OLYMPIC TIN HỌC SINH VIÊN LẦN THỨ XXIX, 2020

Khối thi: SIÊU CÚP

Thời gian làm bài: 240 phút

Ngày thi: 09-12-2020

Nơi thi: ĐẠI HỌC CẦN THƠ

TỔNG QUAN ĐỀ THI

Bài 1. Con đường trọng yếu — CROAD

Một đất nước có n thành phố được đánh số từ 1 tới n. Có m con đường cao tốc hai chiều được đánh số từ 1 tới m, mỗi con đường nối trực tiếp hai thành phố phân biệt. Các con đường này đảm bảo từ một thành phố bất kỳ có thể đi đến một thành phố bất kỳ khác hoặc trực tiếp hoặc gián tiếp thông qua các các con đường khác. Giữa hai thành phố có thể có nhiều hơn một con đường nối trực tiếp.

Theo kế hoạch của Ban quản lý đường bộ, sắp tới mỗi đợt sẽ kiểm tra định kì một trong m con đường cao tốc. Trong thời gian kiểm tra người dân sẽ bị cấm ra, vào con đường này. Một con đường được xem là *trọng yếu* đối với hai thành phố u và v khi và chỉ khi mọi cách đi từ u đến v đều phải đi qua con đường đó (không được phép đi qua con đường đang kiểm tra). Lưu ý là nếu không có đường đi nào từ u đến v thì đối với hai thành phố này không có con đường nào là trọng yếu. Công ty BMAP quyết định xây dựng một phần mềm tra cứu giúp người dân nơi đây tìm xem, với một cặp thành phố (u, v), nếu một con đường được chọn để kiểm tra thì sẽ có những con đường nào là trọng yếu trong số $m - 1$ con đường còn lại.

Yêu cầu: Bạn hãy giúp công ty BMAP đưa ra kết quả tra cứu cho người dân.

Dữ liệu

Vào từ dòng nhập chuẩn (`stdin`):

- Dòng đầu tiên chứa 2 số nguyên dương n và m $(n, m \geq 2)$;
- Dòng thứ i trong số m dòng tiếp theo $(1 \leq i \leq m)$ chứa 2 số nguyên dương u và v $(u, v \leq n; u \neq v)$ cho biết đường cao tốc thứ i nối trực tiếp hai thành phố u và v;
- Dòng tiếp theo chứa một số nguyên dương q là số lượng truy vấn;
- Dòng thứ j trong số q dòng tiếp theo $(1 \leq j \leq q)$ chứa 3 số nguyên dương k, u và v $(k \leq m; u, v \leq n; u \neq v)$ mô tả truy vấn thứ j nếu con đường k đang trong đợt kiểm tra thì cần tìm số con đường trọng yếu giữa hai thành phố u và v.

Các số trên cùng một dòng cách nhau bởi dấu cách.

Kết quả

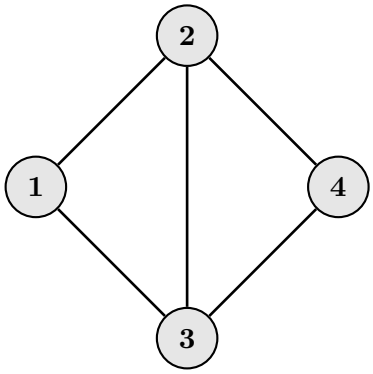
In ra dòng xuất chuẩn (`stdout`) q dòng, dòng thứ j $(1 \leq j \leq q)$ ghi một số duy nhất là tổng số con đường trọng yếu tìm được đối với truy vấn thứ j trong dữ liệu vào.

Hạn chế

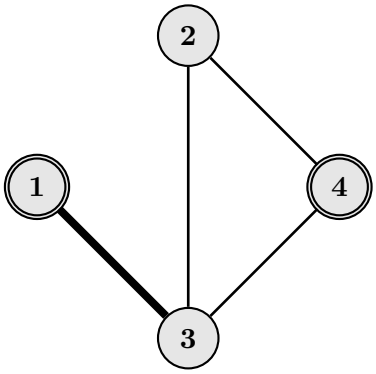
- Subtask 1 (15 điểm): $n, m, q \leq 500$;
- Subtask 2 (15 điểm): $n, m, q \leq 5000$;
- Subtask 3 (20 điểm): $n, m, q \leq 200000$, $k = 1$ trong mọi truy vấn;
- Subtask 4 (20 điểm): $n, m, q \leq 200000$, $u = 1$ và $v = 2$ trong mọi truy vấn;
- Subtask 5 (30 điểm): $n, m, q \leq 200000$.

