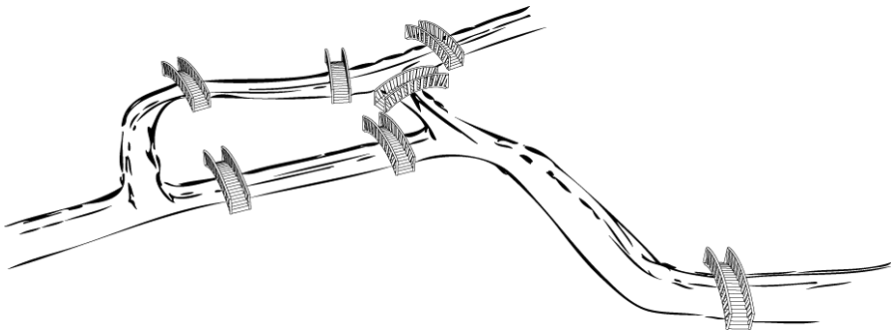
TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC TP.HCM (HUFLIT)
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP THỰC HÀNH

LÝ THUYẾT ĐỒ THỊ

(GRAPH THEORY)



Biên soạn: Tôn Quang Toại

THÁNG 1, 2024

NỘI DUNG

Buổi 1. Nhập, xuất và thao tác cơ bản trên đồ thị 1

Buổi 2. Vận dụng các thao tác cơ bản trên đồ thị..... 5

Buổi 3. Tìm kiếm trên đồ thị bằng thuật toán Breadth First Search – BFS..... 11

Buổi 4. Vận dụng thuật toán BFS 14

Buổi 5. Tìm kiếm trên đồ thị bằng thuật toán Depth First Search – DFS 17

Buổi 6. Vận dụng thuật toán DFS..... 19

Buổi 7. Tìm đường đi ngắn nhất 23

Buổi 8. Vận dụng thuật toán tìm đường đi ngắn nhất 27

Buổi 9. Cây khung và Cây khung nhỏ nhất 31

Buổi 10. Tìm đường đi, chu trình Euler..... 35

Buổi 1. Nhập, xuất và thao tác cơ bản trên đồ thị

Bài 1. Bậc của đồ thị vô hướng

Cho đồ thị vô hướng $G = (V, E)$ có n $(n \leq 1000)$ đỉnh, các đỉnh được đánh số từ 1 đến n. Đồ thị G được lưu trong một file văn bản dưới dạng một ma trận kề. Hãy tổ chức cấu trúc dữ liệu ma trận kề để biểu diễn đồ thị G, và viết chương trình đọc đồ thị G từ file đã cho vào cấu trúc dữ liệu đó, sau đó tính bậc của các đỉnh trong đồ thị G (bậc của một đỉnh là số cạnh liên thuộc với đỉnh đó).

Dữ liệu vào: File văn bản AdjecencyMatrix.INP

- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

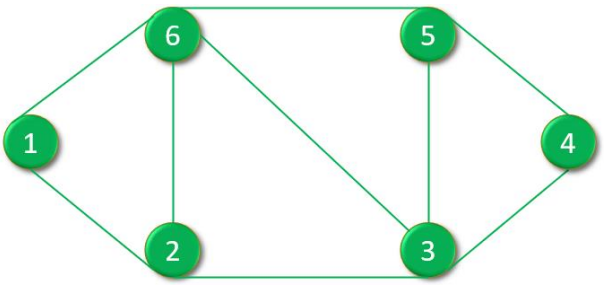
Dữ liệu ra: File văn bản AdjecencyMatrix.OUT

- Dòng thứ nhất chưa số n là số đỉnh của đồ thị.
- Dòng thứ hai chứa n số nguyên tương ứng là bậc của các đỉnh $1, 2, \ldots, n$

(Các số trên cùng một dòng, cách nhau bằng 1 khoảng trắng)

Ví dụ

AdjecencyMatrix .INP	AdjecencyMatrix.OUT
6 0 1 0 0 0 1 1 0 1 0 0 1 0 1 0 1 1 1 0 0 1 0 1 0 0 0 1 1 0 1 1 1 1 0 1 0	6 2 3 4 2 3 4



Bài 2. Bậc vào, bậc ra

Cho đồ thị có hướng $G = (V, E)$ có n đỉnh được đánh số từ 1 đến n. Bậc vào của đỉnh x $(x \in V)$ là số cung đi vào đỉnh x. Bậc ra của đỉnh x là số cung đi ra đỉnh x. Hãy tính bậc vào và bậc ra của tất cả các đỉnh trong đồ thị G.

Dữ liệu vào: File văn bản BacVaoRa.INP

- Dòng đầu tiên chứa số nguyên n $(n \leq 1000)$ là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số, biểu diễn ma trận kề của đồ thị.

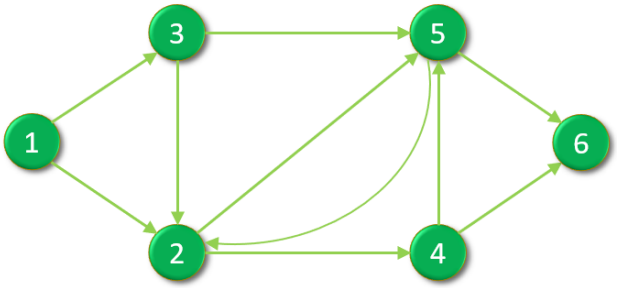
Dữ liệu ra: File văn bản BacVaoRa.OUT

- Dòng đầu là số nguyên dương n là số đỉnh của đồ thị.
- n dòng tiếp theo, dòng thứ i gồm hai số nguyên là bậc vào và bậc ra của đỉnh i

(Các số trên cùng một dòng, cách nhau bằng 1 khoảng trắng)

Ví dụ

BacVaoRa.INP	BacVaoRa.OUT
6 0 1 1 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0 0 0 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0	6 0 2 3 2 1 2 1 2 3 2 2 0



Bài 3. Danh sách kề

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 10^5$) đỉnh, các đỉnh được đánh số từ 1 đến n và m cạnh ($m \leq 10^5$). Hãy tổ chức cấu trúc dữ liệu cho đồ thị dưới dạng danh sách kề, và viết chương trình đọc đồ thị G từ file đã cho, sau đó tính bậc của các đỉnh trong đồ thị G.

Dữ liệu vào: File văn bản AdjecencyList.INP

- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

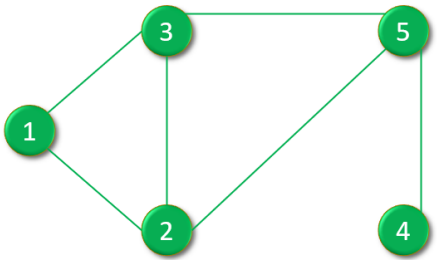
Chú ý: Đỉnh cô lập (đỉnh không nối với các đỉnh khác) thì dòng đó rỗng

Dữ liệu ra: File văn bản AdjecencyList.OUT

- Dòng thứ nhất chưa số n là số đỉnh của đồ thị.
- Dòng thứ hai chứa n số nguyên tương ứng là bậc của các đỉnh $1, 2, \ldots, n$

Ví dụ:

AdjecencyList.INP	AdjecencyList.OUT
5 2 3 1 3 5 1 2 5 5 2 3 4	5 2 3 3 1 3



Bài 4. Danh sách cạnh

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 10^5$) đỉnh, các đỉnh được đánh số từ 1 đến n và m cạnh ($m \leq 10^5$) được biểu diễn dưới dạng danh sách cạnh trong một file văn bản. Hãy tổ chức cấu trúc dữ liệu cho đồ thị dưới dạng danh sách cạnh, và viết chương trình đọc đồ thị G từ file đã cho, sau đó tính bậc của các đỉnh trong đồ thị G.

