**ĐỀ BÀI NGÀY 30 – 10 - 2021**

**Câu 1. (Cắt xâu-Cutstr)**

Cho 2 xâu và . Ta định nghĩa xâu con của là xâu con gồm các kí tự liên tiếp trong . Gọi là số xâu con không gối lên nhau lớn nhất có thể lấy được từ xâu sao cho các xâu con này đều bằng . Ví dụ ta có xâu S = “aaaaa”, T = “aa”, khi đó

Ta định nghĩa hàm là số cách cắt từ xâu ra xâu con không gối lên nhau sao cho các xâu con này đều bằng . Trong ví dụ trên ta có:

: **|aa|**aaa, a**|aa|**aa, aa**|aa|**a, aaa**|aa|**

: |**aa**||**aa**|a, |**aa**|a|**aa**|, a|**aa**||**aa**|

Hãy tính .

**Input:**

* Dòng đầu tiên chứa xâu .
* Dòng thứ hai chứa xâu .

**Output:**

* Ghi ra một dòng duy nhất là kết quả của bài toán. Do kết quả có thể rất lớn nên bạn cần ghi ra phần dư trong phép chia kết quả cho

**Example:**

|  |  |
| --- | --- |
| CUTSTR.INP | CUTSTR.OUT |
| heknowsimeanityeahyouknowimeanityouknow  imeanit | 3 |
| Aaaaa  aa | 7 |

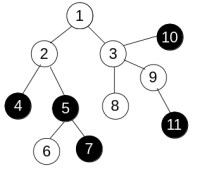
**Subtasks:**

* Subtask 1 (10%):
* Subtask 2 (30%):
* Subtask 3 (60%):

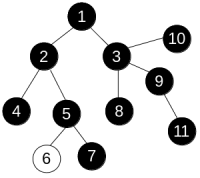
**Câu 2. (Cây một màu- Colortree)**

Cho một đồ thị cây gồm đỉnh, mỗi đỉnh được tô một trong hai màu, trắng hoặc đen. Trên đồ thị cây này ta định nghĩa một thao tác với là một đỉnh bất kì của cây, thao tác này có tác dụng đổi màu tất cả các đỉnh mà tất cả các đỉnh nằm trên đường đi ngắn nhất từ đến (kể cả ) có cùng một màu.

Ví dụ ta có đồ thị cây như sau:



Ta thực hiện thao tác với là một trong các đỉnh đều thu được đồ thị như sau:



Bạn hãy tìm cách sử dụng ít thao tác nhất sao cho tất cả các đỉnh của cây có cùng một màu. Với đồ thị minh họa ở trên số thao tác ít nhất cần sử dụng là 2, áp dụng lần lượt thao tác lên các đỉnh 3, 6.

**Input:**

* Dòng đầu tiên chứa số nguyên .
* Dòng thứ hai chứa số nguyên tương ứng là màu của các đỉnh, ứng với màu trắng, ứng với màu đen.
* Tiếp theo có dòng, mỗi dòng gồm 2 số nguyên miêu tả một cạnh của đồ thị.

**Output:**

* Ghi ra một dòng duy nhất là số thao tác ít nhất cần thực hiện để đưa toàn bộ các đỉnh của cây về cùng một màu.

**Example:**

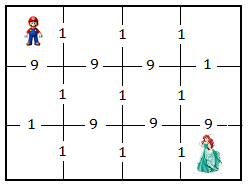
|  |  |
| --- | --- |
| COLORTREE.INP | COLORTREE.OUT |
| 11  0 0 0 1 1 0 1 0 0 1 1  1 2  1 3  2 4  2 5  5 6  5 7  3 8  3 9  3 10  9 11 | 2 |
| 4  0 0 0 0  1 2  2 3  3 4 | 0 |

**Subtasks:**

* Subtask 1 (20%):
* Subtask 2 (30%): .
* Subtask 3 (50%): Ràng buộc như trong đề bài.

**Câu 3. Giải cứu công chúa**

   Mụ phù thủy có một khu hầm hình chữ nhật gồm n x m căn hầm. Các căn hầm được đánh số từ dòng từ 1 đến n theo chiều từ trên xuống dưới, đánh số cột từ 1 đến m theo chiều từ trái qua phải. Giữa hai căn hầm sát nhau có cửa thông nhau mà phải mất một khoảng thời gian nào đó mới có thể mở cửa để đi từ căn hầm này sang căn hầm kia. Sau khi bắt cóc công chúa, mụ phù thủy giam nàng tại căn hầm cuối cùng [n,m] - dòng n cột m . Chàng thợ sửa ống nước Super Mario đang ở căn hầm đầu tiên [1, 1]. Bạn hãy tìm các căn hầm mà Mario phải đi qua để giải cứu công chúa một cách nhanh nhất.



