: phải.

### Giới hạn:

- $1 \leq N \leq 1000$
  - $1 \leq Q \leq 100000$
  - $1 \leq r, c \leq N$
  - Con cờ **không bao giờ đặt vào tường**.
- 

### Output:

- Với mỗi truy vấn, in ra:
  - **Số bước** để thoát khỏi mê cung, hoặc

- **-1** nếu không bao giờ thoát được.

### Ví dụ:

Input :

10 10

#####

#.#.#.#...

#.....#.#

####.##..#

#....#####

#..#.#...#

#.#..#.#.#

#.#.##.#.#

#.#.....#

###.#####

2 10 U

2 10 L

4 8 U

10 4 U

8 7 U

8 9 U

6 2 U

5 2 U

3 5 D

7 4 R

Output :

1

11

7

61

-1

54

-1



## Tóm tắt đề bài:

- Cho **một mê cung  $N \times N$**  (gồm đường đi và tường).
- Một **con cò** bắt đầu ở **một vị trí đường đi** và quay về **một hướng ban đầu**.
- **Chiến thuật di chuyển**: ưu tiên đi **bên phải, trước mặt, trái, phía sau** (theo hướng hiện tại).
- Với **Q truy vấn**, xác định:
  - Con cò có **thoát ra khỏi mê cung** không?
  - Nếu có, in **số bước di chuyển**. Nếu không, in **-1**.