Dữ liệu vào: File văn bản EdgeList.INP

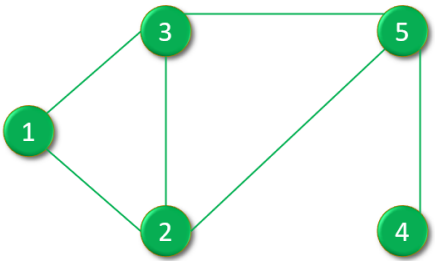
- Dòng đầu tiên chứa hai số nguyên n, m là số đỉnh và số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng là cặp số biểu diễn một cạnh của đồ thị (các số cách nhau 1 khoảng trắng)

Dữ liệu ra: File văn bản EdgeList.OUT

- Dòng đầu là số nguyên dương n là số đỉnh của đồ thị.
- Dòng thứ hai chứa n số nguyên tương ứng là bậc của các đỉnh $1, 2, \ldots, n$

Ví dụ:

EdgeList.INP	EdgeList.OUT
5 6 1 2 1 3 2 3 2 5 3 5 4 5	5 2 3 3 1 3



Buổi 2. Vận dụng các thao tác cơ bản trên đồ thị

Bài 1. Chuyển danh sách cạnh sang danh sách kề

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 10^5$) đỉnh, các đỉnh được đánh số từ 1 đến n và m cạnh ($m \leq 10^5$) được lưu trong file văn bản dưới dạng danh sách cạnh. Hãy viết chương trình chuyển đổi đồ thị này sang danh sách kề.

Dữ liệu vào: File văn bản Canh2Ke.INP

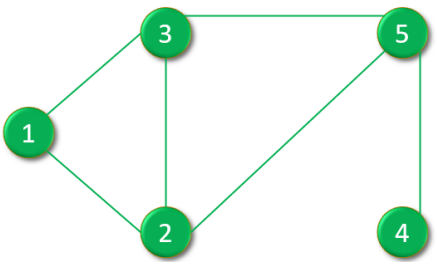
- Dòng đầu tiên chứa hai số nguyên: n, m tương ứng là số đỉnh và số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa hai đỉnh mô tả cạnh nối 2 đỉnh đó

Dữ liệu ra: File văn bản Canh2Ke.OUT

- Dòng đầu tiên chứa số đỉnh n
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị (các đỉnh trong danh sách được sắp xếp từ nhỏ đến lớn).

Ví dụ

Canh2Ke.INP	Canh2Ke.OUT
5 6 1 2 1 3 2 3 2 5 3 5 4 5	5 2 3 1 3 5 1 2 5 5 2 3 4



Bài 2. Chuyển danh sách kề sang danh sách cạnh

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 10^5$) đỉnh, các đỉnh được đánh số từ 1 đến n và m cạnh ($m \leq 10^5$) được lưu trong file văn bản dưới dạng danh sách kề. Hãy viết chương trình chuyển đổi đồ thị này sang danh sách cạnh.

Dữ liệu vào: File văn bản Ke2Canh.INP

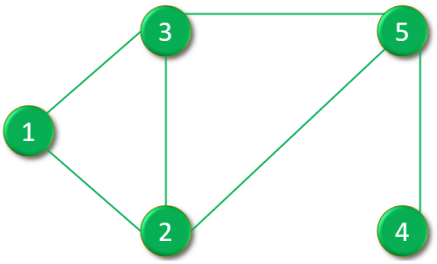
- Dòng đầu tiên chứa số đỉnh n
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị (các đỉnh trong danh sách được sắp xếp từ nhỏ đến lớn).

Dữ liệu ra: File văn bản Ke2Canh.OUT

- Dòng đầu tiên chứa hai số nguyên: n, m tương ứng là số đỉnh và số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa hai đỉnh mô tả cạnh nối 2 đỉnh đó

Ví dụ

Ke2Canh.INP	Ke2Canh.OUT
5 2 3 1 3 5 1 2 5 5 2 3 4	5 6 1 2 1 3 2 3 2 5 3 5 4 5



Bài 3. Bồn chứa

Cho đồ thị có hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n. "Bồn chứa" trong đồ thị G là đỉnh chỉ có cung vào mà không có cung ra. Viết chương trình tìm các đỉnh bồn chứa trong đồ thị G.

Dữ liệu vào: File văn bản BonChua.INP

- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo là ma trận kề của đồ thị.

Dữ liệu ra: File văn bản BonChua.OUT

- Dòng đầu là số nguyên dương k là số lượng bồn chứa trong đồ thị (Ghi 0 nếu G không có bồn chứa).
- Nếu $k > 0$ thì dòng thứ hai chứa danh sách các đỉnh bồn chứa (các đỉnh được sắp theo thứ tự từ nhỏ đến lớn).

Bài 4. Đồ thị chuyển vị

Cho đồ thị có hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n. Đồ thị chuyển vị của đồ thị G là đồ thị $G^T = (V, E^T)$. Trong đó

$$E^T = \{(u, v) : (v, u) \in E\}$$

Hãy xây dựng G^T từ G.

Dữ liệu vào: File văn bản ChuyenVi.INP

- Dòng đầu tiên chứa số đỉnh n
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng

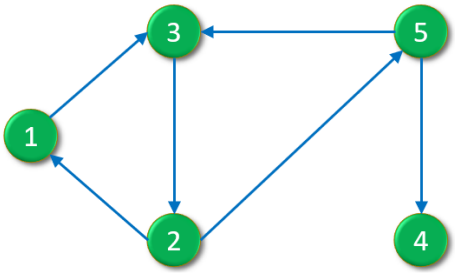
với một cung (i, j) của đồ thị G (các đỉnh trong danh sách được sắp xếp từ nhỏ đến lớn).

Dữ liệu ra: File văn bản ChuyenVi.OUT

- Dòng đầu tiên chứa số đỉnh n
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cung (i, j) của đồ thị G^T (các đỉnh trong danh sách được sắp xếp từ nhỏ đến lớn).

Ví dụ

DSKe2Canh.INP	DSKe2Canh.OUT
5 3 1 5 2 3 4	5 2 3 1 5 5 2



Bài 5. Độ dài trung bình của cạnh

Cho đồ thị vô hướng có trọng số $G = (V, E)$ có n ($n \leq 10^5$) đỉnh, các đỉnh được đánh số từ 1 đến n, và m ($m \leq 10^5$). Tìm các cạnh có độ dài dài nhất và tính độ dài trung bình của các cạnh.

Dữ liệu vào: File văn bản TrungBinhCanh.INP

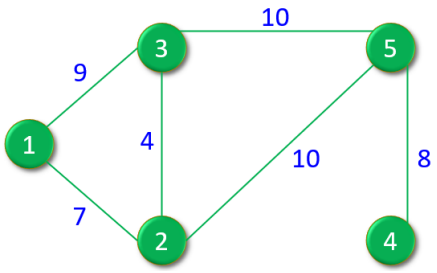
- Dòng đầu tiên chứa hai số nguyên: n, m tương ứng là số đỉnh và số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa ba số nguyên: u, v, w mô tả cạnh (u, v) có trọng số w.

Dữ liệu ra: File văn bản TrungBinhCanh.OUT

- Dòng thứ nhất chứa độ dài trung bình các cạnh (lấy 2 số lẻ thập phân)
- Dòng thứ 2 chứa số k là số lượng cạnh có độ dài dài nhất.
- k dòng tiếp theo k bộ số (u, v, w) k cạnh dài nhất

Ví dụ

TrungBinhCanh.INP	TrungBinhCanh.OUT
5 6 1 2 7 1 3 9 2 3 4 2 5 10 3 5 10 4 5 8	7.00 2 2 5 10 3 5 10



Bài tập làm thêm

Bài 1. Chuyển ma trận kề sang danh sách kề

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n được lưu trong file văn bản dưới dạng ma trận kề. Hãy viết chương trình chuyển đổi đồ thị này sang danh sách kề.

