|  |  |
| --- | --- |
|  |  |

# Môn thi: Tin học

Thời gian làm bài:***180*** *phút, không kể thời gian giao đề*

Ngày thi thứ nhất

#### Tổng quan bài thi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên bài** | **File nguồn** | **File Input** | **File Output** | **Thời gian** |
| Hội nghị khoa học | conference.\* | conference.in | conference. out | 5 giây |
| Mua hàng giảm giá | discount.\* | discount.in | discount.out | 1 giây |
| Nhảy parkour | parkour.\* | parkour.in | parkour.out | 2 giây |

**Dấu \* là Pas hoặc Cpp tương ứng với ngôn ngữ lập trình Pascal hoặc C++**

**Hãy lập trình giải các bài toán sau:**

**Câu I.** (*6 điểm*) **Hội nghị khoa học**

Một hội nghị khoa học sắp sửa diễn ra tại thành phố X. ***N*** người được đánh nhãn từ 1 tới ***N*** sẽ tham dự hội nghị. ***K*** trong số họ có nhãn từ 1 tới ***K*** là các nhà khoa học sẽ trình bày phát minh của mình trong hội nghị lần này. ***N*** - ***K*** người còn lại là nhà báo, và họ sẽ viết báo về tất cả các phát minh mà họ được nghe tới. Hội nghị sẽ được diễn ra trong ***M*** ngày liên tiếp.

Tất nhiên, các nhà khoa học đều muốn phát minh của mình được công chúng biết tới, do đó họ muốn ít nhất một nhà báo sẽ biết được về phát minh của họ. Mỗi một trong ***M*** ngày diễn ra hội nghị, đúng 2 trong số ***N*** người tham dự hội nghị sẽ có một cuộc gặp gỡ với nhau, và họ sẽ trao đổi với nhau mọi phát minh mà họ đã được nghe kể/nghĩ ra **trước đó**. Ví dụ, Khi 2 người ***A*** và ***B*** gặp nhau, ***A*** biết mọi phát minh trong tập hợp các phát minh ***U***, ***B*** biết mọi phát minh trong tập hợp các phát minh ***V***, sau cuộc gặp gỡ cả ***A*** và ***B*** đều biết mọi phát minh trong tập hợp các phát minh ***U*** hợp ***V***.

Mỗi nhà khoa học sẽ chỉ nghĩ ra đúng một phát minh. Các nhà khoa học rất lười biếng nên họ sẽ chỉ nghĩ ra phát minh của mình vào ngày **muộn nhất** có thể, sao cho vẫn có ít nhất một **nhà báo** biết được về nó. Phát minh sẽ được nghĩ ra vào sáng sớm, trước lúc cuộc gặp gỡ diễn ra. Do đó nếu một nhà khoa học nghĩ ra phát minh của mình vào một ngày, nhà khoa học đó có thể kể về phát minh này trong cuộc gặp gỡ cùng ngày (nếu nhà khoa học này là 1 trong 2 người tham gia cuộc gặp gỡ đó).

**Yêu cầu:**

* Với mỗi nhà khoa học, tìm ngày muộn nhất mà nhà khoa học đó sẽ nghĩ ra phát minh của mình, sao cho vẫn có ít nhất một **nhà báo** nghe được về phát minh đó.
* Tìm tất cả các nhà báo được nghe kể về ít nhất một phát minh nào đó.
* Với mỗi nhà khoa học, tìm **nhà báo** đầu tiên được nghe kể về phát minh của nhà khoa học đó.

**Input:** đọc từ file **conference.in**

Dòng đầu tiên chứa số 3 nguyên ***N***, ***M***, ***K*** (1 <= ***K*** <= ***N*** <= 10^6, 1 <= ***M*** <= 10^6).

***M*** dòng sau, dòng thứ ***i*** (1 <= ***i*** <= ***M***) gồm 2 số ***x***, ***y*** (1 <= ***x***, ***y*** <= ***N***, ***x*** != ***y***) là nhãn của 2 người sẽ gặp nhau vào ngày thứ ***i***.