Ví dụ

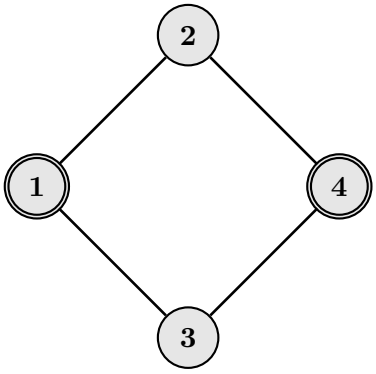
Dữ liệu	Kết quả
4 5 1 2 1 3 2 3 2 4 3 4 3 1 1 4 3 1 4 4 1 4	1 0 1



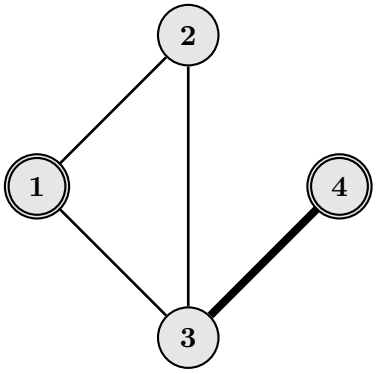
Đồ thị ban đầu



Truy vấn 1



Truy vấn 2



Truy vấn 3

Hình minh họa ví dụ

Bài 2. Phép XOR — MXOR

Long có một dãy số nguyên không âm $a = a_1, a_2, \ldots, a_n$ ($a_i \leq 10^9, \forall i, 1 \leq i \leq n$) và muốn tìm ra một cặp (i, j), $1 \leq i < j \leq n$, sao cho $a_i \wedge a_j$ lớn nhất có thể. Ở đây $\wedge$ là phép toán xor, hay còn gọi là phép hoặc triệt tiêu. Dữ liệu đảm bảo cặp (i, j) tối ưu là duy nhất.

Vốn tính lập dị, tuy Long muốn được giúp nhưng lại không muốn tiết lộ dãy a. Thay vào đó, bạn có thể trao đổi với Long bằng các hàm sau:

- `int get_n()`: trả về số phần tử của dãy a;
- `int max_xor(vector<int> &`I`, vector<int> &`J`)` (hoặc `int max_xor(int[]` I`, int[]` J`)` đối với Java): trả về $\max_{i \in I, j \in J} a_i \wedge a_j$ với I và J là hai tập con không giao nhau của tập $\{1, 2, \ldots, n\}$;
- `void answer(int` i`, int` j`)`: hàm này nhận vào hai tham số i, j là câu trả lời cho Long.

Long sẽ trả lời không quá 33 câu hỏi dạng `max_xor(`I`, `J`)`. Nếu bạn hỏi quá nhiều, hỏi với I và J không phải tập con của $\{1, 2, \ldots, n\}$ hoặc hỏi với I và J có phần tử chung thì bài của bạn sẽ bị chấm sai.

Để sử dụng các hàm trên, với C++ bạn cần khai báo `#include"MXORLIB.h"` ở đầu chương trình, sau đó các hàm đó có thể được gọi ở bất kỳ đâu trong chương trình của bạn. Xem file `MXORLIB.h` và `MXOR.cpp` trong mục đính kèm để hiểu rõ hơn cách tương tác. Lưu ý đây là thư viện ví dụ, có thể khác với thư viện dùng để chấm bài. Trình chấm đảm bảo rằng nếu chương trình của bạn biên dịch được với thư viện ví dụ thì cũng biên dịch được trên hệ thống (để MXORLIB.h và chương trình của bạn vào cùng một thư mục rồi biên dịch như bình thường).

Đối với Java, hệ thống cung cấp sẵn một `class` tên là `MXORLIB` và bạn không cần phải khai báo gì thêm. Các hàm trên được để `static`, thí sinh gọi `MXORLIB.get_n()`, `MXORLIB.max_xor(`I`, `J`)`, `MXORLIB.answer(`i`, `j`)` để tương tác.