Dữ liệu vào: File văn bản MaTranKe2DSKe.INP

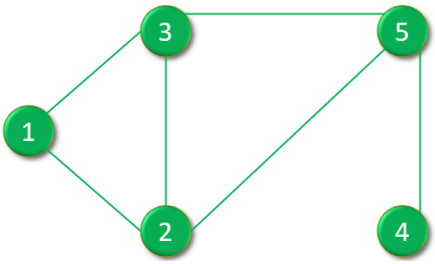
- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

Dữ liệu ra: File văn bản MaTranKe2DSKe.OUT

- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị (các đỉnh trong danh sách được sắp xếp từ nhỏ đến lớn).

Ví dụ

MaTranKe2DSKe.INP	MaTranKe2DSKe.OUT
5 0 1 1 0 0 1 0 1 0 1 1 1 0 0 1 0 0 0 0 1 0 1 1 1 0	5 2 3 1 3 5 1 2 5 5 2 3 4



Bài 2. Chuyển ma trận kề sang danh sách cạnh

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n được lưu trong file văn bản dưới dạng ma trận kề. Hãy viết chương trình chuyển đổi đồ thị này sang danh sách cạnh.

Dữ liệu vào: File văn bản MaTranKe2DSCanh.INP

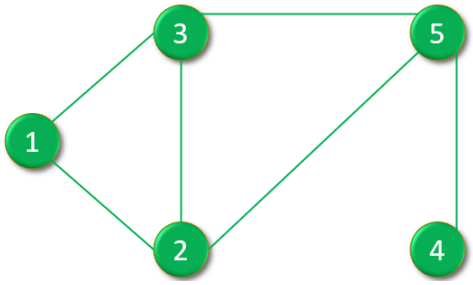
- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

Dữ liệu ra: File văn bản MaTranKe2DSCanh.OUT

- Dòng đầu tiên chứa số đỉnh m là số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa hai đỉnh mô tả cạnh nối 2 đỉnh đó

Ví dụ:

MaTranKe2DSCanh.INP	MaTranKe2DSCanh.OUT
5 0 1 1 0 0 1 0 1 0 1 1 1 0 0 1 0 0 0 0 1 0 1 1 1 0	6 1 2 1 3 2 3 2 5 3 5 4 5



Viết các hàm tiện ích cho phép chuyển đổi qua lại giữa các cách biểu diễn đồ thị khác nhau

- Chuyển từ Danh sách kề sang ma trận kề.
- Chuyển từ Danh sách cạnh sang ma trận kề.

Buổi 3. Tìm kiếm trên đồ thị bằng thuật toán Breadth First Search – BFS

Bài 1. Liệt kê các đỉnh liên thông với đỉnh s

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n và một đỉnh s ($s \in V$). Hãy cho biết từ đỉnh s có thể đi đến được những đỉnh nào (sử dụng thuật toán Breadth First Search – BFS). Khi một đỉnh có nhiều đỉnh kề, thì các đỉnh được xét theo thứ tự từ nhỏ đến lớn.

Dữ liệu vào: File văn bản BFS.INP

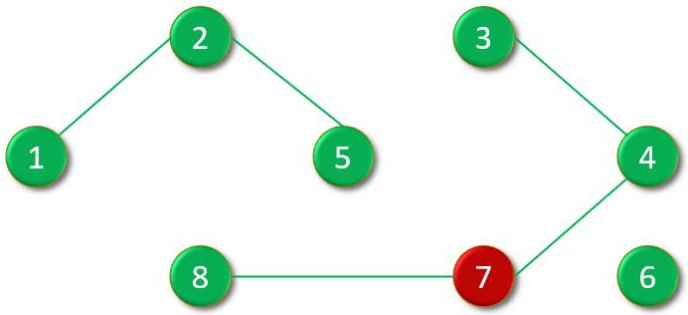
- Dòng đầu tiên chứa hai số nguyên: n, s tương ứng là số đỉnh của đồ thị và đỉnh s.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản BFS.OUT

- Dòng đầu tiên ghi số k là số lượng đỉnh có thể đi đến được từ đỉnh s.
- Dòng thứ hai ghi k đỉnh tìm được.

Ví dụ

BFS.INP	BFS.OUT
8 7 2 1 5 4 3 7 2 4 8 7	3 4 8 3



Bài 2. Tìm đường đi

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n và hai đỉnh x, y ($x, y \in V$ và $x \neq y$). Hãy tìm đường đi từ đỉnh x đến đỉnh y bằng thuật toán BFS.

Dữ liệu vào: File văn bản TimDuong.INP

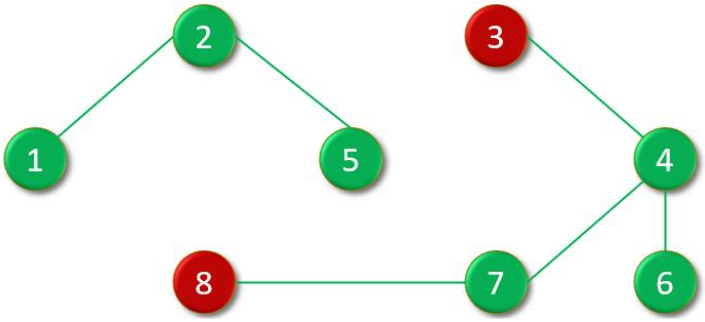
- Dòng đầu tiên chứa số 3 số nguyên: n, x, y.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản TimDuong.OUT

- Dòng đầu tiên ghi số nguyên dương k là số đỉnh nằm trên đường đi từ đỉnh x đến đỉnh y (Tính luôn cả đỉnh x và y).
- Dòng thứ hai chứa k số nguyên là các đỉnh trên đường đi từ x đến y.

Ví dụ

TimDuong.INP	TimDuong.OUT
8 3 8 2 1 5 4 3 6 7 2 4 4 8 7	4 3 4 7 8



Bài 3. Kiểm tra đồ thị liên thông

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n. Đồ thị được gọi là liên thông nếu từ một đỉnh ta có thể đi đến các đỉnh khác. Hãy viết chương trình kiểm tra đồ thị G có liên thông không.

Dữ liệu vào: File văn bản LienThong.INP

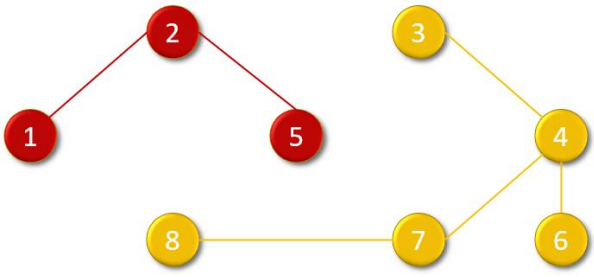
- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản LienThong.OUT

- Dòng duy nhất ghi ra chữ "YES" nếu đồ thị liên thông, ngược lại ghi chữ "NO"

Ví dụ

LienThong.INP	LienThong.OUT
8 2 1 5 4 3 6 7 2 4	NO



4 8 7	

Bài 4. Đếm số miền liên thông

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n. Miền liên thông là tập đỉnh liên thông với nhau và nếu thêm một đỉnh khác thì không còn liên thông nữa. Hãy viết chương trình cho biết G có bao nhiêu miền liên thông.

