



Chuyên đề xấp xỉ và tô pô

QUI HOẠCH TRÊN TH CỐ HẠNG, KHÔNG CHU TRÌNH

Lê Thanh Bình

THPT Chuyên Nguyễn Trãi

Giới thiệu các bài toán có nội dung thực tế là một phần quan trọng trong chương trình tin học chuyên ngành này, tôi xin trao đổi về các bài toán nghiên cứu nội dung như các lý thuyết thực tế là "Các bài toán qui hoạch trên thực cố hạng, không có chu trình".

Chuyên đề trình bày một số kinh nghiệm khi dạy về thực cố hạng không có chu trình. Một trong những nội dung khá lý thú là đây hai nội dung chính của chương trình tin học là Qui hoạch và lý thuyết thực cố hạng. Chính nội dung này cho phép xây dựng cho học sinh một cách nhìn tổng quan khi tiếp cận hai dạng toán này.

Phần lớn các ví dụ minh họa trong chuyên đề liệt kê các kết quả sinh học khác nhau. Mục đích làm như vậy là tôi muốn trao đổi về các bài toán nghiên cứu về cách xây dựng một toán thực sao cho có thể ôn tập kiến thức học sinh, và có thể bám sát các chương trình thi trong các kỳ thi sinh học tin học.

I-MỘT SỐ KHÁI NIỆM VÀ BÀI TOÁN CƠ BẢN

Như chúng ta đã biết, thực cố hạng có thể hình dung như là một tập (V, E) trong đó V là tập hợp các đỉnh (trong các bài toán tin học thì V là tập hợp hữu hạn các đỉnh có thể đánh số $1, 2, \dots, N$) còn E là tập hợp các cung cạnh.

Một thực cố hạng không có chu trình là thực cố hạng không có chu trình khép kín. Có thể hình dung đây là thực cố hạng mà số lượng đỉnh trong tập các thành phần liên thông nhỏ hơn hoặc bằng 1.

Một thực cố hạng không có chu trình luôn tồn tại một tập đỉnh topo. Chính xác hơn, một tập đỉnh topo là một cách sắp xếp các đỉnh của thực cố hạng thành một dãy

$$x_1, x_2, \dots, x_n$$

Sao cho mọi cặp $(x_i, x_j) \in E$ đều kéo theo $i < j$.

Vì cách sắp xếp topo trên thực cố hạng không có chu trình là nội dung tiên quyết làm các bài toán qui hoạch trên logic thực này. Lý



do n g i n là n u nh coi m i nh c a th là m t tr ng thái thì v i v i c s p x p topo chúng ta có m t th t trên các tr ng thái này và ây chính là cách t i p c n v n theo quan i m qui ho ch ng.

Có hai cách chính xây d ng m t s p x p topo trên th có h ng không có chu trình:

Cách th nh t: D a vào m t tiêu chí t nhiên mà n u s p x p t ng/gi m theo tiêu chí này thì ng nhiên ta có m t s p x p topo.

Ví d 1 (VOI 2008): Cho n hình tròn bán kính r_1, r_2, \dots, r_n . Ta nói t ng tròn bán kính a có th nh y t i hình tròn bán kính b n u t n t i m t hình tròn bán kính c sao cho $a+c=b$ (*). Hãy tìm ng i qua nhi u hình tròn nh t.

D nh n th y r ng i u ki n (*) ch ng t t m t hình tròn ch có th nh y n m t hình tròn có bán kính l n h n nên hi n nhiên r ng n u ta s p x p l i các hình tròn sao cho bán kính c a chúng t ng d n ta s có m t s p x p topo.

Thông th ng các tiêu chí t nhiên này th ng d th y và v i c s p x p topo qui v v i c s p x p t ng/gi m trên tiêu chí này. Do ó, hi n nhiên tiêu chí s p x p ph i d a trên d li u có m i quan h s p x p hoàn toàn (thông th ng là các s).

Cách th hai: D a vào thu t toán Tarjan tìm thành ph n liên thông m nh. Chú ý r ng khi th là không có chu trình thì các thành ph n liên thông m nh u có s l ng nh b ng 1. Do v y trong tr ng h p này ta ch c n li t kê các nh theo th t sau c a phép duy t th u tiên chi u sâu. Mã gi c a nó c v i t nh d i ây

PROCEDURE visit(u)

ánh d u u c th m

For $v \in \text{Ke}(u)$ do if (v ch a c th m) then

visit(v)

a u vào danh sách s p topo

(Có th tham kh o mã Pascal trong sách giáo khoa chuyên tin. T p 1)

Cách th hai c dùng khi không th tìm c tiêu chí t nhiên trong v i c s p x p topo. Tuy r ng ây là cách t ng quát áp d ng cho m i tr ng h p nh ng theo kinh nghi m c a tôi thì thông th ng khi s p x p topo ta hay s d ng cách th nh t h n.

Gi s trên th có h ng không có chu trình $G=(V,E)$ ta ã có m t s p x p topo x_1, x_2, \dots, x_n . Khi ó ta có hai bài toán c b n sau:



Bài toán 1: Cho m i cung c a th m t tr ng s . Hãy tìm ng i dài nh t t nh s n nh t

t $f[x_i]$ l n là dài ng i dài nh t t s n xi. Khi ó

$$f[x_i] = \max\{f[x_k] + d(x_k, x_i) : (x_k, x_i) \in E\}$$

M t i u lý thú là thay vì tính toán trên các cung ng c (các cung t i x_i) c a th theo nh cách t duy truy n th ng c a qui ho ch ng, chúng ta s s a (update) theo các cung xuôi (ây là c i m chính khi th c hi n qui ho ch ng trên DAG vì nói chung xây d ng các cung ng c là m t v n khá ph c t p):

PROCEDURE DuongDiMax

For $i \in \{1, \dots, n\}$ $f[i] = -\infty$

For $i \in \{1, \dots, n\}$

u = x[i]

if (u=s) $f[u] = 0$

if ($f[u] < -\infty$)

For $v \in Ke(u)$ if $f[v] < f[u] + d(u, v)$ then $f[v] = f[u] + d(u, v)$

Hoàn toàn t ng t ta có th tìm ng i ng n nh t

Bài toán 2: m s ng i t nh s t i nh t?

G i $f[x_i]$ là s ng i t s n x_i ta có công th c

$$f[x_i] = \sum \{f[x_k] : (x_k, x_i) \in E\}$$

Và m t l n n a ta có ch ng trình qui ho ch ng t ng t nh trên:

PROCEDURE SoDuongDi

For $i \in \{1, \dots, n\}$ $f[i] = 0$

For $i \in \{1, \dots, n\}$

u = x[i]

if (u=s) $f[u] = 1$

if ($f[u] < -\infty$)

For $v \in Ke(u)$ $f[v] = f[v] + f[u]$

Hai bài toán trên là hai bài toán c b n trong các bài toán qui ho ch ng trên th có h ng. M t l n n a nh c l i i u c bi t c a qui ho ch ng trên

th có h ng là ta tính toán theo cung c a th , do v y ta th c hi n vì c s a (update) nh n thay vì tính max, tính min ho c m nh trong qui ho ch ng thông th ng (lý do n gi n là xây d ng th ng c nói chung là khá ph c t p và t n kém)



II-M TS BÀI T P MINH H A

Bài t p 1 (VOI 2008):

Nh y lò cò là trò ch i dân gian c a Vi t Nam. Ng i trên hành tinh X c ng r t thích trò ch i này và h ã c i biên trò ch i này nh sau: Trên m t ph ng v n vòng tròn c ánh s t l n n. T i vòng tròn i ng i ta i n s nguyên d ng ai. Hai s trên hai vòng tròn tùy ý không nh t thì t ph i khác nhau. T i p n ng i ta v các m i tên, m i m i tên h ng t m t vòng tròn n m t vòng tròn khác. Quy t c v m i tên là: N u có ba s ai, aj, ak th a mãn $ak = ai + aj$ thì v m i tên h ng t vòng tròn i n vòng tròn k và m i tên h ng t vòng tròn j n vòng tròn k. Ng i ch i ch c di chuy n t m t vòng tròn n m t vòng tròn khác n u có m i tên xu t phát t m t trong s các vòng tròn, di chuy n theo cách m i tên ã v i n các vòng tròn khác. Ng i th ng cu c s là ng i tìm c cách di chuy n qua nhi u vòng tròn nh t.

Yêu c u: Hãy xác nh xem trong trò ch i mô t trên, nhi u nh t có th di chuy n c qua bao nhiêu vòng tròn.

Nh ph n tr c ã nh n xét, n u coi m i vòng tròn là m t nh c a th . Hai vòng tròn k nhau n u có th nh y tr c ti p n nhau thì ta có m t DAG và bài toán qui v tìm ng i dài nh t trên th này.

M t i m c n l u ý là ch ng trình ch y t yêu c u thì i u ki n $j \in Ke(i)$ c n th c hi n vi c tìm ki m nh phân ki m tra

Bài t p 2:

M t ông ch có 2 cái máy cày cho thuê, có N ng i nông dân ng ký thuê máy cày. Ng i th i mu n thuê máy b t ut th i i m s[i] nh t th i i m t[i]. Giá thuê m t máy cày trong m t n v th i gian m t m t ng, nh v y n u cho ng i th i thuê ông ch có th thu v c t[i]-s[i]+1 ng. T i m t th i i m m t máy có nhi u nh t m t ng i s d ng.

Yêu c u: Tính s ti n nhi u nh t có th thu c.

Input:

Dòng u là s nguyên N ($N \leq 100$)

N dòng sau, m i dòng 2 s nguyên th hi n s s[i], t[i] ($0 \leq s[i] \leq t[i] \leq 109$)

Output:

G m 1 dòng duy nh t ch a l s là s ti n l n nh t có th thu c.

Ta xây d ng th nh sau:



T p nh $V=\{(i,j): v \text{ i ý ngh a là máy th nh t làm công vi c cu i cùng là } i \text{ và máy th hai làm công vi c cu i cùng là } j\}$

T p cung $E=\{(i,j)-(i,k): n \text{ u sau khi làm } j \text{ máy th } 2 \text{ làm c công vi c } k \text{ và } (i,j)-(k,j) \text{ n u sau khi làm } i \text{ máy th nh t làm c } k\}$

D th y bài toán qui v tìm ng i dài nh t t nh $(0,0)$.

ây là DAG và m t s p x p topo t nhiên là s p x p theo th i gian k t thúc thuê máy t ng d n. Do v y ta hoàn toàn có th s d ng mô hình bài toán 1 gi i quy t:

Bài t p 3:

Cho th có h ng N nh ($N \leq 16$) trong ó các c nh có tr ng s . Hãy tìm ng i Haminton (ng i qua t t c các nh) ng n nh t?

Ta xây d ng m t th m i trong ó m i nh là m t c p g m dãy bit (b_1, b_2, \dots, b_n) v i $b_i = 1$ n u nh nh i ã i qua còn $b_i = 0$ n u nh nh i ch a i qua và nh u th hi n nh cu i cùng trên hành trình là u . Nh v y m i nh là m t c p (x, u) v i x là s nguyên th hi n dãy bit trên. nh (x, u) i n c (y, v) n u nh bit v c a x b ng 0 và bit v c a y b ng 1 (các bit khác trùng nhau) và u i n c v.

D th y r ng th xây d ng nh trên là DAG v i s p x p topo t nhiên là s p x p các nh (x, u) theo s l ng bit 1 c a x t ng d n. Khi ó trên s p x p topo này các nh c chia thành t ng l p (x có 0 bit 1, x có 1 bit 1, ..., x có n bit 1) và ta có th s d ng mô hình bài toán 1 (tìm ng i dài nh t t t p nh có 1 bit 1 n t p nh có n bit 1) v i m t chú c i ti n là k th p v i m t hàng i.

III-CÁC TH CÓ H NG KHÔNG CÓ CHU TRÌNH C M SINH

Khi d y h c sinh các thu t toán c b n nh duy t th u tiên chi u r ng, duy t th u tiên chi u sâu, tìm ng i ng n nh t trên th không có chu trình âm, c n ph i nh n m nh n các s n ph m c a các thu t toán này. M t i u r t thú v là có r t nhi u s n ph m là th b ph n có h ng không có chu trình mà tôi t m g i là các th có h ng không có chu trình c m sinh. Có r t nhi u bài t p v th trong các k thi g n ây s d ng các th b ph n này.

1. DAG ng i ít c nh nh t

Khi th c hi n duy t th u tiên chi u r ng (BFS) t nh s ta g i $d[i]$ là dài ng i ít c nh nh t t s n i ($d[i] = \infty$ n u không có ng i t s n i). Xây d ng th b ph n $G'=(V',E')$ nh sau:

$$V' \equiv V$$



$$E' = \{(u, v) \in E: d[v] = d[u] + 1\}$$

th này còn c g i là th ng i ít c nh nh t vì t t c các ng i ng n nh t (theo ngh a ít c nh nh t) u i qua các cung c a th này.

D dàng nh n th y G' là m t DAG và c bi t, s p x p topo t nhiên c a nó là s p x p theo giá tr $d[i]$ t ng d n. Nó c ng chính là th t c a các nh khi a vào/l y ra kh i hàng i trong phép duy t BFS (i u này khi n cho ta không ph i th c hi n m t phép sort y n a). Chính xác h n, n u g i x_1, x_2, \dots, x_p là các nh theo th t a vào hàng i thì ta có m t s p x p topo trên G' .