Ví dụ

Dãy a ban đầu là {4 3 1 2}

Gọi hàm	Giá trị trả về

`get_n()`
`max_xor({1}, {2, 3})`
`answer(1, 2)`
	`4`
	`7`
	`Kết thúc chương trình. Bạn đã trả lời đúng và chương trình đạt điểm của ví dụ này.`

Hạn chế

- Subtask 1 (16 điểm): $n \leq 17$
- Subtask 2 (36 điểm): $17 < n \leq 10^4$
- Subtask 3 (48 điểm): $10^4 < n \leq 10^5$

Bài 3. Vị trí hạnh phúc — HAPPOS

Dây chuyền sản xuất nhà máy VAIP có n vị trí sản xuất được đánh số $\{1, 2, \ldots, n\}$ và bố trí dưới dạng một đồ thị có cấu trúc như sau:

- mỗi một vị trí sản xuất là một đỉnh của đồ thị;
- hai vị trí kề nhau tương ứng với cạnh của đồ thị;
- đồ thị là vô hướng liên thông và không có chu trình.

Có n công nhân được đánh số $\{1, 2, \ldots, n\}$ được được đào tạo để phụ trách các vị trí trong dây chuyền. Do sở trường của mỗi người nên người i sẽ cảm thấy hạnh phúc nếu được xếp phụ trách ở vị trí i ($\forall i, 1 \leq i \leq n$). Ban đầu các công nhân được bố trí đúng vị trí hạnh phúc của mình, nghĩa là người i được bố trí ở vị trí i. Để có thể thích ứng với nhiều vị trí khác nhau mà không làm ảnh hưởng sản xuất, sau mỗi ngày, người quản lý chọn ra đúng hai công nhân kề nhau, nghĩa là ở vị trí hai đỉnh kề nhau trên đồ thị, và đảo vị trí của họ. Sau một số ngày đảo chỗ, đến hôm nay người quản lý nhận ra là quên mất không ghi lại nhật ký vị trí các lần đảo để có thể làm ngược lại quá trình những ngày vừa qua nhằm đưa tất cả công nhân về vị trí hạnh phúc của họ.

Yêu cầu: Cho biết vị trí các công nhân hiện đang phụ trách ngày hôm nay, bạn hãy giúp người quản lý tìm ra thứ tự đảo vị trí hai công nhân mỗi ngày để đưa tất cả công nhân về vị trí hạnh phúc của họ sao cho số ngày phải sử dụng là ít nhất có thể.

Đây là bài toán chỉ cần nộp các file kết quả đầu ra. Thí sinh được cho 10 file đầu vào tương ứng với 10 test, đối với mỗi file đầu vào thí sinh cần nộp một file kết quả đầu ra mô tả kế hoạch hoán đổi vị trí theo từng ngày. Với mỗi file kết quả đầu ra mô tả đúng đắn quá trình hoán đổi, điểm của thí sinh phụ thuộc vào số ngày sử dụng để đưa toàn bộ công nhân về vị trí hạnh phúc (xem cách tính điểm trong phần **Chấm điểm**).

Dữ liệu

Thí sinh được cung cấp 10 file dữ liệu đầu vào với tên tương ứng là: input_0.txt, input_1.txt,..., input_9.txt. Mỗi file dữ liệu đầu vào có khuôn dạng như sau:

- Dòng đầu chứa một số nguyên dương n là số đỉnh của đồ thị;
- Mỗi dòng trong số $n - 1$ dòng tiếp theo chứa hai số nguyên dương u và v là hai đỉnh đầu mút một cạnh của đồ thị;
- Dòng tiếp theo chứa n số nguyên dương, số thứ i là số hiệu của công nhân hiện đang đứng tại vị trí i.

Các số trên một dòng cách nhau bởi dấu cách.