Dữ liệu vào: File văn bản DemLienThong.INP

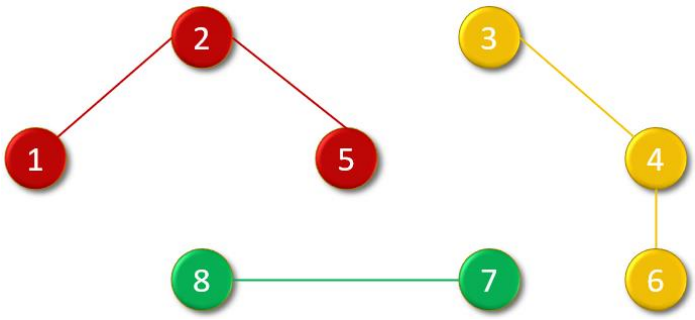
- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản DemLienThong.OUT

- Dòng duy nhất ghi số lượng miền liên thông tìm được

Ví dụ

DemLienThong.INP	DemLienThong.OUT
8 2 1 5 4 3 6 2 4 8 7	3



Buổi 4. Vận dụng thuật toán BFS

Bài 1. Liệt kê miền liên thông

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n. Hãy liệt kê các thành phần liên thông trong đồ thị bằng thuật toán BFS.

Dữ liệu vào: File văn bản MienLienThongBFS.INP

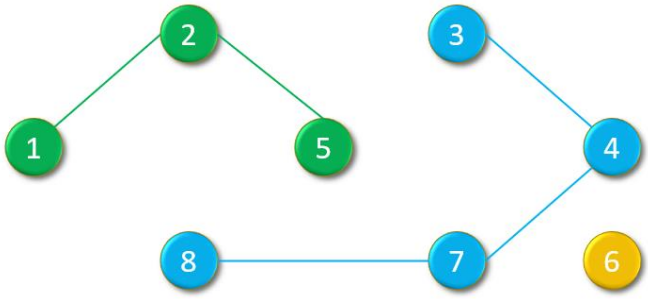
- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cung (i, j) của đồ thị.

Dữ liệu ra: File văn bản MienLienThongBFS.OUT

- Dòng đầu tiên ghi số nguyên dương k là số lượng thành phần liên thông của đồ thị.
- Dòng thứ i trong k dòng tiếp theo liệt kê các đỉnh của từng thành phần liên thông thứ k.

Ví dụ

MienLienThongBFS.INP	MienLienThongBFS.OUT
8 2 1 5 4 3 7 2 4 8 7	3 1 2 5 3 4 7 8 6



Chú ý: có một dòng trống giữa dòng có dữ liệu "2" và "4 8" để mô tả đỉnh 6 không kề với bất kỳ đỉnh nào

Bài 2. Cạnh cầu

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n và cạnh $(x, y) \in E$. Cạnh cầu của đồ thị là cạnh nếu xóa đi thì số miền liên thông của đồ thị tăng lên. Hãy kiểm tra xem cạnh (x, y) có phải là cạnh cầu của đồ thị G không.

Dữ liệu vào: File văn bản CanhCau.INP

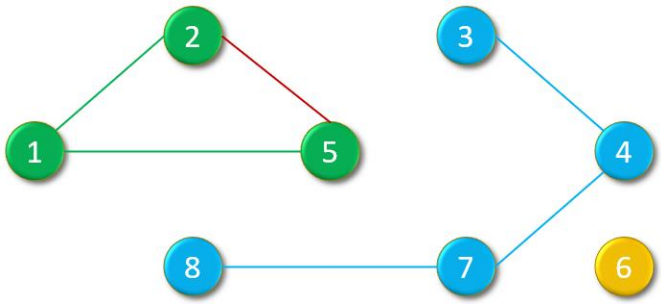
- Dòng đầu tiên chứa ba số đỉnh n, x, y
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cung (i, j) của đồ thị.

Dữ liệu ra: File văn bản CanhCau.OUT

- Dòng duy nhất chứa chữ "YES" nếu (x, y) là cạnh cầu, ngược lại ghi chữ "NO"

Ví dụ

CanhCau.INP	CanhCau.OUT
8 2 5 2 5 1 5 4 3 7 1 2 4 8 7	NO



Bài 3. Đỉnh khớp

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n và đỉnh x. Đỉnh x ($x \in V$) là đỉnh khớp nếu xóa các cạnh kề của đỉnh x thì số miền liên thông của đồ thị tăng lên ít nhất là 2. Hãy kiểm tra xem đỉnh x có phải là đỉnh khớp của đồ thị G không.

Dữ liệu vào: File văn bản DinhKhop.INP

- Dòng đầu tiên chứa hai số nguyên: n, x
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cung (i, j) của đồ thị.

Dữ liệu ra: File văn bản DinhKhop.OUT

- Dòng duy nhất chứa chữ "YES" nếu x là đỉnh khớp, ngược lại ghi chữ "NO"

Bài 4. Đi trên lưới

Cho lưới hình chữ nhật có kích thước $n \times m$ gồm các số $0, 1$. Các dòng được đánh số từ 1 đến n từ trên xuống dưới, các cột được đánh số từ 1 đến m từ trái sang phải. Ô trong bảng có giá trị 0 gọi là ô nước, ô có giá trị 1 gọi là đất. Từ một ô tại vị trí (i, j) là đất có thể đi đến 1 trong 4 ô xung quanh nếu các ô đó cùng là ô đất.

Cho hai ô $s(i_1, j_1)$ và $t(i_2, j_2)$ thuộc ô có giá trị 1. Hãy tìm một đường đi ngắn nhất từ $s(i_1, j_1)$ đến $t(i_2, j_2)$.

Dữ liệu vào: File văn bản Grid.INP

- Dòng thứ nhất chứa hai số nguyên n, m $(n, m \leq 1000)$
- Dòng thứ hai chứa 4 số i_1, j_1, i_2, j_2
- n dòng sau, mỗi dòng chứa m số 0, 1

Dữ liệu vào: File văn bản Grid.OUT

- Dòng đầu tiên ghi số nguyên k. Nếu $k = 0$ thì không có đường đi, ngược lại k là số ô đi từ s đến t (tính luôn 2 ô s, t)
- k dòng sau, mỗi dòng chứa hai số nguyên là trình tự tọa độ (chỉ số dòng, chỉ số cột) của các ô phải đi qua để đi từ ô s đến ô t

Ví dụ

Grid.INP	Grid.OUT
6 8 3 2 5 7 0 1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 1 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0	8 3 2 3 3 4 3 4 4 4 5 4 6 4 7 5 7

Buổi 5. Tìm kiếm trên đồ thị bằng thuật toán Depth First Search – DFS

Bài 1. Liệt kê các đỉnh liên thông với s

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n và một đỉnh s ($s \in V$). Hãy cho biết từ đỉnh s có thể đi đến được những đỉnh nào (sử dụng thuật toán Depth First Search – DFS). Khi một đỉnh có nhiều đỉnh kề, thì các đỉnh được xét theo thứ tự từ nhỏ đến lớn.

Dữ liệu vào: File văn bản DFS.INP

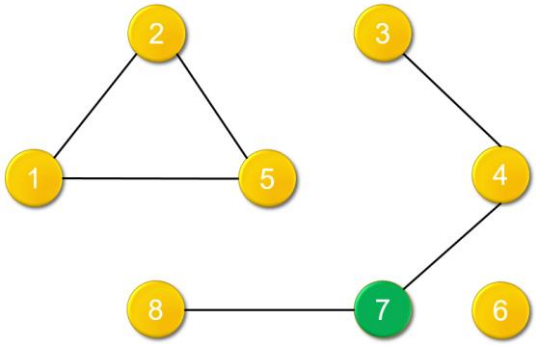
- Dòng đầu tiên chứa hai số nguyên: n, s tương ứng là số đỉnh của đồ thị và đỉnh s.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản DFS.OUT

- Dòng đầu tiên ghi số k là số lượng đỉnh tìm được.
- Dòng thứ hai ghi k đỉnh tìm được.