V i th G' ta có th xây d ng m t s bài toán m r ng cho BFS nh :

Bài toán 3: Hãy tìm ng i ng n nh t (ít c nh nh t) t nh s n nh t. N u nh có nhi u ng i nh v y thì tìm ng i có:

Giá thành nh nh t/l n nh t

S nh i qua nhi u nh t/ít nh t

gi i bài toán 3, tr c tiên chúng ta th c hi n BFS xây d ng th G' sau ó trên G' ta gi i bài toán tìm ng i ng n nh t/dài nh t d a trên s p x p topo c a nó (bài toán 1)

Bài toán 4: Hãy m s ng i ng n nh t t s n t?

ây chính là bài toán 2 (m s ng i) trên th G' v i m t s p x p topo

Bài t p 4 (VOI 2009):

Trong m ng xã h i, m i trang web c t ch c trên m t máy tính thành viên và cung c p d ch v truy nh p t i m t s trang web khác. truy nh p t i m t trang web nào ó không có trong danh m c k t n i tr c ti p c a mình, ng i dùng ph i truy nh p t i trang web khác có k t n i v i mình, d a vào danh m c d ch v c a trang web này chuy n t i trang web khác theo tùy ch n, c nh th cho n khi t i c trang web mình c n. Th i gian truy nh p t i m t trang web ph thu c ch y u và s l n m trang web trong quá trình truy nh p. Nh v y, ng i dùng c n ch ng ch n l trình truy nh p h p lí.

Sau m t th i gian làm vi c trên m ng, Sáng - m t thành viên nhi t thành ã tích l y kinh nghi m, t o m t c s d li u, cho bi t t m t trang web có th i t i nh ng trang web nào trong m ng. Trong c s d li u, các trang web c ánh s t l n n và có m b n ghi, m i b n ghi có d ng c p có th t (u, v) cho bi t trang web u có k t n i t i trang web v (l_u, v_n, u_v). C s d li u ch a c chu n hóa, vì v y có th ch a các c p (u, v) gi ng nhau.



Trang web có a Sáng có s hi u là s. D a vào c s d li u, Sáng có th xác nh l trình truy nh p nhanh nh t (t c là s l n ph i m trang web là ít nh t) t trang web s t i trang web u b t kì. Tuy v y, m ng xã h i, m i chuy n u có th x y ra: m t khu v c nào ó b m t i n, máy c a m t thành viên b h ng, trang web ó ang b óng nâng c p, ... K t qu là m t vài trang web nào ó có th t m th i không ho t ng. Nh v y, n u t s có ít nh t hai l trình nhanh nh t khác nhau t i u thì kh n ng th c hi n truy nh p c m t cách nhanh nh t t i u là l n h n so v i nh ng trang web ch có duy nh t m t l trình nhanh nh t. Hai l trình g i là khác nhau n u có ít nh t m t trang web có l trình này mà không có l trình kia ho c c hai l trình cùng i qua nh ng trang web nh nhau nh ng theo các trình t khác nhau. Nh ng trang web mà t s t i ó có ít ra là hai l trình nhanh nh t khác nhau c g i là n nh i v i s. Trang web mà t s không có l trình t i nó là không n nh i v i s.

Yêu c u: Cho các s nguyên d ng n, m, s và m c p s (u, v) xác nh t u có th k t n i tr c ti p t i c v. Hãy xác nh s l ng trang web n nh i v i s.

D li u:

Dòng u tiên ch a 3 s nguyên n, m và s ($2 \leq 10000$, $1 \leq m \leq 50000$, $1 \leq s \leq n$).

M i dòng trong m dòng ti p theo ch a 2 s nguyên u và v ($1 \leq u, v \leq n$, $u \neq v$).

Các s trên m t dòng c ghi cách nhau ít nh t m t d u cách.

K t qu : M t s nguyên - s trang web n nh i v i s.

Bài toán trên là bài toán m các nh có ít nh t hai ng i ng n nh t t s. Ph ng pháp gi i nó là bài toán 4 (v i m t l u ý là ta không th c s m mà ch c n phân bi t các nh có 0, 1, h n 1 ng i ng n nh t t s)

2. DAG ng i ng n nh t

Các thu t toán Dijkstra, Ford_bellman tìm ng i ng n nh t t m t nh s u cho m t m ng $\text{dist}[i]$ là kho ng cách ng n nh t t nh s n nh i ($\text{dist}[i] = \infty$ n u không có ng i t s n i). T ng t nh trên, sau khi có m ng $\text{dist}[i]$ ta có th b ph n $G' = (V', E')$ nh sau:

$$V' \equiv V$$

$$E' = \{(u, v) \in E : \text{dist}[v] = \text{dist}[u] + L(u, v)\}$$

G' c ng là DAG, DAG này là DAG ng i ng n nh t. N u s d ng thu t toán Dijkstra thì m t s p x p topo trên DAG này là th t l y ra các nh kh i hàng i u tiên, còn n u s d ng Ford_Bellman thì ta ph i th c hi n m t phép sort trên m ng dist .



C ng nh DAG ng i ít c nh nh t, chúng ta c ng có m t s bài toán đ a trên DAG này gi ng nh bài toán 3, bài toán 4. D i ây là m t s ví d i n hình:

Bài t p 5 (VOI 2007):

Trên m t m ng l i giao thông có n nút, các nút c ánh s t 1 n n và gi a hai nút b t k có không quá m t ng n i tr c ti p (ng n i tr c ti p là m t ng hai chi u). Ta g i ng i t nút s n nút t là m t dãy các nút và các ng n i tr c ti p có d ng:

$$s = u_1, e_1, u_2, \dots, u_i, e_i, u_{i+1}, \dots, u_{k-1}, e_{k-1}, u_k = t,$$

trong ó u_1, u_2, \dots, u_k là các nút trong m ng l i giao thông, e_i là ng n i tr c ti p gi a nút u_i và u_{i+1} (không có nút u_j nào xu t hi n nhi u h n m t l n trong dãy trên, $j = 1, 2, \dots, k$).

Bì t r ng m ng l i giao thông c xét luôn có ít nh t m t ng i t nút 1 n nút n.

M t robot ch a y bình v i w n v n ng l ng, c n i t tr m c u ho t t i nút 1 n n i x y ra ho ho n nút n, trong th i gian ít nh t có th . Th i gian và chi phí n ng l ng robot i trên ng n i tr c ti p t nút i n nút j t ng ng là t_{ij} và c_{ij} ($1 \leq i, j \leq n$). Robot ch có th i c trên ng n i tr c ti p t nút i n nút j n u n ng l ng còn l i trong bình ch a không ít h n c_{ij} ($1 \leq i, j \leq n$). N u robot i n m t nút có tr m ti p n ng l ng (m t nút có th có ho c không có tr m ti p n ng l ng) thì nó t ng c n p y n ng l ng vào bình ch a v i th i gian n p coi nh không áng k .

Yêu c u: Hãy xác nh giá tr w nh nh t robot i c trên m t ng i t nút 1 n nút n trong th i gian ít nh t.

Input

Dòng u tiên ch a m t s nguyên d ng n ($2 \leq n \leq 500$);

Dòng th hai ch a n s , trong ó s th j b ng l ho c 0 t ng ng nút j có ho c không có tr m ti p n ng l ng ($j = 1, 2, \dots, n$);

Dòng th ba ch a s nguyên d ng m ($m \leq 30000$) là s ng n i tr c ti p có trong m ng l i giao thông;

Dòng th k trong s m dòng ti p theo ch a 4 s nguyên d ng i, j, t_{ij} , c_{ij} ($t_{ij}, c_{ij} \leq 10000$) mô t ng n i tr c ti p t nút i n nút j, th i gian và chi phí n ng l ng t ng ng.

Hai s liên ti p trên m t dòng trong file đ li u cách nhau ít nh t m t đ u cách.

Output: Ghi ra s nguyên d ng w tìm c.



Trước tiên sẽ dùng thuật toán Dijkstra chúng ta tìm được DAG ngắn nhất. Một lần nữa chú ý rằng sắp xếp topo trên DAG này chính là thứ tự ra các nhánh hàng đầu tiên. Trên DAG ngắn nhất này ta giải bài toán tìm ngắn nhất thứ tự. Kết quả dùng đây có thể là tìm kiếm nhánh phân và ta đi đến bài toán cơ bản "Cho ngắn nhất x , hỏi rằng vì ngắn nhất này có thể đi đến c hay không?" bài toán này hoàn toàn giải bằng qui hoạch động.

Bài tập 6 (IOICamp maraton 2006):

Ngày 27/11 tôi là ngày tổ chức thi học kỳ I trường H BK. Là sinh viên năm thứ nhất, Hi u không muốn vì điểm mà gặp trực tiếp phòng thi nên đã chuẩn bị khá kỹ càng. Chỉ còn lại một công việc khá gay go là Hi u không biết đi đường nào thì đi đường là nhanh nhất.

Thế nhưng ngày Hi u không quan tâm thì vấn đề này lại làm cho nên bây giờ Hi u không biết phải làm sao cả. Bên thành phố là ga có N nút giao thông và M con đường nối các nút giao thông này. Có 2 loại con đường là đường 1 chiều và đường 2 chiều. Chiều dài của mỗi con đường là một số nguyên dương.

Nhà Hi u nút giao thông 1 còn trường H BK nút giao thông N. Vì một lần trình bạn đi từ nhà Hi u thì đi đường có thể gặp nhiều yếu tố khác nhau là gặp nhiều đèn đỏ, đi qua công trường xây dựng, ... phải giảm tốc độ cho nên Hi u muốn biết là có tất cả bao nhiêu lần trình bạn đi từ nhà thì đi đường. Bên đây là trình giúp Hi u giải quyết bài toán khó này.

Input

Dòng đầu tiên ghi hai số nguyên N và M.

M dòng tiếp theo, mỗi dòng ghi 4 số nguyên dương K, U, V, L. Trong đó:

K = 1 có nghĩa là có đường 1 chiều từ U đến V với độ dài L.

K = 2 có nghĩa là có đường 2 chiều giữa U và V với độ dài L.

Output: Ghi hai số là độ dài ngắn nhất và số lần trình bạn đi ngắn nhất. Bị trừ số lần trình bạn đi ngắn nhất không vượt quá phạm vi int64 trong pascal hay long long trong C++.

Đầu tiên chúng ta xây dựng DAG ngắn nhất (bằng thuật toán Dijkstra). Bài toán quy về bài toán ngắn nhất trên DAG này. Đây chính là bài toán 3

Bài tập 7 (IOICAMP4):



Theo th ng kê cho bi t m c t ng tr ng kinh t c a n c Peace trong n m 2006 r t áng kh quan. C n c có t ng c ng N thành ph l n nh c ánh s tu n t t l n N phát tri n khá ng u. Gi a N thành ph này là m t m ng l i g m M ng i hai chi u, m i tuy n ng n i 2 trong N thành ph sao cho không có 2 thành ph nào c n i b i quá 1 tuy n ng. Trong N thành ph này thì thành ph l và thành ph N là 2 trung tâm kinh t l n nh t n c và h th ng ng m b o luôn có ít nh t m t cách i t thành ph l n thành ph N.

Tuy nhiên, c 2 trung tâm này u có d u hi u quá t i v m t dân s . Vì v y, c vua Peaceful quy t nh ch n ra thêm m t thành ph n a u t thành m t trung tâm kinh t th ba. Thành ph này s t m ng ng m i ho t ng th ng nh t, c ng nh m i lu ng l u thông ra vào t i n hành nâng c p c s h t ng. Nh ng trong th i gian s a ch a y, ph i b o m ng i ng n nh t t thành ph l n thành ph N không b thay i, n u không n n kinh t qu c gia s b trì tr .

V trí và ng n i gi a N thành ph c mô t nh m t th N nh M c nh. Hãy giúp nhà vua m s l ng thành ph có th ch n làm trung tâm kinh t th ba sao cho thành ph c ch n th a mãn các i u ki n trên

Input

Dòng u tiên ghi 2 s nguyên d ng N và M là s thành ph và s tuy n ng.

Dòng th i trong s M dòng ti p theo ghi 3 s nguyên d ng xi, yi và di v i ý ngh a tuy n ng th i có dài di và n i gi a 2 thành ph xi, yi.

Output:

Dòng u tiên ghi s t nhiên S là s l ng các thành ph có th ch n làm trung tâm kinh t th ba.

S dòng ti p theo, m i dòng ghi 1 s nguyên d ng là s th t c a thành ph c ch n (In ra theo th t t ng d n)

M t thành ph c ch n là thành ph mà khi rút nó ra kh i th không nh h ng n s l ng ng i ng n nh t t l n n.

t f[u] là s l ng ng i ng n nh t t l n u và g[u] là s l ng ng i ng n nh t t u n n (hai m ng này có th tính trên các DAG ng i ng n nh t c a th xuôi và th ng c). u là thành ph c ch n khi $f[u]*g[u]<f[n]$

3. DAG Liên thông m nh



Khi tìm thành phần liên thông mạnh nhất của đồ thị có hướng là đồ thị các thành phần liên thông mạnh trong đó mỗi đỉnh của đồ thị này là một thành phần liên thông mạnh của đồ thị ban đầu và thành phần liên thông A khác với thành phần liên thông B nếu chúng có chung các đỉnh của đồ thị ban đầu ít nhất một đỉnh của A nằm trong đỉnh của B.

Định lý thứ nhất các thành phần liên thông mạnh là một DAG (vì nếu không ta có thể mở rộng một thành phần liên thông mạnh nào đó) đây là DAG liên thông mạnh.