Kết quả

Đối với mỗi file dữ liệu đầu vào, thí sinh cần nộp một file kết quả đầu ra mô tả kế hoạch hoán đổi vị trí theo từng ngày, các file kết quả đầu ra có tên tương ứng là: output_0.txt, output_1.txt,..., output_9.txt. Mỗi file kết quả đầu ra có khuôn dạng:

- Dòng đầu tiên chứa một số nguyên t là số ngày bạn cần để đưa tất cả công nhân về vị trí hạnh phúc;
- Dòng thứ i trong số t dòng tiếp theo ghi hai số nguyên dương u và v là hai vị trí sản xuất mà công nhân ở đó phải đảo chỗ ngày thứ i.

Thí sinh có thể nộp từng file kết quả đầu ra tương ứng với file dữ liệu đầu vào hoặc nén các file kết quả đầu ra thành một file có tên submission.zip để nộp. Điểm số của mỗi test là điểm cao nhất đạt được trong các lần nộp file kết quả đầu ra của test đó. Điểm số của bài là tổng điểm của từng test.

Lưu ý: Kích thước tối đa cho phép của mỗi file nộp lên server là 10Mb.

Hạn chế

- Subtask 1 (30 điểm): $n \leq 10$;
- Subtask 2 (70 điểm): $10 < n \leq 1000$.

Chấm điểm:

Với mỗi file dữ liệu đầu vào, gọi GK là số ngày cần để đưa toàn bộ công nhân về vị trí hạnh phúc của họ theo phương án của Ban giám khảo (giá trị này thí sinh không được biết, chỉ dùng khi chấm), TS là số ngày cần để đưa toàn bộ công nhân về vị trí hạnh phúc của họ theo phương án của thí sinh trong file đầu ra tương ứng với file đầu vào. Đặt $P = (TS - GK)/GK$, khi đó, thí sinh sẽ nhận được:

- 0 điểm nếu $P > 1$;
- 10 điểm nếu $P \leq 0$;
- $-\log_{10}(P \times 0.9999 + 0.0001) \times 2.5$ điểm nếu $0 < P \leq 1$;

trên tổng số 10 điểm của test đó.

Lưu ý: Thí sinh sẽ không được điểm nếu file output nộp lên không hợp lệ.

Ví dụ

Dữ liệu	Kết quả
3 1 2 2 3 2 1 3	1 1 2

Bài 4. Lệnh tiến công — ATTACK

"*Ngày* `DD` *tới, vào giờ* `HH:MM`*, toàn quân tổng tiến công*!"

Đó là quân lệnh từ sở chỉ huy cần được truyền tới tất cả các đơn vị chiến đấu. Tuấn làm việc trong bộ phận phụ trách thiết kế các module đảm bảo thông tin được truyền đi an toàn và chính xác. Thông tin được truyền đi sẽ gồm ba giai đoạn:

1. Mã hóa thông tin (encode): từ sở chỉ huy, thông tin (`DD, HH, MM`) sẽ được mã hóa thành một dãy bit;
2. Truyền tin (transmission): sử dụng những phương tiện thô sơ để truyền dãy bit tới các máy nhận ở các đơn vị;
3. Giải mã thông tin (decode): từ dãy bit nhận được, giải mã lại thông tin (`DD, HH, MM`).

Tuấn phụ trách thiết kế module 1 và 3 nhưng gặp rất nhiều khó khăn. Lý do chính là do kĩ thuật truyền tin thô sơ và phụ thuộc vào yếu tố kĩ năng của con người, nên không phải thông tin lúc nào cũng được truyền đi một cách hoàn toàn chính xác. Thông tin truyền đi có thể bị sai sót tối đa 1 bit. Vì vậy, yêu cầu của hệ thống mã hóa này phải có khả năng phát hiện lỗi, và tự sửa chữa sai sót. Nói cách khác, cho dù thông tin truyền đi có thể sai 1 bit nhưng vẫn phải tự khôi phục được.

Yêu cầu: Hãy giúp Tuấn thực hiện được công việc của mình.

Giao tiếp:

Bạn cần viết hai hàm sau:

1. `string encode (string message)`

   Hàm nhận vào một xâu độ dài 8 có dạng `"DD HH:MM"` trong đó `DD` là một số nguyên trong đoạn [1,31], `HH` là một số nguyên trong đoạn [0,23] và `MM` là một số nguyên trong đoạn [0,59]. Hàm cần trả về một xâu không quá 50 kí tự chỉ gồm các kí tự 0 hoặc 1 là xâu được mã hóa.