Ví dụ

DFS.INP	DFS.OUT
8 7 2 5 1 5 4 3 7 2 1 4 8 7	3 4 3 8



Bài 2. Tìm đường đi

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n và hai đỉnh x, y ($x, y \in V$ và $x \neq y$). Hãy tìm đường đi từ đỉnh x đến đỉnh y bằng thuật toán DFS.

Dữ liệu vào: File văn bản TimDuongDFS.INP

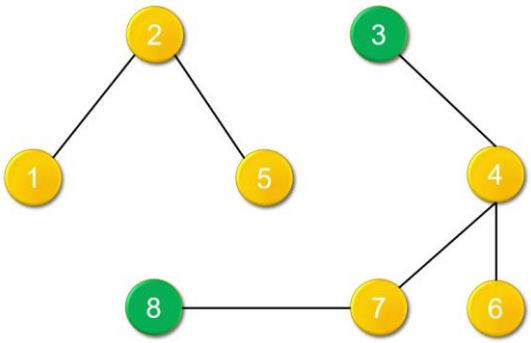
- Dòng đầu tiên chứa số 3 số nguyên: n, x, y.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản TimDuongDFS.OUT

- Dòng đầu tiên ghi số nguyên dương k là số đỉnh nằm trên đường đi từ đỉnh x đến đỉnh y (Tính luôn cả đỉnh x và y).
- Dòng thứ hai chứa k số nguyên là các đỉnh trên đường đi từ x đến y.

Ví dụ

TimDuongDFS.INP	TimDuongDFS.OUT
8 3 8 2 1 5 4 3 6 7 2 4 4 8 7	4 3 4 7 8



Buổi 6. Vận dụng thuật toán DFS

Bài 1. Đồ thị phân đôi

Cho đồ thị vô hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n. G được gọi là đồ thị phân đôi nếu chúng ta có thể chia tập đỉnh V thành 2 tập đỉnh V_1, V_2 không giao nhau sao cho các cạnh của G nối một đỉnh trong V_1 với một đỉnh trong V_2 (không có cạnh nối hai đỉnh trong cùng một tập đỉnh). Hãy kiểm tra đồ thị G có thể phân đôi không.

Dữ liệu vào: File văn bản PhanDoi.INP

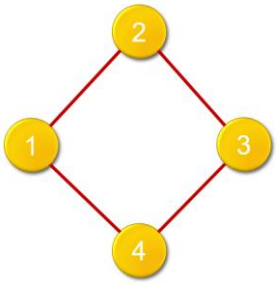
- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cạnh (i, j) của đồ thị.

Dữ liệu ra: File văn bản PhanDoi.OUT

- Dòng duy nhất ghi chữ "YES" nếu G là đồ thị phân đôi, ngược lại ghi "NO".

Ví dụ

PhanDoi.INP	PhanDoi.OUT
4 2 4 1 3 2 4 1 3	YES



Bài 2. Kiểm tra chu trình

Cho đồ thị có hướng $G = (V, E)$ có n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n. Hãy kiểm tra đồ thị G có chu trình không.

Dữ liệu vào: File văn bản ChuTrinh.INP

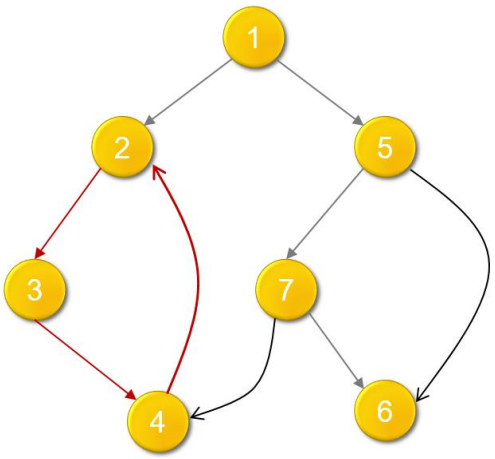
- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cung (i, j) của đồ thị.

Dữ liệu ra: File văn bản ChuTrinh.OUT

- Dòng duy nhất ghi chữ "YES" nếu G là đồ thị chứa chu trình, ngược lại ghi "NO".

Ví dụ

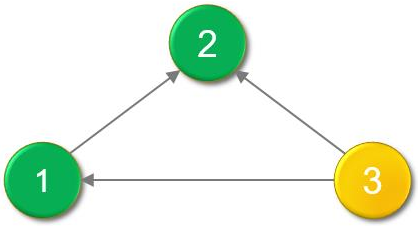
ChuTrinh.INP	ChuTrinh.OUT
7 2 5 3 4 2 6 7 4 6	YES



Bài 3. Sắp xếp topo

Cho đồ thị có hướng, không có chu trình $G = (V, E)$ gồm n đỉnh ($n \leq 10^5$), các đỉnh được đánh số từ 1 đến n. Hãy sắp xếp các đỉnh sao cho nếu có cung (u, v) thì u phải đứng trước v trong thứ tự đó.

Ví dụ 1

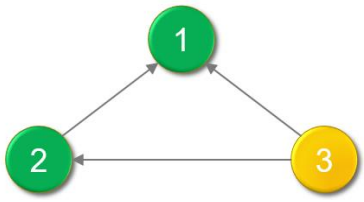


Ta có các cung

- 3 → 1: nên 3 phải nằm trước 1
- 3 → 2: nên 3 phải nằm trước 2
- **1 → 2**: nên 1 phải nằm trước 2

Một cách sắp xếp các đỉnh của đồ thị (sắp xếp topo): 3 → 1 → 2

Ví dụ 2



Ta có các cung

- 3 → 1: nên 3 phải nằm trước 1
- 3 → 2: nên 3 phải nằm trước 2
- **2 → 1**: nên 2 phải nằm trước 1

Một cách sắp xếp các đỉnh của đồ thị (sắp xếp topo): 3 → 2 → 1

Dữ liệu vào: File văn bản TopoSort.INP

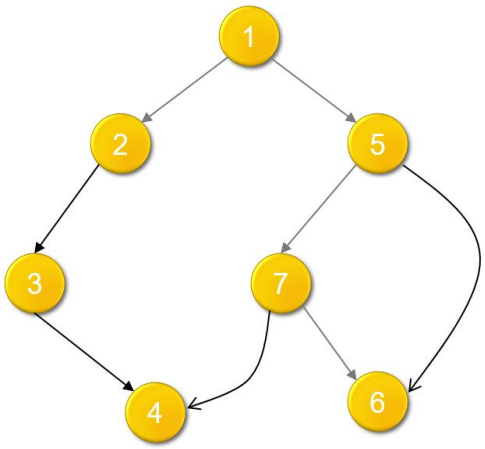
- Dòng đầu tiên chứa số đỉnh n của đồ thị.
- n dòng tiếp theo, dòng thứ i chứa một danh sách các đỉnh, mỗi đỉnh j trong danh sách tương ứng với một cung (i, j) của đồ thị.

Dữ liệu ra: File văn bản TopoSort.OUT

- Dòng duy nhất chứa các đỉnh của đồ thị được sắp topo.

Ví dụ

TopoSort.INP	TopoSort.OUT
7 2 5 3 4 6 7 4 6	1 5 7 6 2 3 4



Buổi 7. Tìm đường đi ngắn nhất

Bài 1. Đường đi ngắn nhất

Cho đồ thị vô hướng có trọng số $G = (V, E, w)$ và có n đỉnh, các đỉnh được đánh số từ 1 đến n và hai đỉnh s, t $(s, t \in V)$. Hãy tìm đường đi ngắn nhất từ đỉnh s đến đỉnh t theo thuật toán Dijkstra.