DAG liên thông mạnh có một số tính chất topo tự nhiên là thuật tìm kiếm các thành phần liên thông mạnh trong thuật toán Tarjan (thành phần liên thông mạnh nào tìm kiếm trước thì xếp trước, thành phần liên thông mạnh nào tìm kiếm sau thì xếp sau).

DAG liên thông mạnh phải xây dựng riêng (bằng một vòng lặp duyệt qua các cung của đồ thị). Hơn nữa, ta cần lưu thêm các thông tin về mỗi đỉnh của đồ thị này.

Bài tập 8:

Tất cả các ngân hàng trong thành phố của Siruseri đều là một chi nhánh. Theo luật của quốc gia này, tất cả các giao dịch phải có một máy ATM. Điều đáng ngạc nhiên là các cửa hàng chỉ in tiền bằng các giao dịch, tuy nhiên, không phải tất cả các giao dịch nào có cửa hàng chỉ in tiền.

Banditji là một tên trộm nổi tiếng. Hắn quy tụ nhóm làm một vụ ngân hàng kho bạc chính trị trong các máy ATM trên thành phố, sau đó ghé vào một cửa hàng chỉ in tiền để nghỉ ngơi. Nhờ có mạng lưới thông tin rộng rãi, Banditji biết các vị trí có máy ATM ngày hôm đó. Xuất phát từ trung tâm, tên trộm lái xe đi dọc theo các phố, vét sạch tiền các ATM gặp trên thành phố. Banditji có thể đi lên hoặc xuống trên một số đoạn phố, nhưng không thu gì cả thêm từ các ATM ở bất kỳ kho bạc nào đó. Lịch trình của Banditji phải kết thúc giao dịch có cửa hàng chỉ in tiền. Banditji biết cách vạch lộ trình để ngân sách tiền trộm của hắn là lớn nhất.

Yêu cầu: Cho biết tiền – số giao dịch, m – số đoạn phố có 2 giao dịch, p – số giao dịch có cửa hàng chỉ in tiền và các vị trí có cửa hàng, ai – số vị trí trong ATM từ giao dịch i, s – giao dịch trung tâm. Hãy xác định tổng số tiền bắt được (n, m ≤ 500 000, 0 ≤ ai ≤ 4 000).

Dữ liệu: Vào từ file văn bản ATM.INP:

Đòng đầu tiên chứa 2 số nguyên n và m,



Mỗi dòng trong mỗi dòng tiếp theo chứa 2 số nguyên u và v xác định hai cạnh giao nhau tại giao điểm,

Dòng thứ i trong n dòng tiếp theo chứa số nguyên ai,

Dòng thứ n+m+2 chứa 2 số nguyên s và p,

Dòng cuối cùng chứa p số nguyên xác định các giao điểm có các hàng chẵn lẻ.

Kết quả: Trả ra file văn bản ATM.OUT mỗi số nguyên – số điểm bất biến.

Sử dụng Tarjan chúng ta tìm được DAG các thành phần liên thông mạnh. Vì mỗi đỉnh (tức là mỗi thành phần liên thông mạnh) chúng ta lưu hai thông tin: tổng số điểm trong các trường ATM và số các hàng lẻ.

Bài toán trở thành tìm kiếm đỉnh có tổng điểm lẻ nhất trong các đỉnh có số các hàng lẻ lẻ nhất hoặc không. Do là DAG và có sắp xếp topo nên bài này có thể làm bằng quy hoạch động từ trên xuống.

Có thể thay DAG cho mỗi bài toán khá phong phú và đa dạng trên thực tế. Các DAG có thể sinh ra trên các thuật toán cơ bản như BFS, Dijkstra, Tarjan có lẽ là những DAG thú vị nhất. Bài này làm cho việc giải quyết các bài toán trên các DAG này là dễ dàng chính là do các sắp xếp topo tự nhiên mà các thuật toán cơ bản mang lại.

Đề quan tâm đến ý nghĩa thì khai thác hết các kết quả của các thuật toán là một thói quen tốt cần xây dựng cho học sinh nhằm làm tốt các bài tập. Nếu các em có kiến thức này thì việc áp dụng các thuật toán một cách uyển chuyển là một thói quen hiển nhiên.

Trên đây là một vài kinh nghiệm muốn trao đổi với các bạn đồng nghiệp. Rất mong các bạn đóng góp ý kiến. Kết thúc, xin trích hai câu cuối trong "Truyện Kiều" của Nguyễn Du:

"Lời quê chắp nhặt dông dài
Mua vui cũng là một chuyện đáng!"



Chuyên x p lo i A



























Chuyên x p lo i A

Nhóm GV Tin tr ng THPT chuyên Lào Cai

**CHUYÊN
NG ING N NH T TRÊN TH
A. M U**

1. Lý do ch n tài

Lý thuyết th là m t l nh v c c phát tri n t r t lâu, c nhi u nhà khoa h c quan tâm nghiên c u nó có vai trò h t s c quan tr ng trong nhi u l nh v c. Trong Tin h c lý thuyết th c ng d ng m t cách r ng rãi có r t nhi u thu t toán c nghiên c u và ng d ng. Trong ch ng trình môn Tin h c c a THPT chuyên ph n lý thuyết th nói chung và các thu t toán tìm ng i ng n nh t trên th là n i dung r t quan tr ng, trong các k thi h c sinh gi i xu t hi n r t nhi u các bài toán liên quan n vi c tìm ng i ng n nh t trên th . Tuy nhiên trong qua trình gi ng d y tôi th y h c sinh v n còn khó kh n trong vi c phân tích bài toán có th áp d ng c thu t toán và cài t gi i bài toán. Vì v y tôi ch n chuyên này giúp h c sinh có cái nhìn t ng quan h n v các thu t toán tìm ng i ng n nh t trên th .

2. M c ích c a tài

V n i dung ki n th c c a các thu t toán tìm ki m trên th ã có r t nhi u tài li u c p n, trong chuyên này tôi ch t ng h p l i các n i dung ki n th c ã có và a vào áp d ng gi i m t s bài toán c th , làm tài li u tham kh o cho h c sinh và giáo viên trong quá trình h c t p và gi ng d y.

A. N I DUNG

I, Gi i thi u bài toán ng i ng n nh t

- Trong th c t có r t nhi u các bài toán ch ng h n nh trong m ng l i giao thông n i gi a các Thành Ph v i nhau, m ng l i các ng bay n i các n c v i nhau ng i ta không ch quan tâm tìm ng i gi a các a i m v i nhau mà ph i l a ch n m t hành trình sao cho ti t ki m chi phí nh t (chi phí có th là th i gian, ti n b c, kho ng cách...). Khi ó ng i ta gán cho m i c nh c a th m t giá tr ph n ánh chi phí i qua c nh ó và c g ng tìm ra m t hành trình i qua các c nh v i t ng chi phí th p nh t.
- Ta i xét m t th có h ng $G = (V, E)$ v i các cung c gán tr ng s (tr ng s ây là chi phí). N u gi a hai nh u, v không có c nh n i thì ta



có thể thêm vào c nh “gi ” v i tr ng s $a_{ij} = +\infty$. Khi ó th G có th gi
thi t là th y .

- N u dãy v_0, v_1, \dots, v_p là m t ng i trên G thì dài c a nó c nh
ngh a là t ng các tr ng s trên các cung c a nó: $\sum_{i=1}^p a(v_{i-1}, v_i)$
- Bài toán t ra là tìm ng i có dài nh nh t t m t nh xu t phát $s \in V$
n nh ích $t \in V$. ng i nh v y g i là ng i ng n nh t t s n t và
dài c a nó ta còn g i là kho ng cách t s n t, kí hi u $d(s, t)$.
- Nh n xét:
 - + Kho ng cách gi a hai nh c a th có th là s âm.
 - + N u nh không t n t i ng i t s n t thì ta s t $d(s, t) = +\infty$.
 - + N u nh trong th , m i chu trình u có dài d ng thì ng i ng n
nh t s không có nh nào b l p l i. ng i không có nh l p l i g i là
ng i c b n. Còn n u trong th có ch a chu trình v i dài âm (g i
là chu trình âm) thì kho ng cách gi a m t s c p nh nào ó c a th là
không xác nh, b i vì b ng cách i vòng theo chu trình này m t s l n
l n, ta có th ch ra ng i gi a các nh này có dài nh h n b t kì s
th c nào cho tr c. Trong nh ng tr ng h p nh v y ta có th t v n
tìm ng i c b n ng n nh t.
 - + Trong th c t , bài toán tìm ng i ng n nh t gi a hai nh c a m t th
liên thông có m t ý ngh a to l n. Nhi u bài toán có th d n v bài toán trên.
Ví d bài toán ch n m t hành trình ti t ki m nh t (theo tiêu chu n kho ng
cách ho c th i gian ho c chi phí) trên m t m ng giao thông ng b ,
ng thu ho c ng không. Bài toán l p l ch thi công các công o n
trong m t công trình l n, bài toán l a ch n ng truy n tin v i chi phí nh
nh t trong m ng thông tin, ... Hi n nay có r t nhi u ph ng pháp gi i bài
toán trên. Trong bài này ta xét các gi i thu t c xây d ng trên c s lý
thuy t th t ra là hi u qu cao nh t.

II, ng i ng n nh t xu t phát t m t nh

- 1, Bài toán tìm ng i ng n nh t xu t phát t m t nh c phát bi u nh
sau : Cho th có tr ng s $G=(V,E,w)$ hãy tìm ng i ng n nh t t nh
xu t phát s n các nh còn l i c a th . dài ng i t nh s n
nh t kí hi u là (s,t) g i là kho ng cách t s t i t n u nh không t n t i
kho ng cách t s t i t thì ta t kho ng cách ó là $+$.



2. Gi i thu t Ford-Bellman

- Thu t toán Ford – Bellman có th dùng tìm ng i ng n nh t xu t phát t m t nh s thu c V trong tr ng h p th $G=(V,E,w)$ không có chu trình âm thu t toán nh sau:
- + G i $d[v]$ là kho ng cách t nh s n nh $v \in V$, $d[v]=0$, $d[t]=+$. Sau ó ta th c hi n phép co t c là m i khi phát hi n $d[u] + a[u, v] < d[v]$ thì c n trên $d[v]$ s c t t lên $d[v] := d[u] + a[u, v]$. Quá trình trên k t thúc khi nào ta không th làm t t thêm c b t c c n trên nào. Khi ó giá tr c a m i $d[v]$ s cho kho ng cách t s n v . Nói riêng $d[t]$ là dài ng n nh t gi a hai nh s và t .

Cài t thu t toán

Procedure Ford_Bellman ;

Begin

For $i := 1$ to n do

begin

$d[i] := \text{maxint}$;

$tr[i] := \text{maxint}$;

end ;

$d[s] := 0$;

Repeat

$Ok := \text{true}$;

For $i := 1$ to n do

if $d[i] < \text{maxint}$ then

for $j := 1$ to n do

if $(a[i,j] < 0) \text{ and } (d[i] + a[i,j] < d[j])$ then

begin

$ok := \text{false}$;

$d[j] := d[i] + a[i,j]$;

$tr[j] := i$;

end;

until ok ;

Nh n xét:

- Vì c ch ng minh tính úng n c a gi i thu t trên d a trên c s nguyên lý t i u c a quy ho ch ng.



- ph c t p tính toán c a gi i thu t Ford-Bellman là $O(n^3)$.
- Th c ch t c a thu t toán này là thu t toán Quy Ho ch ng trong ó , $d[i]$ là m ng dài ng n nh t i t s n i . v y n u t là nh c n thi t thì $d[t]$ là dài c n tìm . Còn n u m u n l u l i ng i thì chúng ta dùng m ng Tr $[i]$ i ng c l i .

3, Thu t toán Dijkstra

Trong tr ng h p th $G=(V,E,w)$ có tr ng s trên các cung không âm thu t toán Dijkstra c p d i ây ho t ng hi u qu h n nhi u so v i thu t toán Ford – Bellman. Thu t toán Dijkstra nh sau:

B c 1: Kh i t o

V i nh $v \in V$, ta g i $d[v]$ là dài ng i t s t i v ban u kh i t o $d[v]=0, d[t]=+\infty, \forall v \neq s$. Nh n c a m i nh có hai tr ng thái t do hay c nh, nh n t do có ngh a là có th t i u c n a, nh n c nh $d[v]$ là ng i ng n nh t t s t i v nên không th t i u c n a. Ta dùng thêm m t m ng Free[] ánh d u n u $d[v]$ là t do thì $Free[v]=True$, ng c l i $Free[v]=False$. Ban u các nh n u t do.

B c 2: L p

B c l p g m hai thao tác :

- C nh nh n: ch n trong các nh có nh n t do l y ra nh u có $d[u]$ nh nh t và c nh $d[u]$
- S a nh n: dùng nh u xét t t c các nh v và s a l i các nh n $d[v]$ theo công th c sau:

$$d[v] = \min(d[v], d[u] + c[u, v])$$

B c l p s k t thúc khi mà nh t(nh ích) ã c c nh nh n.

B c 3: K t h p v i l u v t ng i trên t ng b c s a nh n, thông báo ng i ng n nh t tìm c ho c cho bi t không t n t i ng i $d[t]=+\infty$.