**Dữ liệu nhập vào từ file MARIO.INP :**

- Dòng thứ nhất là hai số nguyên n và m cách nhau một khoảng trắng (1 ≤ n, m ≤ 700).

- Trong n dòng tiếp theo mỗi dòng gồm m-1 số nguyên aij là khoảng thời gian để mở cửa từ căn hầm [i, j] sang căn hầm [i, j+1].  (1 ≤ aij ≤ 105)

- Trong n-1 dòng tiếp theo mỗi dòng gồm m số nguyên bij là khoảng thời gian để mở cửa từ căn hầm [i, j] sang căn hầm [i +1, j].  (1 ≤ bij ≤ 105)

**Dữ liệu xuất ra file MARIO.OUT :**

- Là số nguyên xác định khoảng thời gian sớm nhất mà Mario giải cứu công chúa.

**Ví dụ**

|  |  |
| --- | --- |
| **MARIO.INP** | **MARIO.OUT** |
| 3 4 1 1 1 1 1 1 1 1 1 9 9 9 1 1 9 9 9 | 11 |

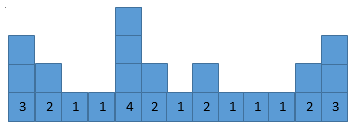
**Ràng buộc :**

* 30% số test có 1 ≤ n , m ≤ 10.
* 30% số test tiếp theo có 1 ≤ n , m ≤ 50.
* 40% số test còn lại có 1 ≤ n , m ≤ 700.

**Câu 4. Trò chơi xếp hình**

 Bon rất thích trò chơi điện tử đặc biệt là trò chơi xếp hình. Nhân ngày sinh nhật, Bon đòi cậu HA mua cho Bon một máy chơi điện tử, vì rất quý cháu nên HA mua cho Bon máy điện tử xếp hình để cậu bé chơi giải trí.

       Có máy điện tử, cứ đến bữa ăn là Bon mang ra chơi để có thể ăn cơm dễ dàng hơn. Hôm nay Bon thấy thấy trên màn hình đã có sẵn N thanh thẳng đứng xếp sát nhau đánh chỉ số từ 1 đến N, các thanh có độ cao là H1, H2, …, HN.



         Phía trên của màn hình xuất hiện lần lượt M thanh nằm ngang, mỗi thanh có độ dài lần lượt là L1, L2, …, LM. Các thanh xuất hiện được xác định tọa độ của mép bên trái thanh đó là vị trí P1, P2, …, PM. Khi một thanh rơi xuống và mép dưới thanh tiếp xúc với một cột nào đó thì mới xuất hiện thanh tiếp theo trên màn hình.

        Bạn hãy giúp Bon xác định xem độ cao của mỗi thanh ngang khi nó tiếp xúc với một cột nào đó trên màn hình.

**Dữ liệu nhập vào từ file GAMES.INP :**

- Dòng 1 chứa hai số N và M (1 ≤ N, M ≤ 105)  
- Dòng 2 chứa N số nguyên không âm H1, H2, …, HN (1 ≤ Hi ≤ 109)  
- M dòng tiếp theo mỗi dòng chứa 2 số Pi và Li (1 ≤ Pi, Li ≤ 105) – là vị trí xuất hiện và chiều dài của thanh thứ i.

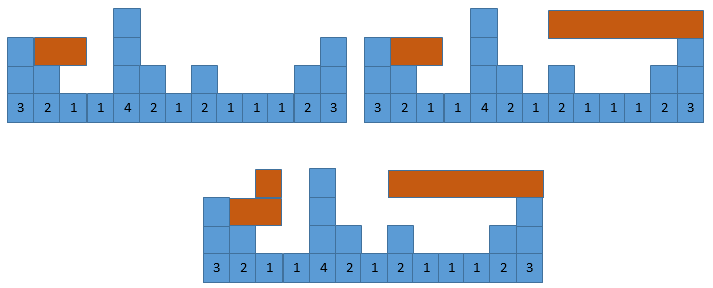
**Dữ liệu xuất ra file GAMES.OUT :**

- M dòng, mỗi dòng i là độ cao của thanh ngang thứ i khi rơi xuống.

### Ví dụ

|  |  |
| --- | --- |
| **GAMES.INP** | **GAMES.OUT** |
| 13 3 3 2 1 1 4 2 1 2 1 1 1 2 3 2 2 8 6 3 1 | 3 4 4 |

Giải thích:



**Ràng buộc :**

* 30% số test có 1 ≤ N, M ≤ 1000, 1 ≤ Pi, Li ≤ 1000.
* 30% số test có 1 ≤ N, M ≤ 105, 1 ≤ Pi, Li ≤ 105. Thỏa mãn Pi + Li  ≤ Pj (Với 1 ≤ i < j≤ M).
* 40% số test có 1 ≤ N, M ≤ 105, 1 ≤ Pi, Li ≤ 105