**Output:** ghi ra file **conference.out**

Dòng đầu tiên gồm ***K*** số, số thứ ***i*** là ngày muộn nhất mà nhà khoa học có nhãn ***i*** có thể nghĩ ra phát minh của mình và vẫn có ít nhất một nhà báo nghe được về nó. Nếu không có nhà báo nào có thể nghe được về phát minh của nhà khoa học ***i***, in ra -1.

Dòng thứ 2 gồm một số ***T*** + 1 số nguyên, số đầu tiên là ***T*** (0 <= ***T*** <= ***N*** - ***K***): số lượng nhà báo được nghe về ít nhất một phát minh nào đó, ***T*** số nguyên tiếp theo là nhãn theo thứ tự tăng dần của ***T*** nhà báo đó.

Dòng cuối cùng gồm ***K*** số, số thứ ***i*** là nhãn của nhà báo đầu tiên nghe được về phát minh của nhà khoa học có nhãn là ***i***. Nếu không có nhà báo nào có thể nghe được về phát minh của nhà khoa học ***i***, in ra -1.

**Subtask:**

*Subtask 1 (50%):* ***N, M*** <= 1000

*Subtask 2 (50%):* Không có ràng buộc gì thêm

**Ví dụ**

|  |  |  |
| --- | --- | --- |
| **conference.in** | **conference.out** | **Giải thích** |
| 6 6 3 6 2 5 2 1 2 3 1 2 4 2 5 | 3 6 -1 2 4 5 4 5 -1 | Ngày muộn nhất mà nhà khoa học có nhãn 1 có thể nghĩ ra phát minh của mình là ngày thứ 3.  Nếu nhà khoa học 1 nghĩ phát minh vào ngày thứ 4, chỉ có người có nhãn 3 (là một nhà khoa học, không phải nhà báo) nghe được về phát minh đó.  Nếu nhà khoa học 1 nghĩ ra phát minh vào ngày thứ 3, người có nhãn 2 sẽ được nghe về phát minh đó cùng ngày, và nhà báo có nhãn 4 sẽ được nghe về phát minh đó (từ người 2) vào ngày thứ 5. |

**Câu II.** (*7 điểm*) **Mua hàng giảm giá**

Cửa hàng gần nhà Nam hiện đang có chương trình giảm giá cuối năm nên Nam quyết định sẽ bỏ tiền ra để mua N món đồ ưa thích của mình. Để cho thuận tiện, N món đồ ưa thích của Nam sẽ được đánh số từ 1 tới N. Chính sách giảm giá của cửa hàng như sau:

* Nếu mua món đồ thứ i với **giá gốc**, người mua sẽ nhận được một phiếu giảm giá 50% cho các món đồ thứ i + 1, i + 2, …, i + k.
* Mỗi một món đồ chỉ được phép áp dụng tối đa một phiếu giảm giá.

**Yêu cầu:** Vì còn đang là học sinh nên Nam không dư giả là bao. Bạn hãy giúp Nam xác định xem số tiền ít nhất bỏ ra để mua được toàn bộ N đồ vật là bao nhiêu.

**Input:** đọc từ file **discount.in**

Dòng đầu tiên chứa hai số nguyên N và k (1 <= N, k <= 100000).

Dòng thứ hai chứa N số nguyên dương chẵn không quá 1 tỷ là giá gốc của N đồ vật.

**Output:** ghi ra file **discount.out**

In ra trên một dòng số tiền ít nhất có thể mua được toàn bộ N đồ vật.

**Subtask:**

*Subtask 1 (30%):* N, k <= 100, giá của các đồ vật không lớn hơn 100

*Subtask 2 (25%):* N, k <= 5000

*Subtask 3 (45%):* Không có ràng buộc gì thêm.

**Ví dụ**

|  |  |  |
| --- | --- | --- |
| **discount.in** | **discount.out** | **Giải thích** |
| 5 2 2 2 2 2 2 | 7 | Mua đồ vật thứ 1, 4 với giá gốc |
| 5 2 2 4 8 10 12 | 23 | Mua đồ vật thứ 1, 3 với giá gốc |

**Câu III** (*7 điểm*). **Nhảy parkour**

Rạp xiếc thành phố đang chuẩn bị tiết mục nhảy parkour để trình diễn nó trong buổi biểu diễn sắp tới.