Cài t thu t toán:

Const

MAX_N = 100;

FI = 'dijkstra.inp';

FO = 'dijkstra.out';

Var

n, nU, s, t : integer;

a : array[1..MAX_N, 1..MAX_N] of integer;

d, tr, U : array[1..MAX_N] of integer;



```
f : text;
Procedure Doc;
Var i, j : integer;
Begin
  assign(f, FI); reset(f);
  read(f, n, s, t);
  for i := 1 to n do
    for j := 1 to n do read(f, a[i, j]);
  close(f);
End;
Procedure Khoi_tao;
Var i : integer;
Begin
  for i := 1 to n do
    begin
      tr[i] := s;
      d[i] := a[s, i];
    end;
  for i := 1 to n do U[i] := i;
  U[s] := U[n];
  nU := n - 1;
End;
Function Co_dinh_nhan : integer;
Var i, j, p : integer;
Begin
  { Tim p }
  i := 1;
  for j := 2 to nU do
    if d[U[i]] > d[U[j]] then i := j;
  p := U[i];
  { Loai p ra khoi U }
  U[i] := U[nU];
  nU := nU - 1;
  Co_dinh_nhan := p;
End;
```



Procedure Sua_nhan(p : integer);

Var i, x : integer;

Begin

for i := 1 to nU do

begin

x := U[i];

if d[x] > d[p] + a[p, x] then

begin

tr[x] := p;

d[x] := d[p] + a[p, x];

end;

end;

End;

Procedure Print(i : integer);

Begin

if i = s then

begin

writeln(f, d[t]);

write(f, s);

exit;

end;

Print(tr[i]);

write(f, ' ', i);

End;

Procedure Ghi;

Begin

assign(f, FO); rewrite(f);

Print(t);

close(f);

End;

Procedure Dijkstra;

Var p : integer;

Begin

Khoi_tao;

repeat



```
p := Co_dinh_nhan;  
Sua_nhan(p);  
until p = t;  
End;  
Begin  
Doc;  
Dijkstra;  
Ghi;  
End.
```

4, Thuật toán Dijkstra và cấu trúc Heap

Cấu trúc Heap và một số phép xử lý trên Heap

* *Mô tả Heap*: Heap là một mô hình một cây nhị phân có cấu trúc sao cho giá trị khóa của mỗi nút không vượt quá giá trị khóa của hai nút con của nó (suy ra giá trị khóa tối đa của Heap là nút gốc).

* *Hai phép xử lý trên Heap*

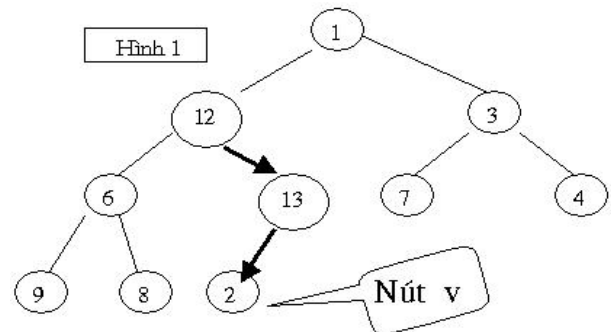
- Phép cập nhật Heap

Vấn đề: Giả sử nút v có giá trị khóa nhỏ hơn, cần chuyển nút v đến vị trí mới trên Heap để bảo toàn cấu trúc Heap

Giải quyết:

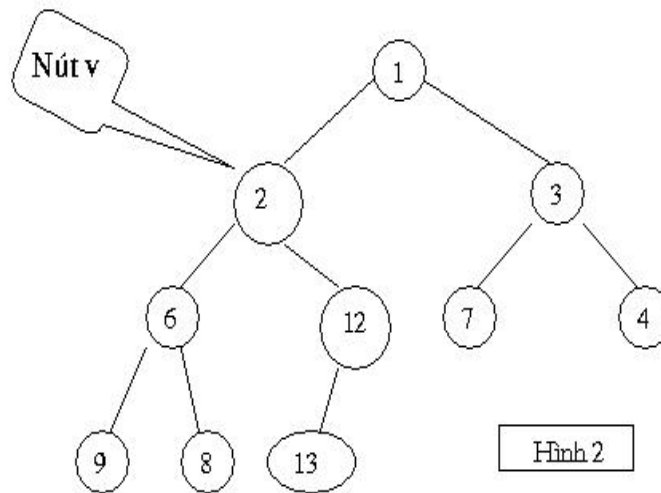
+ Nếu nút v chỉ có trong Heap thì ta thêm nút v thành nút cuối cùng của

Heap (hình 1)



+ Chuyển nút v đến vị trí hiện tại và vị trí thích hợp bằng cách tìm vị trí thích hợp để nút v đứng. Nếu nút cha có giá trị khóa lớn hơn giá trị khóa của v . Trên đường đi từ nút cha xuống nút con, nút cha cuối cùng chính là vị trí mới của nút v (hình 2).

Chú ý: trên cây nhị phân, nếu đánh số các nút từ gốc đến lá và từ trái sang phải thì ta thấy: khi biết số i của nút cha là i có thể suy ra số của hai nút con là $2*i$ và $2*i+1$, ngược lại số j của nút con là j thì số của nút cha là $j \div 2$.



- Phép lo i b g c c a Heap

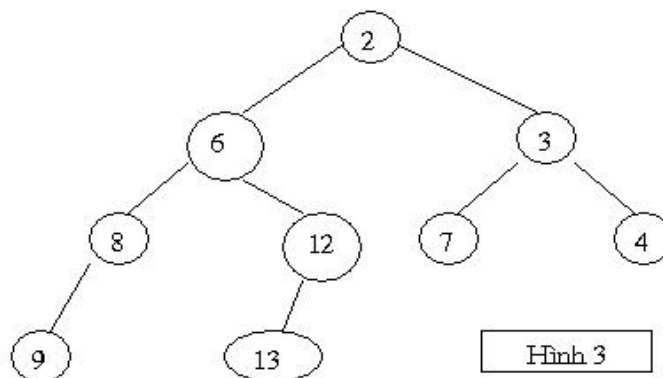
V n : Gi s c n lo i b nút g c kh i Heap, hãy s p x p l i Heap (g i là phép vun ng)

Gi i quy t:

+ Tìm ng i t g c v phía lá, i qua các nút con có giá tr khoá nh h n trong hai nút con cho n khi g p lá.

+ Trên d c ng i y, kéo nút con lên v trí nút cha c a nó.

Ví d trong hình v 2 n u b nút g c có khoá b ng 1, ta s kéo nút con lên v trí nút cha trên ng i qua các nút có giá tr khoá là 1, 2, 6, 8 và Heap m i nh hình 3



Thu t toán Dijkstra t ch c trên c u trúc Heap (t m kí hi u là Dijkstra_Heap)

T ch c Heap: Heap g m các nút là các nh i t do (ch a c nh nh n ng i ng n nh t), v i khoá là nh n ng i ng n nh t t s n i là $d[i]$.



Nút g c chính là nh t do có nhãn $d[i]$ nh nh t. M i l n l y nút g c ra c nh nhãn c a nó và s a nhãn cho các nh t do khác thì ph i th c hi n hai lo i x lí Heap ã nêu (phép c p nh t và phép lo i b g c).

V y thu t toán Dijkstra t ch c trên Heap nh sau:

C p nh t nút l c a Heap (t ng ng v i nút s có giá tr khoá b ng 0)

Vòng l p cho n khi Heap r ng (không còn nút nào)

Begin

+ L y nh u t i nút g c c a Heap (phép lo i b g c Heap)

+ N u $u = t$ thì thoát kh i vòng l p

+ ánh d u u là nh ã c c nh nhãn

+ Duy t danh sách cung k tìm các cung có nh u b ng u, nh cu i là v

N u v là nh t do và $d[v] > d[u] + \text{kho ng cách}(u, v)$ thì

Begin

S a nhãn cho v và ghi nh n nh tr c v là u

Trên Heap, c p nh t l i nút t ng ng v i nh v.

End;

End;

* **ánh giá**

+ Thu t toán Dijkstra t ch c nh nêu m c 1. Có ph c t p thu t toán là $O(N^2)$, nên không th th c hi n trên th có nhi u nh.

+ Các phép x lí Heap ã nêu (c p nh t Heap và lo i b g c Heap) c n th c hi n không quá $2 \cdot \lg M$ phép so sánh (n u Heap có M nút). S M t i a là N (s nh c a th) và ngày càng nh d n (t i 0). Ngoài ra, n u th th a (s cung ít) thì thao tác tìm nh v k v i nh u là không áng k khi ta t ch c danh sách các cung k này theo t ng o n có nh u gi ng nhau (d ng Forward Star). Do ó trên th th a, ph c t p c a Dijkstra_Heap có th t t i $O(N \cdot k \cdot \lg N)$ trong ó k không áng k so v i N

+ *K t lu n*: Trên th nhi u nh ít cung thì Dijkstra_Heap là th c hi n c trong th i gian có th ch p nh n.

III, ng i ng n nh t gi a t t c các c p nh - Thu t toán Floyd

Ta có th gi i bài toán tìm ng i ng n nh t gi a t t c các c p nh c a th b ng cách s d ng n l n gi i thu t ã Ford – Bellman ho c Dijkstra , trong



ó ta s ch n s l n l t là các nh c a th . Khi ó ta s thu c gi i thu t v i ph c t p là $O(n^4)$ (n u s d ng gi i thu t Ford - Bellman) ho c $O(n^3)$ i v i tr ng h p th có tr ng s không âm ho c không có chu trình.

Trong tr ng h p t ng quát vì c s d ng gi i thu t Ford-Bellman n l n không ph i là cách làm t t nh t. ây ta xét gi i thu t Floyd gi i bài toán trên v i ph c t p tính toán $O(n^3)$.

u vào là th cho b i ma tr n tr ng s $a[i, j], i, j = 1, 2, \dots, n$.

u ra : - Ma tr n ng i ng n nh t gi a các c p nh: $d[i, j] (i, j = 1, 2, \dots, n)$.

- Ma tr n ghi nh n ng i $tr[i, j] (i, j = 1, 2, \dots, n)$ trong ó $tr[i, j]$ ghi nh n nh i tr c nh j trong ng i ng n nh t t i n j .

Procedure Floyd;

Var i, j, k : integer;

Begin

{ Kh i t o }

for $i := 1$ to n do

for $j := 1$ to n do

begin

$d[i, j] := a[i, j];$

$tr[i, j] := i;$

end;

{ B c l p }

for $k := 1$ to n do

for $i := 1$ to n do

for $j := 1$ to n do

if $d[i, j] > d[i, k] + d[k, j]$ then

begin

$d[i, j] := d[i, k] + d[k, j];$

$tr[i, j] := tr[k, j];$

end;

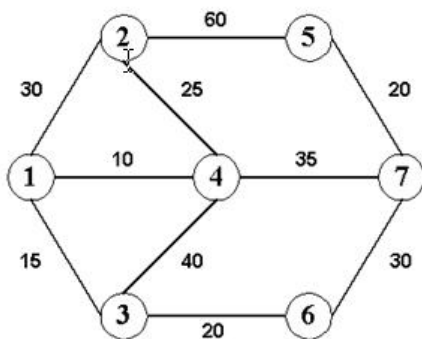
End;

6, M t s bài toán tìm ng i ng n nh t

Bài toán 1: H ng d n viên du l ch



Ông G. là m t h ng d n viên du l ch. Công vi c c a ông ta là h ng d n m t vài “tua” du l ch t thành ph này n thành ph khác. Trên các thành ph này, có m t vài con ng hai chi u c n i gi a chúng. M i c p thành ph có ng k t n i u có d ch v xe buýt ch ch y gi a hai thành ph này và ch y theo ng n i tr c ti p gi a chúng. M i d ch v xe buýt u có m t gi i h n l n nh t l ng khách mà xe buýt có th tr c. Ông G có m t t m b n ch các thành ph và nh ng con ng n i gi a chúng. Ngoài ra, ông ta c ng có thông tin v m i d ch v xe buýt gi a các thành ph . Ông hi u r ng ông không th a t t c các khách du l ch n thành ph th m quan trọng cùng m t chuy n i. L y ví d : V b n g m 7 thành ph , m i c nh c n i gi a các thành ph bi u th nh ng con ng và các s vi t trên m i c nh cho bi t cho bi t gi i h n hành khách c a d ch v xe buýt ch y trên tuy n ng ó.



Bây gi , n u ông G mu n a 99 khách du l ch t thành ph 1 n thành ph 7. Ông ta s ph i yêu c u ít nh t là 5 chuy n i, và l trình ông ta nên i là 1 – 2 – 4 – 7.

Nh ng, Ông G. nh n th y là th t khó tìm ra t t c l trình t t nh t sao cho ông ta có th a t t c khách du l ch n thành ph th m quan v i s chuy n i là nh nh t. Do v y mà

ông ta c n s tr giúp c a các b n.

D li u: Vào t file **Tourist.inp**

- T p **Tourist.inp** s ch a m t hay nhi u tr ng h p test.
- Dòng u tiên trong m i tr ng h p test ch a hai s nguyên N ($N \leq 100$) và R mô t l n l t s thành ph và s ng i gi a các thành ph .
- R dòng ti p theo, m i dòng ch a 3 s nguyên: $C1, C2, P$. $C1, C2$ mô t l trình ng i t thành ph $C1$ n thành ph $C2$ và P ($P > 1$) là gi i h n l n nh t có th ph c v c a d ch v xe buýt gi a hai thành ph .

Các thành ph c ánh d u b ng m t s nguyên t l n N . Dòng th $(R+1)$ ch a ba s nguyên S, D, T mô t l n l t thành ph kh i hành, thành ph c n n và s khách du l ch c ph c v .