Dữ liệu vào: File văn bản Dijkstra.INP

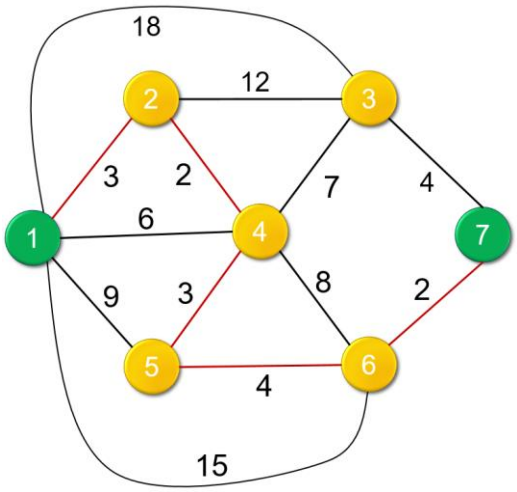
- Dòng đầu tiên chứa 4 số nguyên $n,\ m,\ s, t$ (tương ứng với số đỉnh ($n \leq 10^5$), số cạnh ($m \leq 10^5$) và 2 đỉnh $s,\ t$ của đồ thị).
- m dòng tiếp theo, mỗi dòng chứa 3 số u, v, w mô tả cung (u, v) có trọng số w $(0 \leq w \leq 10^4)$.

Dữ liệu ra: File văn bản Dijkstra.OUT

- Dòng thứ nhất ghi một số nguyên là độ dài đường đi ngắn nhất tìm được
- Dòng thứ hai ghi các đỉnh của đường đi từ đỉnh s đến đỉnh t (bao gồm cả 2 đỉnh s, t)

Ví dụ

Dijkstra.INP	Dijkstra.OUT
7 13 1 7 1 2 3 1 3 18 1 4 6 1 5 9 1 6 15 2 3 12 2 4 2 3 4 7 3 7 4 4 5 3 4 6 8 5 6 4 6 7 2	14 1 2 4 5 6 7



Bài 2. Đường đi ngắn nhất qua đỉnh trung gian

Cho đồ thị vô hướng có trọng số $G = (V, E, w)$ gồm n đỉnh, các đỉnh được đánh số từ 1 đến n và ba đỉnh s, t và x $(s, t, x \in V)$. Hãy tìm đường đi ngắn từ đỉnh s đến đỉnh t và đường đi đó phải đi qua đỉnh x.

Dữ liệu vào: File văn bản NganNhatX.INP

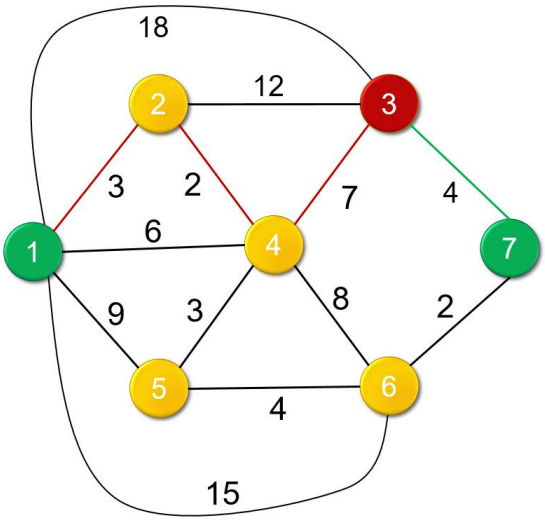
- Dòng đầu tiên chứa 5 số nguyên $n,\ m,\ s, t, x$ (tương ứng với số đỉnh ($n \leq 10^5$), số cạnh ($m \leq 10^5$) và 3 đỉnh $s,\ t, x$ của đồ thị).
- m dòng tiếp theo, mỗi dòng chứa 3 số u, v, w mô tả cung (u, v) có trọng số w $(0 \leq w \leq 10^4)$.

Dữ liệu ra: File văn bản NganNhatX.OUT

- Dòng thứ nhất ghi một số nguyên là độ dài đường đi ngắn nhất tìm được
- Dòng thứ hai ghi các đỉnh của đường đi từ đỉnh s đến đỉnh t đi qua đỉnh x (bao gồm cả 2 đỉnh s, t)

Ví dụ

NganNhatX.INP	NganNhatX.OUT
7 13 1 7 3 1 2 3 1 3 18 1 4 6 1 5 9 1 6 15 2 3 12 2 4 2 3 4 7 3 7 4 4 5 3 4 6 8 5 6 4 6 7 2	**16** **1 2 4 3 7**



Bài 3. Đường đi ngắn nhất giữa các cặp đỉnh

Cho đồ thị vô hướng có trọng số $G = (V, E, w)$ gồm n đỉnh, các đỉnh được đánh số từ 1 đến n. Hãy tìm đường đi ngắn nhất giữa các cặp đỉnh theo thuật toán Floyd – Warshall (tức là tìm ma trận $dist[i, j]$ là độ dài đường đi ngắn nhất từ đi từ đỉnh i đến đỉnh j).

Dữ liệu vào: Đọc từ file FloydWarshall.INP

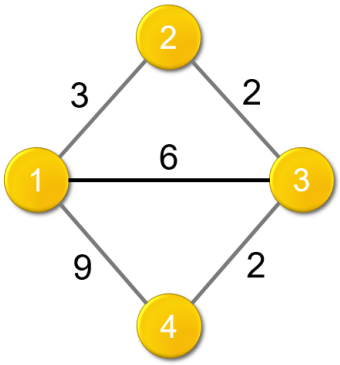
- Dòng đầu tiên chứa 1 số nguyên n (số đỉnh của đồ thị)
- n dòng sau, mỗi dòng chứa n số nguyên mô tả ma trận trọng số của đồ thị

Dữ liệu ra: Ghi ra file FloydWarshall.OUT

- Dòng đầu tiên chứa 1 số nguyên n (số đỉnh của đồ thị)
- n dòng sau, mỗi dòng chứa n số nguyên là ma trận $dist[i, j]$

Ví dụ

FloydWarshall.INP	FloydWarshall.OUT
4 0 3 6 9 3 0 2 0 6 2 0 2 9 0 2 0	4 0 3 5 7 3 0 2 4 5 2 0 2 7 4 2 0



Buổi 8. Vận dụng thuật toán tìm đường đi ngắn nhất

Bài 1. Đi ra biên

Cho bảng kích thước $n \times m$ $(n, m \leq 1000)$ các số tự nhiên. Từ một ô có thể di chuyển sang một ô kề cạnh với nó, nhưng không được đi ra ngoài bảng. Hãy tìm một cách đi từ ô (x, y) cho trước đến một ô biên sao cho tổng số của các ô đi qua là nhỏ nhất.

Dữ liệu vào: File văn bản RaBien.INP

- Dòng đầu tiên chứa các số n, m, x, y.
- n dòng tiếp theo, mỗi dòng chứa m số.

Dữ liệu ra: File văn bản RaBien.OUT

- Dòng duy nhất chứa tổng giá trị các số của các ô đã đi qua.

Ví dụ

RaBien.INP	RaBien.OUT
7 6 2 2 19 24 23 18 20 16 23 01 01 05 01 16 17 16 01 13 01 20 18 01 01 17 04 21 14 20 17 04 02 28 01 02 01 02 01 59 14 04 09 60 24 18	22

Bài 2. Chọn thành phố để tổ chức họp

Có n $(n \leq 100)$ thành phố được đánh số từ 1 đến n. Khoảng cách giữa hai thành phố i và j là a_{ij}. Người ta muốn tổ chức một cuộc họp quy tụ n lãnh đạo thành phố. Hãy tìm một thành phố để tổ chức cuộc họp sao cho khoảng cách của người đi xa nhất là nhỏ nhất có thể.

Dữ liệu vào: File văn bản ChonThanhPho.INP

- Dòng đầu tiên chứa số n.
- n dòng tiếp theo, mỗi dòng gồm n số nguyên mô tả giá trị a_{ij}.