Trong tiết mục này, sẽ có N chiếc cột được xếp thành một hàng. Các cột được đánh số từ 1 tới N từ trái qua phải. Cột thứ i (1 <= i <= N) có độ cao là H\_i.

M chuyên gia parkour sẽ nhảy trên N chiếc cột này. Các chuyên gia parkour sẽ được đánh số từ 1 tới M. Chuyên gia thứ i (1 <= i <= M) sẽ nhảy từ cột A\_i đến cột B\_i. Nếu A\_i < B\_i, chuyên gia thứ i sẽ xuất phát từ cột A\_i, liên tục nhảy sang cột kề ngay bên phải, cho đến khi đến cột B\_i. Nếu A\_i > B\_i, tương tự, chuyên gia thứ i sẽ liên tục nhảy sang bên trái.

Với mỗi chuyên gia parkour ta biết được giới hạn mà người đó có thể nhảy lên, nhảy xuống. Chuyên gia parkour i không thể nhảy lên quá U\_i đơn vị độ cao và không thể nhảy xuống quá D\_i đơn vị độ cao. Nghĩa là, chuyên gia i chỉ có thể nhảy từ cột với độ cao h1 đến cột với độ cao h2 khi và chỉ khi h1 - D\_i <= h2 <= h1 + U\_i.

Trong buổi tập luyện ngày hôm nay, chủ rạp xiếc phát hiện ra một điều: không phải chuyên gia parkour i nào cũng nhảy được từ cột A\_i đến cột B\_i ! Do đó, cần phải điều chỉnh độ cao của các cột để ai cũng có thể hoàn thành nhiệm vụ nhảy. Để khán giả không nhận ra việc điều chỉnh cột, chủ rạp xiếc muốn chêch lệch lớn nhất về độ cao giữa cột cũ và cột mới là nhỏ nhất có thể. Nghĩa là, gọi P\_i (1 <= i <= N) là độ cao mới của chiếc cột thứ i sau khi điều chỉnh, chủ rạp xiếc muốn *max*{ |P\_i - H\_i| } là nhỏ nhất có thể. **Lưu ý:** P\_i phải không âm và có thể là số thực.

**Yêu cầu:** Bạn hãy xác định và in ra giá trị nhỏ nhất của *max*{ |P\_i - H\_i| }.

**Input:** đọc từ file **parkour.in**

Dòng đầu tiên chứa số nguyên T (1 <= T <= 10) là số lượng test.

T nhóm dòng sau, mỗi nhóm dòng gồm một test.

Mỗi nhóm dòng có định dạng:

* Dòng đầu tiên chứa 2 số nguyên N, M (2 <= N <= 200000, 1 <= M <= 20)
* Dòng thứ hai chứa N số nguyên H\_1, H\_2, …, H\_N (0 <= H\_i <= 1000000).
* M dòng tiếp theo, dòng thứ i (1 <= i <= M) chứa 4 số nguyên A\_i, B\_i, U\_i, D\_i (1 <= A\_i, B\_i <= N, 0 <= U\_i, D\_i <= 1000000).

**Output:** ghi ra file **parkour.out**

Với mỗi test, theo đúng thứ tự được cho trong input, in ra trên một dòng một số thực là giá trị nhỏ nhất của *max*{ |P\_i - H\_i| }. **Bạn phải in đúng 6 chữ số thập phân sau dấu phẩy**.

**Subtask:**

*Subtask 1 (30%):* M = 1

*Subtask 1 (30%):* A\_i < B\_i với mọi 1 <= i <= M

*Subtask 2 (40%):* Không có ràng buộc gì thêm

**Ví dụ**

|  |  |
| --- | --- |
| **parkour.in** | **parkour.out** |
| 4 2 1 0 10 1 2 3 8 5 1 50 59 55 51 47 1 5 2 1 5 2 46 38 38 22 8 4 1 3 2 3 5 4 1 8 5 53 25 24 81 77 40 29 21 1 8 14 9 3 1 5 7 6 8 2 1 5 4 1 8 8 1 11 10 | 3.500000 4.500000 15.000000 24.000000 483009.500000 |

---------------------------**HẾT**---------------------------