K t qu : a ra file **Tourist.out**

Ghi ra s l trình nh nh t c n ph i i qua các thành ph th a m n yêu c u bài.



Ví dụ : Tourist.inp

```
7 10
1 2 30
1 3 15
1 4 10
2 4 25
2 5 60
3 4 40
3 6 20
4 7 35
5 7 20
6 7 30
1 7 99
0 0
```

Tourist.out

```
5
```

L i g i :

Đây là một bài toán hay, đòi hỏi các bạn phải nắm vững thuật toán Dijkstra. Bài toán này là bài toán biến thể của bài toán kinh điển tìm đường ngắn nhất. Với con đường (u,v) gọi $C[u,v]$ là số ngắn nhất mà có thể đi trên con đường đó trong một lần. $C[u,v]-1$ số là số khách tối đa có thể đi trên con đường đó trong một lần. $C[u,v] = 0$ thì con đường u và v không có con đường nào. Gọi $D[i]$ là số khách nhỏ nhất có thể đi từ i đến T thì mục tiêu là tìm $D[T]$. Với mỗi i khác j và i , ta có $D[j] = \min(D[i], C[i,j])$. Số khách có thể đi cùng một lúc thì mục tiêu là tìm $D[T]$. Một chú ý nữa là khi tính số lần đi, các bạn cần dùng các phép div , mod tính.

Chương trình thể hiện thuật toán trên (phức tạp: n^2)

$\{\mathbb{R}^+, \mathbb{Q}^+\}$

```
const
  INP = 'tourist.inp';
  OUT = 'tourist.out';
  maxn = 100;
var
  fi, fo: text;
```



```
c: array [1..maxn, 1..maxn] of longint;  
d: array [1..maxn] of longint;  
chua: array [1..maxn] of boolean;  
n, m, s, t, w: longint;  
procedure open_file;  
begin  
    assign(fi, INP); reset(fi);  
    assign(fo, OUT); rewrite(fo);  
end;  
procedure close_file;  
begin  
    close(fi);  
    close(fo);  
end;  
procedure read_data;  
var i, u, v, x: longint;  
begin  
    readln(fi, n, m);  
    for i := 1 to m do  
        begin  
            readln(fi, u, v, x);  
            c[u, v] := x - 1;  
            c[v, u] := x - 1;  
        end;  
    readln(fi, s, t, w);  
end;  
function min2(x, y: longint): longint;  
begin  
    if x > y then min2 := y else min2 := x;  
end;  
procedure process;  
var  
    i, max, last: longint;  
begin  
    fillchar(chua, sizeof(chua), true);
```



```
fillchar(d, sizeof(d), 0);
chua[s] := false;
last := s;
d[s] := maxlongint;
Kh i t o d[s] = vô cùng hay t t c m i n g i u có th n S cùng lúc
while chua[t] do
begin
for i := 1 to n do
{Tìm các nh i k v i last c p nh t l i}
if chua[i] and (d[i] < min2(c[last, i], d[last])) then
d[i] := min2(c[last, i], d[last]);
max := -1;
for i := 1 to n do
if chua[i] and (d[i] > max) then
begin
max := d[i];
last := i;
end;
chua[last] := false;
end;
end;
procedure write_result;
begin
if w mod d[t] = 0 then
writeln(fo, w div d[t])
else
writeln(fo, w div d[t] + 1);
end;
begin
open_file;
read_data;
process;
write_result;
close_file;
end.
```



Bài 2: Chuyền Hàng

B n m t kho hàng hình ch nh t kích th c $m \times n$ c chia thành các ô vuông n v (m hàng, n c t: các hàng ánh s t trên xu ng d i, các c t ánh s t trái qua ph i). Trên các ô vuông c a b n có m t s ký hi u:

- Các ký hi u # ánh d u các ô ã có m t ki n hàng x p s n.
- M t ký hi u *: ánh d u ô ang có m t rô b t.
- M t ký hi u \$: ánh d u ô ch a ki n hàng c n x p.
- M t ký hi u @: ánh d u v trí mà c n ph i x p ki n hàng vào \$ vào ô ó.
- Các ký hi u d u ch m ".": Cho bi t ô ó tr ng.

T i m t th i i m, rô b t có th th c hi n m t trong s 6 ng tác ký hi u là:

- L, R, U, D: T ng ng v i phép di chuy n c a rô b t trên b n : sang trái, sang ph i, lên trên, xu ng d i. Th c hi n m t phép di chuy n m t l công
- +, - : Ch th c hi n khi rô b t ng ô bên c nh ki n hàng \$. Khi th c hi n thao tác +, rô b t ng yên và y ki n hàng \$ làm ki n hàng này tr t theo h ng y, n khi ch m m t ki n hàng khác ho c t ng nhà kho thì d ng l i. Khi th c hi n thao tác -, rô b t kéo ki n hàng \$ v phía mình và lùi l i l ô theo h ng kéo. Th c hi n thao tác y ho c kéo m t C công. Rô b t ch c di chuy n vào ô không ch a ki n hàng c a kho.

Hãy tìm cách h ng d n rô b t th c hi n các thao tác a ki n hàng \$ v v trí @ sao cho s công ph i dùng là ít nh t.

D li u: Vào t file v n b n CARGO.INP

- Dòng 1: Ghi ba s nguyên d ng m, n, C (m, n 100; C 100)
- m dòng ti p theo, dòng th i ghi n ký ki u trên hàng i c a b n theo úng th t trái qua ph i.

Các ký hi u c ghi li n nhau.

K t qu : Ghi ra file v n b n CARGO.OUT

- Dòng 1: Ghi s công c n th c hi n
- Dòng 2: M t dãy liên ti p các ký t thu c {L, R, U, D, +, -} th hi n các ng tác c n th c hi n c a rô b t.

R ng bu c: Luôn có ph ng án th c hi n yêu c u bài.

Ví d :



CARGO.INP	CARGO.OUT
6 8 3	23
###.###	+RRRR-UR+DDDRD+
*\$.###	
####.###	
####.##	
#@.##	
#####	

Phân tích:

Thuật toán: Ta sử dụng thuật toán Dijkstra giải bài toán này.

*** Mô hình bài toán :**

Mô hình của bài toán này gồm 3 trường phân biệt với các nhau khác:

- i : Tên của điểm i ($i = 1..m$)
- j : Tên của điểm j ($j = 1..n$)
- h : Hướng của robot trong các hướng số với hướng ($h = 1..4$: Bắc, Đông, Nam, Tây).

	1	
2	\$	3
	4	

Bài có thể quan niệm mô hình là (i,j,u,v) : trong đó i,j : tên của điểm i ; u,v : tên của robot trong các hướng. Nhiệm vụ là tìm đường đi ngắn nhất và không chệch hướng để đi từ (i,j,u,v) đến (i,j,u,v) . Ta chỉ cần biết hướng của robot so với hướng là có thể tính được hướng của robot bằng cách dùng 2 hướng liên tiếp các số:

dx : array[1..4] of integer = (-1,0,1,0)

dy : array[1..4] of integer = (0,1,0,-1)

Khi đó, tên của robot sẽ là: $u := i + dx[h]$; $v := j + dy[h]$;

- Hai điểm $(i1,j1,h1)$ và $(i2,j2,h2)$ gọi là kề nhau nếu qua 1 trong 2 thao tác + hoặc - hướng của robot yêu cầu kéo từ ô $(i1,j1)$ đến ô $(i2,j2)$ và robot có thể di chuyển từ ô $(u1,v1)$ đến ô $(u2,v2)$ ($u1 = i1 + dx[h1]$; $v1 = j1 + dy[h1]$; $u2 = i2 + dx[h2]$; $v2 = j2 + dy[h2]$). Tất nhiên các ô $(i2,j2)$ và $(u2,v2)$ phải là ô không chệch hướng.

- Trường số gọi là 2 điểm là C (số công mà robot yêu cầu hướng từ ô $(i1,j1)$ đến ô $(i2,j2)$) bằng với công của robot di chuyển từ ô $(u1,v1)$ đến ô $(u2,v2)$.

Giải bài toán bằng cách tìm đường đi ngắn nhất của robot trong các hướng



hàng là hs và ô c n x p ki n hàng vào là ô (ie, je). Khi ó, ta s dùng thu t toán Dijkstra tìm ng i ng n nh t t nh (is, js, hs) n nh (ie, je, he) v i he thu c $\{1..4\}$.

M ng d s là 1 m ng 3 chi u: $d[i, j, h]$: dài ng i ng n nh t t nh xu t phát (is, js, hs) n nh (i, j, h). K t qu c a bài toán s là $d[ie, je, he]$ v i he thu c $\{1..4\}$.

ghi nh n ph ng án ta s dùng 3 m ng 3 chi u $tr1, tr2, tr3$. Khi ta đi t nh ($i1, j1, h1$) n nh ($i2, j2, h2$) thì ta s gán: $tr1[i2, j2, h2] := i1$; $tr2[i2, j2, h2] := j1$; $tr3[i2, j2, h2] := h1$ ghi nh n các thông tin: t a dòng, c t, hu ng c a d nh tr c nh ($i2, j2, h2$). T 3 m ng này ta có th d dàng l n l i ng i.

Bài 3: Ông Ngâu bà Ngâu

H n các b n ã bi t ngày "ông Ngâu bà Ngâu" hàng n m, ó là m t ngày y m a và n c m t. Tuy nhiên, m t ngày tr c ó, nhà Tr i cho phép 2 "ông bà" c oàn t . Trong v tr vùng thiên hà n i ông Ngâu bà Ngâu ng tr có N hành tinh ánh s t 1 n N, ông hành tinh Adam (có s hi u là S) và bà hành tinh Eva (có s hi u là T). H c n tìm n g p nhau.

N hành tinh c n i v i nhau b i m t h th ng c u v ng. Hai hành tinh b t k ch có th không có ho c duy nh t m t c u v ng (hai chi u) n i gi a chúng. H luôn i t i m c tiêu theo con ng ng n nh t. H i v i t c không i và nhanh h n t c ánh sáng. i m g p m t c a h ch có th là t i m t hành tinh th 3 nào ó.

Yêu c u: Hãy tìm m t hành tinh sao cho ông Ngâu và bà Ngâu cùng n ó m t lúc và th i gian n là s m nh t. Bi t r ng, hai ng i có th cùng i qua m t hành tinh n u nh h n hành tinh ó vào nh ng th i i m khác nhau.

D li u Trong file v n b n ONBANGAU.INP g m

Dòng u là 4 s N M S T (N 100, 1 S T N), M là s c u v ng. M dòng ti p, m i dòng g m hai s I J L th hi n có c u v ng n i gi a hai hành tinh I, J và c u v ng ó có dài là L ($1 \leq I \leq J \leq N, 0 < L \leq 200$).

K t qu Ra file v n b n ONBANGAU.OUT, do tính ch t c u v ng, m i n m m t khác, nên n u nh không t n t i hành tinh nào tho mãn yêu c u thì ghi ra m t dòng ch CRY. N u có nhi u hành tinh tho mãn thì ghi ra hành tinh có ch s nh nh t.



Ví d :

ONBANGAU.INP	ONBANGAU.OUT
4 4 1 4 1 2 1 2 4 1 1 3 2 3 4 2	2

T t ng thu t toán:

Chúng ta có m t s nh n xét sau:

+ Hai hành tinh b t kì ch c n i n nhau b i nhi u nh t m t c u v ng

+ Ông Ngâu và bà Ngâu luôn i t i m c tiêu theo con ng ng n nh t

+ H i v i v n t c không i và nhanh h n v n t c ánh sáng

Th c ch t ây là m t bài toán th : T hành tinh S(n i ông Ngâu) ta xây d ng b ng SP. Trong ó SP[i] là ng i ng n nh t t hành tinh S n hành tinh i (do ông Ngâu luôn i t i m c tiêu theo con ng ng n nh t). SP[i] = 0 t c là không có ng i t hành tinh S n hành tinh i. T ng t ta s xây d ng b ng TP, trong ó TP[i] là ng i ng n nh t t hành tinh T n hành tinh i. Và TP[i] = 0 t c là không có ng i t hành tinh T n hành tinh i.

Do yêu c u c a bài toán là tìm hành tinh khác S và T mà 2 ông bà Ngâu cùng n m t lúc và trong th i gian nhanh nh t. T c là ta s tìm hành tinh h sao cho (h khác S và T) và (SP[h] = ST[h]) t giá tr nh nh t khác 0. N u không có hành tinh h nào tho mãn thì ta thông báo CRY

xây d ng m ng SP và ST ta có r t nhi u gi i thu t khác nhau. ây ta ch n gi i thu t Dijkstra tìm ng i ng n nh t gi a 2 nh th .