Dữ liệu ra: File văn bản ChonThanhPho.OUT

- Dòng đầu là thành phố đăng cai tổ chức.

- Dòng thứ hai, thời gian của người phải đi xa nhất.

Bài 3. Đường tròn

Trên mặt phẳng cho n đường tròn, các đường tròn được đánh số từ 1 đến n. Đường tròn thứ i được cho bởi bộ ba số thực (x_i, y_i, r_i), trong đó (x_i, y_i) là toạ độ của tâm đường tròn và r_i là bán kính của đường tròn. Một đối tượng nằm trong đường tròn có thể di chuyển tự do trong đường tròn đó với chi phí bằng 0. Nhưng để di chuyển đối tượng đến vị trí ngoài đường tròn thì trước tiên phải di chuyển đường tròn đang chứa đối tượng đến đường tròn nào đó chứa điểm cần đến. Chi phí di chuyển giữa hai đường tròn bằng khoảng cách giữa chúng. Một đối tượng đang ở trong đường tròn s, hãy tìm phương án di chuyển đối tượng đến đường tròn t sao cho tốn chi phí ít nhất.

Dữ liệu vào: File văn bản DuongTron.INP

- Dòng đầu tiên chứa ba số nguyên n, s, t.
- n dòng tiếp theo, dòng i chứa ba số nguyên x_i, y_i, r_i.

Dữ liệu ra: File văn bản DuongTron.OUT

- Dòng duy nhất chứa chi phí ít nhất tìm được (xuất 2 số lẻ thập phân)

Bài 4. Đến trường

Gia đình Tuấn sống ở thành phố XYZ. Hàng ngày, mẹ đi ô tô đến cơ quan làm việc còn Tuấn đi bộ đến trường học. Thành phố XYZ có N nút giao thông được đánh số từ 1 đến N. Nhà Tuấn nằm ở nút giao thông 1, trường của Tuấn nằm ở nút giao thông K, cơ quan của mẹ nằm ở nút giao thông N. Từ nút đến nút có không quá một đường đi một chiều, tất nhiên, có thể có đường đi một chiều khác đi từ nút đến nút. Nếu từ nút đến nút có đường đi thì thời gian đi bộ từ nút đến nút hết a_{ij} phút, còn đi ô tô hết b_{ij} $(0 < b_{ij} \leq a_{ij})$ phút.

Hôm nay, Mẹ và Tuấn xuất phát từ nhà lúc 7 giờ. Tuấn phải có mặt tại trường lúc 7 giờ 59 phút để kịp vào lớp học lúc 8 giờ. Tuấn băn khoăn không biết có thể đến trường đúng giờ hay không, nếu không Tuấn sẽ phải nhờ mẹ đưa đi từ nhà đến một nút giao thông nào đó.

Yêu cầu: Cho biết thông tin về các đường đi của thành phố XYZ. Hãy tìm cách đi để Tuấn đến trường không bị muộn giờ còn mẹ đến cơ quan làm việc sớm nhất.

Dữ liệu vào: File văn bản SCHOOL.INP có dạng:

- Dòng đầu ghi ba số nguyên dương N, M, K ($3 \leq N \leq 10.000; M \leq 10^5$; $1 < K < N$), trong đó N là số nút giao thông, M là số đường đi một chiều, K là nút giao thông - trường của Tuấn.
- M dòng tiếp theo, mỗi dòng chứa 4 số nguyên dương i, j, a_{ij}, b_{ij} ($1 \leq i, j, \leq N, b_{ij} \leq a_{ij} \leq 60$) mô tả thông tin đường đi một chiều từ i đến j.

Hai số liên tiếp trên một dòng cách nhau một dấu cách. Dữ liệu bảo đảm luôn có nghiệm.

Dữ liệu ra: File văn bản SCHOOL.OUT

- Gồm một dòng chứa một số nguyên là thời gian sớm nhất mẹ Tuấn đến được cơ quan còn Tuấn thì không bị muộn học.

Ví dụ

SCHOOL.INP	SCHOOL.OUT
5 6 3 1 4 60 40 1 2 60 30 2 3 60 30 4 5 30 15 4 3 19 10 3 5 20 10	55

Giải thích: Hành trình

- Tuấn và Mẹ đi Xe hơi: $1 \rightarrow 4$: 40 phút
- Tuấn đi xe đạp: $4 \rightarrow 3$: 19 phút
- Mẹ đi xe hơi: $4 \rightarrow 5$: 15 phút
- Thời gian của Tuấn: $40 + 19 = 59$
- Thời gian của Mẹ: $40 + 15 = 55$

Buổi 9. Cây khung và Cây khung nhỏ nhất

Bài 1. Tìm cây khung

Cho đồ thị vô hướng liên thông $G = (V, E, w)$ có n đỉnh, các đỉnh được đánh số từ 1 đến n. Hãy tìm cây khung của đồ thị G theo thuật toán DFS tại đỉnh 1.

Dữ liệu vào: File văn bản CayKhung.INP

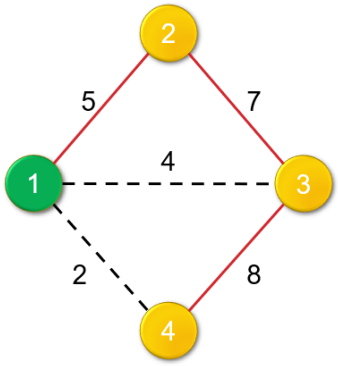
- Dòng đầu tiên chứa hai số nguyên n, m ($n, m \leq 10^5$), trong đó n là số đỉnh, m là số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa hai số u, v, w mô tả cạnh (u, v) có trọng số w trong đồ thị.

Dữ liệu ra: File văn bản CayKhung.OUT

- Dòng đầu tiên ghi số $(n - 1)$ là số cạnh trong cây khung.
- $n - 1$ dòng tiếp theo, mỗi dòng gồm ba số nguyên u, v, w mô tả cạnh (u, v) có trọng số w là một cạnh trong cây khung.

Ví dụ

CayKhung.INP	CayKhung.OUT
4 5 1 2 5 1 3 4 1 4 2 2 3 7 3 4 8	3 1 2 5 2 3 7 3 4 8



Bài 2. Tìm cây khung nhỏ nhất theo thuật toán Kruskal

Cho đồ thị vô hướng, liên thông, có trọng số $G = (V, E, w)$ gồm n đỉnh, các đỉnh được đánh số từ 1 đến n. Hãy tìm cây khung nhỏ nhất của đồ thị G theo thuật toán Kruskal và tính tổng độ dài các cạnh của cây khung tìm được.

Dữ liệu vào: File văn bản Kruskal.INP

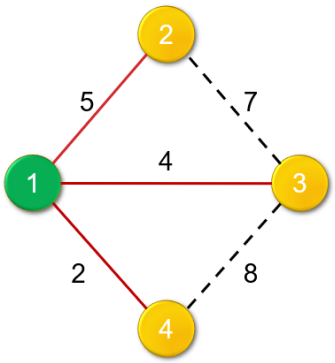
- Dòng đầu tiên chứa hai số nguyên n, m $(n, m \leq 10^5)$, trong đó n là số đỉnh, m là số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa ba số u, v, w cho biết cạnh (u, v) có trọng số w $(1 \leq w \leq 10^4)$.

Dữ liệu ra: File văn bản Kruskal.OUT

- Dòng đầu tiên ghi số hai số tương ứng là số cạnh và tổng trọng số của cây khung nhỏ nhất.
- $n - 1$ dòng tiếp theo, mỗi dòng gồm ba số u, v, w cho biết cạnh (u, v) là cạnh trong cây khung nhỏ nhất có trọng số w.