2. `string decode (string encryptedMessage)`

   Hàm nhận vào một xâu có độ dài không quá 50 kí tự, chỉ gồm các kí tự 0 hoặc 1. Hàm cần trả ra một xâu có độ dài 8 có định dạng giống xâu đầu vào của hàm encode là xâu đã được giải mã.

Làm bài:

Trên hệ thống, các bạn có thể download file `attack_public.zip` chứa các file hỗ trợ làm bài. Trong đó:

- `attack.cpp / attack.java` là file các bạn cần làm việc và viết thuật toán mã hóa.
- `stub.cpp / stub.java` là file hỗ trợ các chức năng nhập xuất dữ liệu.
- `compile_cpp.sh / compile_java.sh` là script hỗ trợ việc biên dịch chương trình.

Bạn có thể làm các việc sau:

- Viết thuật toán mã hóa của bạn vào file `attack.cpp` hoặc `attack.java`. Các bạn có thể viết thêm các hàm phụ trợ khác.
- Biên dịch và thực thi bằng script được cung cấp như sau:
  - Vào thư mục chứa file `attack.cpp` hoặc `attack.java`, click chuột phải và chọn Open in Terminal.

'

  - Biên dịch file mã nguồn ra file thực thi bằng cách chạy lệnh `./compile_cpp.sh` hoặc `./compile_java.sh`.
  - Thực thi bằng cách chạy lệnh `./attack` hoặc `./run_java.sh`.

- File thực thi `attack` nhận dữ liệu vào từ dòng nhập chuẩn (stdin) ở một trong hai dạng:

  1. `ENCODE DD HH MM`
     - Ví dụ: ENCODE 1 12 5 với ý nghĩa gọi hàm encode("01 12:05")
  2. `DECODE str`
     - Ví dụ DECODE 00011101 với ý nghĩa gọi hàm decode("00011101")
  3. File thực thi sẽ in ra một xâu là kết quả của hàm tương ứng mà chương trình bạn chạy.

Ngoài ra, trong quá trình làm bài, các bạn có thể tự thay đổi các file `stub.cpp / stub.java` để phục vụ theo cách nhập xuất dự liệu mà bạn muốn.

Nộp bài:

Các bạn chỉ nộp file `attack.cpp` hoặc `attack.java` mà không nộp file nào khác.

Chấm bài:

Bài của các bạn sẽ được chấm như sau. Ban giám khảo chuẩn bị một file hệ thống chấm (được gọi là manager).

- Chương trình manager sẽ gọi hàm encode của bạn với một bộ dữ liệu (`DD, HH, MM`) nào đó và thu được xâu `encryptedMessage`.
- Chương trình manager sẽ lặp lại các bước sau nhiều lần:
  - Sửa một bit hoặc không sửa bit nào tạo thành xâu `receivedMessage`.
  - Chương trình manager sẽ gọi hàm decode của bạn với đầu vào là xâu `receivedMessage` và so sánh kết quả thu được với bộ dữ liệu ban đầu.

Lưu ý rằng, những lần hàm encode và decode của bạn được gọi sẽ được chạy ở các tiến trình độc lập. Vì vậy, nếu hàm encode có sử dụng và lưu trữ dữ liệu vào các biến toàn cục, các dữ liệu này sẽ không tồn tại khi hàm decode được thực thi.

Chấm điểm:

Với mỗi bộ dữ liệu, bạn sẽ không được điểm nếu:

- Tương tác sai quy cách (dữ liệu trả ra không đúng định dạng)
- Chạy sinh lỗi
- Chạy quá thời gian
- Xâu sau khi giải mã khác với xâu ban đầu

Gọi độ dài xâu mã hóa của bạn là L, bạn sẽ nhận được điểm dựa trên độ tốt của L như sau:

Độ dài xâu mã hóa của bạn	Điểm của test
$1 \leq L \leq 21$	100%
$L = 22$	90%
$L = 23$	80%
$24 \leq L \leq 25$	60%
$26 \leq L \leq 30$	40%
$31 \leq L \leq 35$	30%
$36 \leq L \leq 40$	20%
$41 \leq L \leq 50$	10%