Ch ng trình nh sau:

```
uses crt;
const MaxN = 101;
fi= 'ONBANGAU.inp';
fo= 'ONBANGAU.out';
var n,m,s,t,h:byte;
a:array[0..MaxN,0..MaxN] of byte;
SP,ST,B:array[0..MaxN] of integer;
f:text;
{ *-----*thnt*-----* }
procedure Init;
var i,u,v,ts:byte;
begin
```



```
fillchar(a,sizeof(a),0);
assign(f,fi);
reset(f);
readln(f,n,m,s,t);
for i:=1 to m do
begin
readln(f,u,v,ts);
a[u,v]:=ts;
a[v,u]:=ts;
end;
close(f);
end;
{ *-----*thnt*-----* }
procedure Build(s:byte);
var tt:array[0..maxN] of byte;
min,i,vtr:integer;
begin
fillchar(tt,sizeof(tt),0);
fillchar(b,sizeof(b),0);
for i:=1 to n do
b[i] := a[s,i];
tt[s]:=1;
min:=0;
while min <> maxint do
begin
min:=maxint; vtr:=0;
for i:=1 to n do
if tt[i] = 0 then
if (b[i] <>0) and (b[i]
begin min:=b[i]; vtr:=i; end;
if vtr <> 0 then tt[vtr]:=1;

for i:=1 to n do
if (tt[i] = 0) then
if a[vtr,i] <> 0 then
```



```
if (b[vtr] + a[vtr,i]
b[i]:=b[vtr] + a[vtr,i];
end;
end;
{ *-----*thnt*-----* }
procedure FindWay;
var i:integer;
begin
build(s); {xay dung mang SP }
SP:=B;
build(t); {xay dung mang ST}
ST:=B;
h:= 0;{hanh tinh can tim}
sp[0]:= Maxint;
for i:=1 to n do
if (SP[i] = ST[i]) then
if (SP[i]<>0) then
if (SP[i] < SP[h]) then
h:=i;
end;
{ *-----*thnt*-----* }
procedure ShowWay;
begin
assign(f,fo);
rewrite(f);
if h <> 0 then writeln(f,h)
else writeln(f,'CRY');
close(f);
end;
{ *-----*thnt*-----* }
begin
Init;
FindWay;
ShowWay;
end.
```



Bài 4: Máy ch

M t Công ty mu n phát tri n m t h th ng m ng máy tính l n bao g m các máy ch cung c p nhi u lo i hình d ch v khác nhau.

M ng d c k t n i t n máy ch b các kênh n i cho phép truy n tin hai chi u. Hai máy ch có th c n i tr c ti p v i nhau b i không quá m t kênh n i. M i máy ch c n i tr c ti p v i không quá 10 máy ch khác và hai máy ch b t k luôn có th k t n i c v i nhau ho c thông qua m t kênh n i tr c ti p gi a chúng ho c thông qua các máy ch khác. i v i kênh n i ta bi t th i gian truy n thông c o b ng mili gi y là m t s d ng. Kho ng cách (tính b ng mili gi y) $d(u,v)$ gi a máy ch u và máy ch v c xác nh nh là dài c a ng i ng n nh t (ng v i th i gian truy n thông trên c nh) n i u và v trên m ng. t n dùng, ta qui c $d(v,v)=0$ v i m i v.

Có m t s máy ch cung c p nhi u d ch v h n các máy ch khác. Vì th m i máy ch v c gán v i m t s t nhiên $r(v)$ g i là h ng c a nó. Máy ch có h ng càng cao càng là máy ch m nh h n. T i m i máy ch d li u v các máy ch g n nó th ng c c t gi . Tuy nhiên không ph i máy ch nào c ng là áng quan tâm. D li u v các máy ch lân c n v i h ng th p h n không c c t gi . Chính xác h n, *máy ch w là đáng quan tâm i v i máy ch v n u v i m i máy ch u sao cho $d(v,u) \leq d(v,w)$ ta có $r(u) \leq r(w)$* . Ch ng h n, t t các các máy ch v i h ng l n nh t u là áng quan tâm i v i t t c các máy ch . N u máy ch v có h ng l n nh t thì rõ ràng ch có các máy ch v i h ng l n nh t m i là áng quan tâm i v i v. G i $B(v)$ là t p các máy ch áng quan tâm i v i máy ch v. Ta g i *kích th c d li u v các máy ch áng quan tâm i v i máy ch v* là $|B(v)|$.

Yêu c u: Tính t ng kích th c d li u v các máy ch áng quan tâm c a t t c các máy ch trong toàn m ng. B i t r ng t ng này có giá tr không v t quá $30n$.

D li u: Vào t file v n b n SERVER.INP:

- + Dòng u ch a hai s n, m trong ó n là s máy ch ($1 \leq n \leq 3000$) và m là s kênh n i ($1 \leq m \leq 5n$)
- + Dòng th i trong s n dòng ti p theo ch a r_i ($1 \leq r_i \leq 10$) là h ng c a máy ch i.
- + Ti p n là m dòng, m i dòng ch a thông tin v m t kênh n i bao g m a, b, t ($1 \leq t \leq 1000$, $1 \leq a, b \leq n$, $a \neq b$), trong ó a, b là ch s c a hai máy ch



c n i b i kênh ang xét còn t là th i gian truy n thông c a kênh (o b ng mili giây).

K t qu : Ghi ra file v n b n SERVER.OUT m t s nguyên duy nh t là t ng kích th c d li u v các máy ch áng quan tâm c a t t c các máy ch trong toàn m ng

Ví d :

SERVER.INP	SERVER.OUT	Gi i thích
4 3 2 3 1 1 1 4 30 2 3 20 3 4 20	9	Ta có: B(1)={1,2} B(2)={2} B(3)={2,3} B(4)={1,2,3,4}

Bài này ta dùng thu t toán Dijkstra gi i ch ng trình nh sau

```

const
    tfi      =    'SERVER.INP';
    tfo      =    'SERVER.OUT';
    maxN     =    3000;
type
    mang1    =    array[1..10] of integer;
    mang2    =    array[1..maxN] of integer;
var
    fi,fo    :    text;
    N,M      :    longint;
    Sol      :    array[1..maxN] of byte;
    r        :    array[1..maxN] of byte;
    a        :    array[1..maxN] of ^mang1;
    d        :    array[1..maxN] of ^mang1;
    kq       :    longint;
    Q        :    array[1..maxN] of longint;
    vt       :    ^mang2;
    qn       :    longint;
    kc       :    array[1..maxN] of longint;
    
```



loai : array[1..maxN] of byte;
Rank : array[1..maxN] of byte;
Q1 : ^mang2;
q1n : integer;
kcold : longint;
t1 : longint;
t2 : longint absolute 0:\$46c;
rmax : byte;

```
procedure CapPhat;  
var i: longint;  
begin  
  for i:=1 to maxN do new(a[i]);  
  for i:=1 to maxN do new(d[i]);  
  new(q1);  
  new(vt);  
end;  
procedure InitQ;  
begin  
  qn:=0;  
end;  
procedure Put(u: longint);  
begin  
  inc(qn);  
  q[qn]:=u;  
end;  
function Get: longint;  
var i,u: longint;  
begin  
  u:=1;  
  for i:=2 to qn do  
    if kc[q[i]]<kc[q[u]] then u:=i;  
  Get:=q[u];  
  q[u]:=q[qn];  
  dec(qn);
```




```
end;  
function Qempty: boolean;  
begin  
    Qempty:=(qn=0);  
end;  
procedure Docdl;  
var i,u,v,w:longint;  
begin  
    assign(fi,tfi); reset(fi);  
    readln(fi,n,m);  
    for i:=1 to N do sol[i]:=0;  
    for i:=1 to N do readln(fi,r[i]);  
    for i:=1 to M do  
        begin  
            readln(fi,u,v,w);  
            inc(sol[u]); a[u]^[sol[u]]:=v; d[u]^[sol[u]]:=w;  
            inc(sol[v]); a[v]^[sol[v]]:=u; d[v]^[sol[v]]:=w;  
        end;  
    close(fi);  
    rmax:=0;  
    for i:=1 to N do  
        if rmax<r[i] then rmax:=r[i];  
    end;  
  
    procedure Dijkstra(xp: longint);  
    var i,u,v,ll: longint;  
        MaxRank: byte;  
    begin  
        InitQ;  
        kcold:=-1;  
        for i:=1 to N do loai[i]:=0;  
        for i:=1 to N do rank[i]:=rmax;  
        MaxRank:=0;  
        Put(xp); loai[xp]:=1; kc[xp]:=0;  
        repeat
```



```
u:=Get; loai[u]:=2;
if MaxRank<r[u] then MaxRank:=r[u];
if kc[u]=kcold then
  begin
    inc(q1n);
    q1^[q1n]:=u;
  end
else
  begin
    q1n:=1;
    q1^[1]:=u;
  end;
kcold:=kc[u];
for i:=1 to q1n do
  Rank[q1^[i]]:=MaxRank;
if (maxRank<rmax) then
for i:=1 to sol[u] do
  begin
    v:=a[u]^i; ll:=d[u]^i;
    if (loai[v]=1) and (kc[v]>kc[u]+ll) then kc[v]:=kc[u]+ll;
    if loai[v]=0 then
      begin
        Loai[v]:=1;
        kc[v]:=kc[u]+ll;
        Put(v);
      end;
    end;
  until Qempty;
end;
function Dem: longint;
var k,i: longint;
begin
  k:=0;
  for i:=1 to N do
    if (Rank[i]<=r[i]) then k:=k+1;
```



```

Dem:=k;
end;
procedure Solve;
var i,k: longint;
begin
  kq:=0;
  for i:=1 to N do
    begin
      Dijkstra(i);
      kq:=kq+Dem;
    end;
  end;
procedure Inkq;
begin
  assign(fo,tfo); rewrite(fo);
  writeln(fo,kq);
  close(fo);
end;
BEGIN
  clrscr;
  t1:=t2;
  CapPhat;
  Docdl;
  Solve;
  Inkq;
  writeln('Total time =',(t2-t1)/18.3:0:4,' s');
  readln;
END.

```

Bài 5: Hành trình trên xe l a

L ch ho t ng c a tuy n ng s t trong m t ngày bao g m thông tin c a t ng chuy n t u có trong ngày ó. Thông tin c a m i chuy n t u bao g m:

- S hi u chuy n t u (c ánh s t l n M),
- Danh sách các ga mà chuy n t u ó d ng l i, m i ga bao g m:
 - + S hi u ga (các ga c ánh s t l tr i),
 - + Gi n (s th c),



+ Gi i(s th c).

Các giá tr th i gian tính theo n v gi và vi t d i d ng th p phân (ví d 7.5 có ngh a là 7 gi 30 phút).

M t hành khách b t k khi i n m t ga nào ó (g i là ga hi n t i) cho bi t yêu c u c a mình g m: th i i m mà t ó anh ta có th i c, s hi u ga c n n và th i gian t i thi u cho m i l n chuy n t u. Nhân viên nhà ga ph i tr l i c là có áp ng c yêu c u c a khách không? N u áp ng c, nhân viên nhà ga ph i a ra c hành trình c n i cho khách.

Hãy gi i bài toán trong 2 tr ng h p:

- Tìm hành trình n ga cu i cùng s m nh t.
- Tìm hành trình ít ph i chuy n t u nh t. N u t n t i nhi u ph ng án nh v y, hãy tìm ph ng án n ga cu i cùng s m nh t.

D li u: File vào g m các dòng:

- Dòng 1: Ghi 4 s theo th t : th i i m i, ga hi n t i, ga c n n và th i gian t i a cho m i l n chuy n t u;
- Dòng 2: Ghi s nguyên d ng M ($M \leq 50$);
- Dòng $i+2$ ($i = 1, 2, \dots, M$): Ghi thông tin c a chuy n t u s hi u i bao g m: s l n ga mà chuy n t u ó d ng l i (≤ 20), danh sách các ga theo trình t i n c a chuy n t u, trong ó m i ga c mô t b i 3 s theo th t : s hi u ga, gi n, gi i.

Các s trên cùng m t dòng ghi cách nhau b i m t d u tr ng.

K t qu : Trong tr ng h p không tìm th y hành trình thì ghi giá tr 0. Trái l i, ghi hành trình tìm c đ i d ng sau:

- Dòng u ghi S là s hi u chuy n t u mà khách b t u i,
- Dòng ti p ghi T_1 là th i i m i c a chuy n t u này,
- Dòng ti p ghi K là s l n khách ph i chuy n t u,
- K dòng ti p, m i dòng ghi thông tin c a m t l n chuy n t u g m s hi u ga mà khách ph i chuy n t u và s hi u chuy n t u c n i ti p (ghi cách nhau m t d u tr ng),
- Dòng cu i ghi T_2 là th i i m n ga cu i cùng c a hành trình.

K t qu c a câu a và câu b ghi cách nhau b i l dòng tr ng.