Ví dụ

Kruskal.INP	Kruskal.OUT
4 5 1 2 5 1 3 4 1 4 2 2 3 7 3 4 8	3 11 1 4 2 1 3 4 1 2 5



Bài 3. Prim

Cho đồ thị vô hướng có trọng số $G = (V, E, w)$ có n đỉnh, các đỉnh được đánh số từ 1 đến n. Hãy tìm cây khung nhỏ nhất của đồ thị G theo thuật toán Prim bắt đầu từ đỉnh 1 và tính tổng độ dài các cạnh của cây khung tìm được.

Dữ liệu vào: File văn bản Prim.INP

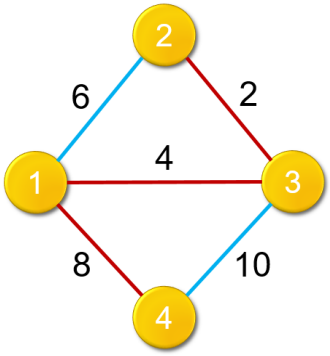
- Dòng đầu tiên chứa hai số nguyên n, m $(n, m \leq 10^5)$, trong đó n là số đỉnh, m là số cạnh của đồ thị.
- m dòng tiếp theo, mỗi dòng chứa ba số u, v, w cho biết cạnh (u, v) có trọng số w $(1 \leq w \leq 10^4)$.

Dữ liệu ra: File văn bản Prim.OUT

- Dòng đầu tiên ghi số hai số tương ứng là số cạnh và tổng trọng số của cây khung nhỏ nhất.
- $n - 1$ dòng tiếp theo, mỗi dòng gồm ba số u, v, w cho biết cạnh (u, v) là cạnh trong cây khung nhỏ nhất có trọng số w.

Ví dụ

Prim.INP	Prim.OUT
4 5 1 2 6 1 3 4 1 4 8 2 3 2 3 4 10	3 14 1 3 4 3 2 2 1 4 8



Bài 4. Cây khung x

Cho đồ thị $G = (V, E)$ có n đỉnh ($1 \leq n \leq 10^5$), các đỉnh được đánh số từ 1 đến n, m cạnh ($1 \leq m \leq 10^5$) và độ dài x. Hãy tìm cây khung nhỏ nhất có độ dài cạnh nhỏ nhất trong cây khung lớn hơn hay bằng x.

Input: CayKhungX.INP

- Dòng thứ nhất chứa ba số n, m, x
- M dòng sau chứa bộ ba số (u_i, v_i, w_i) mô tả cạnh thứ (u_i, v_i) có trọng số w_i $(1 \leq w_i \leq 10^4)$

Output: CayKhungX.OUT

- Chứa số -1 nếu không có cây khung thỏa điều kiện bài toán, nếu có thì ghi tổng trọng số các cạnh của cây khung tìm được.

Ví dụ

CayKhungX.INP	CayKhungX.OUT
6 9 5 1 2 2 1 6 5 2 6 4 2 3 10 5 2 8 5 3 13 6 5 7 5 4 7 3 4 1	37

Bài 5. Xây dựng các con đường

Có n ngôi làng được đánh số từ 1 đến n, và bạn nên xây dựng một số con đường để mỗi hai làng có thể thông nhau với nhau. Ta nói hai làng A và B là thông nhau khi và chỉ khi có một con đường giữa A và B, hoặc tồn tại một làng C sao cho có một con đường giữa A và C và, C và B thông nhau.

Chúng ta biết rằng đã có một số con đường giữa một số làng và công việc của bạn là xây dựng một số con đường sao cho tất cả các làng được thông nhau và chiều dài của tất cả các con đường được xây dựng là tối thiểu.

Dữ liệu vào: File văn bản Roads.INP

- Dòng đầu tiên là số nguyên n $(3 \leq n \leq 100)$ là số ngôi làng.
- n dòng tiếp theo, dòng thứ i chứa chứa n số nguyên, số nguyên thứ j trong n số nguyên này là khoảng cách (giá trị khoảng cách là một số nguyên thuộc $[1,1000]$) giữa làng i và làng j.
- Sau đó là một số nguyên Q $\left(0 \leq Q \leq \frac{n(n+1)}{2}\right)$
- Q dòng tiếp theo, mỗi dòng chứa hai số nguyên a và b $(1 \leq a < b \leq n)$ cho viết con đường giữa làng a và làng b đã được xây dựng.

Output: File văn bản Roads.OUT

- Chứa một dòng duy nhất chứa một số nguyên là chiều dài của tất các con đường cần được xây sao cho tất cả các làng đều thông nhau và giá trị này là giá trị nhỏ nhất.

Ví dụ

Roads.INP	Roads.OUT
3 0 990 692 990 0 179 692 179 0 1 1 2	179

Buổi 10. Tìm đường đi, chu trình Euler

Bài 1. Kiểm tra Euler trên đồ thị vô hướng

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n. Kiểm tra đồ thị G có phải là đồ thị có chu trình Euler, hay có đường đi Euler không.

Dữ liệu vào: File văn bản EulerVoHuong.INP

- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

Dữ liệu ra: File văn bản EulerVoHuong.OUT

- Dòng duy nhất chứa số nguyên k, trong đó
  - $k = 0$: G là đồ thị không có chu trình Euler, cũng không có đường đi Euler
  - $k = 1$: G là đồ thị có chu trình Euler
  - $k = 2$: G là đồ thị có đường đi Euler

Bài 2. Kiểm tra Euler trên đồ thị có hướng

Cho đồ thị có hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n. Kiểm tra đồ thị G có phải là đồ thị có chu trình Euler, hay có đường đi Euler không.

Dữ liệu vào: File văn bản EulerCoHuong.INP

- Dòng đầu tiên chứa số nguyên n là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

Dữ liệu ra: File văn bản EulerCoHuong.OUT chứa một dòng duy nhất chứa số nguyên k, trong đó

- $k = 0$: G là đồ thị không có chu trình Euler, cũng không có đường đi Euler
- $k = 1$: G là đồ thị có chu trình Euler
- $k = 2$: G là đồ thị có đường đi Euler

Bài 3. Tìm chu trình Euler

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n. Giả sử G có chu trình Euler, hãy tìm chu trình Euler trên đồ thị G bắt đầu từ đỉnh x ($x \in V$).

Dữ liệu vào: File văn bản ChuTrinhEuler.INP

- Dòng đầu tiên chứa hai số nguyên n, x: n là số đỉnh của đồ thị và đỉnh x.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

Dữ liệu ra: File văn bản ChuTrinhEuler.OUT

- Dòng duy nhất chứa các đỉnh $x, y_1, y_2, \dots, x$ là chu trình Euler xuất phát từ đỉnh x

Bài 4. Vẽ k nét

Cho đồ thị vô hướng $G = (V, E)$ có n ($n \leq 1000$) đỉnh, các đỉnh được đánh số từ 1 đến n. Một nét vẽ trên đồ thị G là một lần đặt bút lên một đỉnh nào đó và vẽ theo các cạnh của đồ thị, mỗi cạnh chỉ được vẽ một lần, trong quá trình vẽ không được nhấc bút lên và vẽ cho đến khi nào không thể vẽ được nữa. Hãy tìm cách vẽ tất cả các cạnh của đồ thị G bằng k nét vẽ, với k là số nhỏ nhất có thể.

Dữ liệu vào: File văn bản kNet.INP

- Dòng đầu tiên chứa số nguyên $n$là số đỉnh của đồ thị.
- n dòng tiếp theo, mỗi dòng chứa n số biểu diễn ma trận kề của đồ thị.

Dữ liệu ra: File văn bản kNet.OUT

- Dòng đầu tiên là số nguyên k là số nét vẽ của đồ thị
- k dòng tiếp theo, mỗi dòng gồm các đỉnh mô tả một nét vẽ