Ví d :

XELUA.INP	XELUA.OUT
6 4 3 1.5	26.02
6	2 3
3 1 7 7 2 8 9.1 3 9.5 9.5	5 4
2 4 6 6 2 7 7.5	10.0
2 2 7.5 7.5 5 8 8	
3 6 8 8 5 9 9.5 3 10 10	5
3 4 6.5 6.5 7 9 9.5 3 11 11	6.5
2 4 7 7 3 12 12	0
	11.0

Ch ng trình nh sau

Const

FI = 'xelua.inp';

FO = 'xelua.out';

MAX_VALUE = 999999999;

Var

n, nU, ga_di, ga_den, dem : integer;

t0, t_di, t_cho : real;

tau, ga, tr, U : array[0..1001] of integer;

d, gio_den, gio_di : array[0..1001] of real;

f : text;

Procedure Doc;

Var m, i, j, k : integer;

Begin

assign(f, FI); reset(f);

read(f, t_di, ga_di, ga_den, t_cho, m);

tau[0] := 0;

ga[0] := ga_di;

gio_den[0] := t_di;

gio_di[0] := t_di;

n := 0;

for i := 1 to m do

begin



```
read(f, k);
for j := 1 to k do
begin
n := n + 1;
tau[n] := i;
read(f, ga[n], gio_den[n], gio_di[n]);
end;
end;
close(f);
End;
Function Khoang_cach(i, j : integer) : real;
Var t : real;
Begin
if tau[i] = tau[j] then
begin
t := gio_di[i] - gio_den[i];
if (j = i+1) and (t <= t_cho) then Khoang_cach := gio_den[j] - gio_den[i]
else Khoang_cach := MAX_VALUE;
end
else
if ga[i] = ga[j] then
begin
t := gio_di[j] - gio_den[i];
if (t >= 0) and (t <= t_cho) then Khoang_cach := t + t0
else Khoang_cach := MAX_VALUE;
end
else Khoang_cach := MAX_VALUE;
End;
Procedure Khoi_tao;
Var i : integer;
Begin
for i := 0 to n do
begin
d[i] := Khoang_cach(0, i);
tr[i] := 0;
```



```
end;
nU := n;
for i := 1 to nU do U[i] := i;
End;
Function Co_dinh_nhan : integer;
Var i, j : integer;
Begin
  i := 1;
  for j := 2 to nU do
    if d[U[j]] < d[U[i]] then i := j;
  Co_dinh_nhan := U[i];
  U[i] := U[nU];
  nU := nU - 1;
End;
Procedure Sua_nhan(p : integer);
Var
  x, i : integer;
  kc : real;
Begin
  for i := 1 to nU do
    begin
      x := U[i];
      kc := Khoang_cach(p, x);
      if d[x] > d[p] + kc then
        begin
          d[x] := d[p] + kc;
          tr[x] := p;
        end;
    end;
  end;
End;
Procedure Print(i : integer);
Begin
  if tr[i] = 0 then
    begin
      writeln(f, tau[i]);
    end;
  end;
```



```
writeln(f, gio_di[i] : 0 : 1);
writeln(f, dem);
exit;
end;
if tau[tr[i]] <> tau[i] then dem := dem + 1;
Print(tr[i]);
if tau[tr[i]] <> tau[i] then writeln(f, ga[i], ' ', tau[i]);
End;
Procedure Ghi;
Var
  dich, i : integer;
  som_nhat : real;
Begin
  som_nhat := MAX_VALUE;
  for i := 1 to n do
    if (ga[i] = ga_den) and (d[i] < som_nhat) then
      begin
        som_nhat := d[i];
        dich := i;
      end;
  if som_nhat = MAX_VALUE then writeln(f, 0)
  else
    begin
      dem := 0;
      Print(dich);
      writeln(f, gio_den[dich] : 0 : 1);
    end;
  writeln(f);
End;
Procedure Dijkstra;
Var p : integer;
Begin
  Khoi_tao;
  while nU > 0 do
    begin
```




```

p := Co_dinh_nhan;
Sua_nhan(p);
end;
Ghi;
End;
Procedure Xu_ly;
Begin
  assign(f, fo); rewrite(f);
  { Cau a }
  t0 := 0;
  Dijkstra;
  { Cau b }
  t0 := 9999;
  Dijkstra;
  close(f);
End;
Begin
  Doc;
  Xu_ly;
End.

```

Bài 6: H i th o tr c tuy n

M t trung tâm qu n tr m t m ng g m $N (\leq 100)$ c ng truy c p c ánh s t 1 n N . Gi a hai c ng có th không có ng n i ho c có ng n i tr c ti p và thông tin truy n hai chi u trên ng n i. M ng có M ng n i tr c ti p gi a các c ng và n u ng n i tr c ti p gi a hai c ng i, j c s d ng thì chi phí truy n tin ph i tr là $c_{ij} (\leq 32767)$.

Trung tâm nh n c h p ng t ch c m t cu c h i th o tr c tuy n t 3 a i m khác nhau truy c p vào m ng t 3 c ng. B n hãy giúp công ty t ch c s d ng các ng n i truy n tin sao cho t ng chi phí là ít nh t có th c.

D li u: File vào g m các dòng:

- Dòng u tiên ghi hai s N và M ;
- M dòng ti p theo, m i dòng ch a 3 s nguyên d ng trong ó 2 s u là ch s c a hai c ng, s th 3 là chi phí khi truy n tin trên hai c ng ó;



- Dòng cuối cùng của 3 số nguyên đầu tiên theo thứ tự là chữ số của 3 chữ số tiếp theo của mã đề thi.

Kết quả : File ra gồm:

- Dòng đầu ghi chữ 'No' nếu không thể tìm được hai số tiếp theo, ngược lại ghi 'Yes'.
- Nếu tìm được cách tìm hai số thì dòng đầu ghi S là chỉ phí nhỏ nhất tìm được và dòng tiếp theo ghi P là số lượng đỉnh đi qua. P dòng tiếp theo là dòng ghi hai số i, j thể hiện một đường đi giữa hai đỉnh i và j có số lượng đỉnh nhỏ nhất. Các số trên một dòng ghi cách nhau bằng một dấu cách.

Ví dụ :

NET.INP	NET.OUT
8 12	Yes
1 2 20	27
2 3 8	4
2 4 3	1 2
2 5 3	2 4
2 6 6	2 5
3 5 2	5 6
3 6 9	
4 7 5	
5 6 1	
5 7 7	
6 8 4	
7 8 6	
1 4 6	

Lưu ý : Chúng ta thay thế các đỉnh bằng các số để tìm đường đi ngắn nhất. Đó là số có một chữ số thể hiện ba chữ số. Mà cây đó là cây có chiều dài nhỏ nhất. Vì vậy ta tìm kiếm là trung gian T (có thể trùng với 1 trong ba chữ số). Thứ tự truy cập T là 3 chữ số để tìm đường đi ngắn nhất. Đó là ta sử dụng thuật toán Floyd. Sau đó tìm xem nào có thể tìm được cách nhỏ nhất của ba lần tìm kiếm thì các đường đi đó chính là các đường đi tối ưu.



Ch ng trình

```
Program Hoi_thao_truc_tuyen;
Uses crt;
Const
  FI = 'net.inp';
  FO = 'net.out';
  MAX_N = 100;
  MAX_VALUE = 999999999;
Var
  n, x, y, z, sum, so_canh : integer;
  c : array[1..MAX_N, 1..MAX_N] of longint;
  tr : array[1..MAX_N, 1..MAX_N] of byte;
  f : text;
Procedure Doc;
Var m, chi_phi, i, j, k : integer;
Begin
  assign(f, FI); reset(f);

  readln(f, n, m);
  for i := 1 to n do
    for j := 1 to n do c[i, j] := MAX_VALUE;
  for k := 1 to m do
    begin
      readln(f, i, j, chi_phi);
      c[i, j] := chi_phi;
      c[j, i] := chi_phi;
    end;
  readln(f, x, y, z);
  close(f);
End;
Procedure Floyd;
Var i, j, k : integer;
Begin
  for i := 1 to n do
    for j := 1 to n do tr[i, j] := i;
```



```
for k := 1 to n do
  for i := 1 to n do
    for j := 1 to n do
      if  $c[i, j] > c[i, k] + c[k, j]$  then
        begin
           $c[i, j] := c[i, k] + c[k, j]$ ;
           $tr[i, j] := tr[k, j]$ ;
        end;
    end;
  end;
End;

Procedure Print(i, j : integer);
Begin
  if  $i = j$  then exit;
  if  $c[tr[i, j], j] < -1$  then
    begin
       $so\_canh := so\_canh + 1$ ;
       $sum := sum + c[tr[i, j], j]$ ;
       $c[tr[i, j], j] := -1$ ;
       $c[j, tr[i, j]] := -1$ ;
    end;
  Print(i,  $tr[i, j]$ );
End;

Procedure Ghi;
Var min, t, i, j : longint;
Begin
  assign(f, FO); rewrite(f);
  min := MAX_VALUE;
  for i := 1 to n do
    if  $min > c[x, i] + c[y, i] + c[z, i]$  then
      begin
        t := i;
         $min := c[x, i] + c[y, i] + c[z, i]$ ;
      end;
  if  $min = MAX\_VALUE$  then write(f, 'No')
  else
    begin
```



```

so_canh := 0;
sum := 0;
Print(x, t);
Print(y, t);
Print(z, t);
writeln(f, 'Yes');
writeln(f, sum);
writeln(f, so_canh);
for i := 1 to n do
    for j := i+1 to n do
        if c[i, j] = -1 then writeln(f, i, ' ', j);
    end;
close(f);
End;
Begin
    Doc;
    Floyd;
    Ghi;
End.

```

Bài 7: Ch trung tâm

Có N a i m dân c ánh s t l n N . Gi a M c p a i m trong s N a i m nói trên có tuy n ng n i chúng. C n xây d ng m t trung tâm d ch v t ng h p t i m t a i m trùng v i m t a i m dân c , sao cho t ng kho ng cách t trung tâm d ch v n N a i m dân c là nh nh t. Ta g i kho ng cách gi a hai a i m là dài ng i ng n nh t n i chúng. Gi s N a i m trên là liên thông v i nhau. N u có nhi u ph ng án thì a ra ph ng án t trung tâm d ch v t i a i m có s h i u nh nh t.

D li u: File vào g m $M+1$ dòng:

- Dòng 1: Ch a hai s nguyên d ng N và M ($N \leq 100$);
- Dòng $i+1$ ($1 \leq i \leq M$): Ch a 3 s nguyên d ng x, y, z , ó hai s u x, y là s h i u c a hai a i m dân c c n i v i nhau b i tuy n ng này, còn s th ba z (≤ 32767) là dài c a tuy n ng này.

K t qu : File ra g m 2 dòng:

- Dòng 1: Ghi v trí trung tâm d ch v ;
- Dòng 2: Ghi t ng kho ng cách t trung tâm d ch v n các a i m dân c .



Ví d :

MARKET.INP	MARKET.OUT
5 7	3
1 2 9	15
2 3 4	
1 4 2	
4 5 5	
5 3 1	
5 1 5	
3 1 4	

```
Program Cho_trung_tam;  
Uses crt;  
Const  
  FI = 'market.inp';  
  FO = 'market.out';  
  MAX_N = 100;  
  MAX_VALUE = 999999999;  
Var  
  n, dia_diem, min : longint;  
  d : array[1..MAX_N, 1..MAX_N] of longint;  
  f : text;  
Procedure Doc;  
Var i, j, k, m : integer;  
Begin  
  assign(f, FI); reset(f);  
  
  read(f, n, m);  
  for i := 1 to n do  
    begin  
      d[i, i] := 0;  
      for j := i+1 to n do  
        begin  
          d[i, j] := MAX_VALUE;  
          d[j, i] := MAX_VALUE;
```



```
    end;
  end;
for k := 1 to m do
  begin
    read(f, i, j);
    read(f, d[i, j]);
    d[j, i] := d[i, j];
  end;
close(f);
End;
Procedure Floyd;
Var sum, i, j, k : longint;
Begin
  for k := 1 to n do
    for i := 1 to n do
      for j := 1 to n do
        if d[i, j] > d[i, k] + d[k, j] then d[i, j] := d[i, k] + d[k, j];
      end;
    end;
  min := MAX_VALUE;
  for i := 1 to n do
    begin
      sum := 0;
      for j := 1 to n do sum := sum + d[i, j];
      if sum < min then
        begin
          dia_diem := i;
          min := sum;
        end;
    end;
  end;
End;
Procedure Ghi;
Begin
  assign(f, FO); rewrite(f);
  writeln(f, dia_diem);
  write(f, min);
```



close(f);
End;
Begin
Doc;
Floyd;
Ghi;
End.

Bài 8: Thành ph trên sao ho

u th k 21, ng i ta thành l p m t d án xây d ng m t thành ph trên sao Ho th k 22 con ng i có th s ng và sinh ho t ó. Gi s r ng trong th k 22, ph ng ti n giao thông ch y u s là các ph ng ti n giao thông công c ng nên i l i gi a hai i m b t k trong thành ph ng i ta có th yên tâm ch n ng i ng n nh t mà không s b tr gi do k t xe. Khi mô hình thành ph c chuy n lên Internet, có r t nhi u ý ki n phân nân v tính h p lý c a nó, c bi t, t t c các ý ki n u cho r ng h th ng ng ph nh v y là quá nhi u, làm t ng chi phí xây d ng c ng nh b o trì.

Hãy b i m t s ng trong d án xây d ng thành ph tho mãn:

+ *N u gi a hai a i m b t k trong d án ban u có ít nh t m t ng i thì s s a i này không làm nh h ng t i dài ng i ng n nh t gi a hai a i m ó.*

+ *T ng dài c a nh ng ng ph c gi l i là ng n nh t có th*

D li u: Vào t file v n b n CITY.INP, ch a b n d án

+ Dòng th nh t ghi s a i m N và s ng ph m (gi a hai a i m b t k có nhi u nh t là m t ng ph n i chúng, $n \leq 200$; $0 \leq m \leq n*(n-1)/2$)

+ m dòng ti p theo, m i dòng ghi ba s nguyên d ng u, v, c cho bi t có ng hai chi u n i gi a hai a i m u, v và dài c a con ng ó là c ($c \leq 10000$)

K t qu : Ghi ra file v n b n CITY.OUT, ch a k t qu sau khi s a i

+ Dòng th nh t ghi hai s k, d. Trong ó k là s ng ph còn l i còn d là t ng dài c a các con ng ph còn l i.

+ k dòng ti p theo, m i dòng ghi hai s nguyên d ng p, q cho bi t c n ph i gi l i con ng n i a i m p v i a i m q

Các s trên m t dòng c a các file CITY.INP, CITY.OUT c ghi cách nhau ít nh t m t d u cách



Ví d :

CITY.INP	CITY.OUT
10 12	9 21
1 2 1	1 2
1 5 2	1 5
2 6 7	3 4
3 4 1	3 7
3 7 2	5 6
4 8 8	6 7
5 6 3	6 9
6 7 1	7 8
6 9 2	9 10
7 8 5	
7 10 8	
9 10 4	

Ch ng trình

```
const
  tfi      = 'CITY.INP';
  tfo      = 'CITY.OUT';
  maxN     = 200;
  Unseen   = 2000000;
type
  mangB    = array[1..maxN] of byte;
  mangL    = array[1..maxN] of LongInt;
var
  fi,fo    : text;
  N,M      : LongInt;
  a        : array[1..maxN] of ^mangL;
  Gr       : array[1..maxN] of ^mangB;
  Tr       : array[1..maxN,1..maxN] of byte;
  S,D      : LongInt;

procedure CapPhat;
var i: integer;
```



```
begin
  for i:=1 to maxN do new(a[i]);
  for i:=1 to maxN do new(Gr[i]);
end;
procedure GiaiPhong;
var i: integer;
begin
  for i:=1 to maxN do Dispose(a[i]);
  for i:=1 to maxN do Dispose(Gr[i]);
end;
procedure Docdl;
var i,j,u,v,l: LongInt;
begin
  assign(fi,tfi); reset(fi);
  readln(fi,N,M);
  for i:=1 to N do
    for j:=1 to N do a[i]^j:=Unseen;
  for i:=1 to N do
    for j:=1 to N do Gr[i]^j:=0;
  for i:=1 to M do
    begin
      readln(fi,u,v,l);
      a[u]^v:=l; a[v]^u:=l;
      Gr[u]^v:=1; Gr[v]^u:=1;
    end;
  close(fi);
end;
procedure Floyd;
var k,i,j: integer;
begin
  Fillchar(Tr,sizeof(Tr),0);
  for k:=1 to N do
    for i:=1 to N do
      for j:=1 to N do
        if a[i]^j>=a[i]^k+a[k]^j then
```



```
begin
  a[i]^j:=a[i]^k+a[k]^j;
  Tr[i,j]:=k;
end;
end;
procedure Solve;
var i,j: LongInt;
begin
  for i:=1 to N do
    for j:=1 to N do
      if (Gr[i]^j=1) and (Tr[i,j]>0) then
        begin
          Gr[i]^j:=0;
          Gr[j]^i:=0;
        end;
      S:=0;
      D:=0;
      for i:=1 to N-1 do
        for j:=i+1 to N do
          if Gr[i]^j=1 then
            begin
              S:=S+a[i]^j;
              D:=D+1;
            end;
        end;
      end;
    procedure inkq;
    var i,j: LongInt;
    begin
      assign(fo,tfo); rewrite(fo);
      writeln(fo,d,' ',S);
      for i:=1 to N-1 do
        for j:=i+1 to N do
          if Gr[i]^j=1 then
            writeln(fo,i,' ',j);
      close(fo);
```



```
end;  
BEGIN  
  CapPhat;  
  Docdl;  
  Floyd;  
  Solve;  
  Inkq;  
  GiaiPhong;  
END.
```

B. KẾT LUẬN

Tìm kiếm giải pháp trên thực tế còn có nhiều thuật toán nữa và cũng còn nhiều cách cài đặt các thuật toán trên hệ thống. Tuy nhiên trong chuyên luận này tôi chỉ đưa ra các cách cài đặt cơ bản nhất để học sinh tự nghiên cứu và phát triển thêm. Vì thời gian và trình độ có hạn nên chuyên luận này có thể còn nhiều hạn chế, tôi rất mong các bạn đồng nghiệp và các em học sinh góp ý.

C. TÀI LIỆU THAM KHẢO

- 1, Tài liệu chuyên tin quyển 2 – H. S. àm
- 2, Giải thuật và lập trình – Lê Minh Hoàng
- 3, Một số tài liệu khác của các đồng nghiệp



Chuyên x p lo i B

NG D NG BFS VÀ DFS TRONG GI I BÀI T P LÝ THUY T TH

T Tìn h c tr ng THPT Chuyên L ng V n T y – Ninh Bình

1. Ph n m u

1.1 Lý do ch n tài.

- B c sang th k 21, nhìn l i th k 20 là th k mà con ng i t c nhi u thành t u khoa h c r c r nh t, m t trong nh ng thành t u ó là s bùng n c a ngành khoa h c máy tính. S phát tri n k đi u c a máy tính trong th k này g n li n v i s phát tri n toán h c hi n i, ó là toán r i r c. Toán r i r c nói chung và lý thuy t th nói riêng là công c thi t y u cho nhi u ngành khoa h c k thu t
- Trong ch ng trình h c t p h c sinh chuyên Tin tr ng THPT c trang b các ki n th c v lý thuy t th nh m ph c v cho vi c l p trình gi i toán, làm bài t p l p trình. B i i u c n b n thông qua gi i bài t p, h c sinh ph i th c hi n nh ng ho t ng nh t nh bao g m c nh n d ng và th hi n nh ngh a, nh lý, quy t c hay ph ng pháp, nh ng ho t ng toán h c ph c h p. H c sinh s n m c lý thuy t m t cách v ng vàng h n thông qua vi c làm bài t p.
- Vi c cung c p thêm m t ph ng pháp gi i bài t p cho h c sinh chuyên Tin là m t nhu c u c n thi t. Hi n nay vi c nghiên c u khai thác m t s y u t c a lý thuy t th c ng c m t s tác gi quan tâm. N u ta có các ph ng pháp giúp h c sinh chuyên Tin trung h c ph thông v n d ng ki n th c v lý thuy t th vào gi i toán thì s giúp h c sinh gi i quy t c m t s l p bài toán góp ph n nâng cao ch t l ng d y h c gi i bài t p cho h c sinh chuyên Tin.
- **BFS** và **DFS** là nh ng thu t toán tìm ki m c b n nh ng r t quan tr ng trên th . Nh ng thu t toán này s là n n móng quan tr ng có th xây d ng và thi t k nh ng thu t gi i khác trong lý thuy t th . Xu t phát t nh ng lý do trên tôi l a ch n tài: “ *ng d ng BFS và DFS trong gi i bài t p lý thuy t th* ”.

1.2. M c tiêu, nhi m v c a tài.

- M c tiêu c a tài: Ch ra h ng v n d ng DFS và BFS trong lý thuy t th vào gi i các bài toán và tìm ra các bi n pháp giúp h c sinh chuyên Tin



trung học phổ thông hình thành và phát triển năng lực và năng lực lý thuyết
th vào giải bài tập lập trình.

- Nhiệm vụ của tài:

- + Tìm hiểu những nội dung cơ bản của lý thuyết thuật c trang b cho học c sinh chuyên Tin. Trong ó đi sâu vào hai *thu t toán tìm kiếm trên th là DFS và BFS*
- + Chỉ ra những bài tập trong chương trình toán có thể v n d ng DFS và BFS giải các bài tập trong lý thuyết thuật
- + Kiểm tra hiểu qu c a các biện pháp, ph ng án lý thuyết thuật vào giải toán trong thực t .

1.3. Ph ng pháp nghiên cứu.

- Nghiên cứu lý luận

- + Tài liệu Giáo khoa chuyên tin, sách nâng cao, sách chuyên .
- + Các tài liệu về lý thuyết thuật và những ứng dụng c a nó trong thực tiễn cụ c s ng và trong d y h c.
- + Các công trình nghiên cứu các vấn liên quan trực tiếp n ph ng pháp thuật .

- Thực nghiệm sư phạm

- + Chỉ ra cho học sinh các dấu hiệu "nh n d ng" và cách thức v n d ng lý thuyết thuật vào giải bài tập toán.
- + Biên soạn những bài tập luyện tập cho học sinh và m t s bài kiểm tra đánh giá khả năng v n d ng lý thuyết thuật vào giải toán.
- + Tiến hành thực nghiệm và đánh giá kết qu thực nghiệm.



2. Ph n n i dung

2.1. C s lý lu n

Theo tri t h c duy v t bi n ch ng, mâu thu n là ng l c thúc y quá trình phát tri n. M t v n c g i ra cho h c sinh h c t p chính là m t mâu thu n gi a yêu c u nhi m v nh n th c v i tri th c và kinh nghi m s n có.

Theo các nhà tâm lý h c, con ng i ch b t u t duy tích c c khi n y sinh nhu c u t duy, t c là khi ng tr c m t khó kh n v nh n th c c n ph i kh c ph c, m t tình hu ng g i v n .

Theo tâm lý h c ki n t o, h c t p ch y u là m t quá trình trong ó ng i h c xây d ng tri th c cho mình b ng cách liên h nh ng c m nghi m m i v i nh ng tri th c ã có.

2.2.Th c tr ng

a. Thu n l i

- c s quan tâm, giúp t n tình c a Ban Giám Hi u và t ch c oàn th trong nhà tr ng. S ng h nhi t tình c a các ng nghi p ã giúp cho quá trình gi ng d y Tin h c c a tôi t hi u qu cao h n.
- H c sinh l p tr ng chuyên nói chung, h c sinh l p chuyên tin nói riêng thông minh, ham h c. Trong l p a s h c sinh tích c c phát bi u xây d ng bài, ó là ngu n ng viên l n trong quá trình gi ng d y c a tôi.
- Nhìn chung, h c t p theo ph ng pháp m i thì h c sinh có h ng thú h c t p h n so v i so v i ph ng pháp d y h c truy n th ng. Vì th , có i u ki n phát tri n t duy và kh n ng đi n t c a các em.

b. Khó kh n

- i ng giáo viên Tin h c còn thi u, c bi t là giáo viên d y chuyên Tin. Công vi c m i giáo viên d y tin h c trong nhà tr ng ph i m nh n r t nhi u, th i gian u t cho chuyên môn còn h n ch .
- D y h c hi n ang theo l i d y nh i nhét, d y luy n thi, i phó v i thi, ki m tra sao cho có i m s cao mà ch a quan tâm n s phát tri n trí tu , n ng l c cá nhân h c sinh. Giáo viên c ng nh h c sinh ch a kh c ph c c nh n th c, thói quen d y h c truy n th ng, n ng v lý thuy t coi nh th c hành ng d ng. Các em h c sinh th ng ch n m lý thuy t, vì c v n d ng lý thuy t làm các bài t p còn h n ch . Giáo viên ph i song hành v i c d y lý thuy t cho h c sinh cùng v i a ra ph ng pháp làm bài t p v n d ng các ki n th c ã h c. Vì c làm bài t p th c hành s giúp h c sinh n m v ng ki n

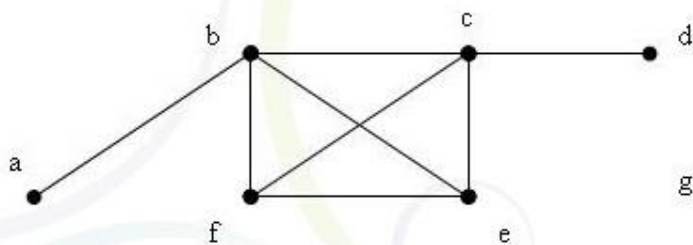


th c, và t ó phát tri n t duy m t cách t ng quát, giúp các em gi i c m t l p bài toán l n. Qua vi c gi i c các bài t p h c s sinh yêu thích, h ng thú v i môn h c h n. tài nghiên c u “ **ng d ng BFS và DFS trong gi i bài t p lý thuy t th** ” s là ngu n tài li u b ích cho giáo viên và h c sinh trong vi c gi ng d y chuyên lý thuy t th .

2.3. Quá trình th c hi n.

a. Các Khái ni m c b n c a lý thuy t th

- **nh ngh a th :** *th là m t c u trúc r i r c bao g m các nh và các c nh n i các nh này. Chúng ta phân bi t các lo i th khác nhau b i ki u và s l ng c nh n i hai nh nào ó c a th .*
- **nh ngh a 1.** *n th vô h ng $G = (V, E)$ bao g m V là t p các nh, và E là t p các c p không có th t g m hai ph n t khác nhau c a V g i là các c nh.*
- **nh ngh a 2.** *a th vô h ng $G = (V, E)$ bao g m V là t p các nh, và E là t p các c p không có th t g m hai ph n t khác nhau c a V g i là các c nh. Hai c nh e_1 và e_2 c g i là c nh l p n u chúng cùng t ng ng v i m t c p nh.*
- **nh ngh a 3.** *Gi th vô h ng $G = (V, E)$ bao g m V là t p các nh và E là t p các c p không có th t g m hai ph n t (không nh t thì t ph i khác nhau) c a V g i là c nh. C nh e c g i là khuyên n u nó có d ng $e = (u, u)$.*
- **nh ngh a 4.** *n th có h ng $G = (V, E)$ bao g m V là t p các nh và E là t p các c p có th t g m hai ph n t khác nhau c a V g i là các cung.*
- **nh ngh a 5.** *a th có h ng $G = (V, E)$ bao g m V là t p các nh và E là t p các c p có th t g m hai ph n t khác nhau c a V g i là các cung. Hai cung e_1, e_2 t ng ng v i cùng m t c p nh c g i là cung l p.*
- **C nh liên thu c:** *Hai nh u và v c a th vô h ng G c g i là k nhau n u (u, v) là c nh c a th G . N u $e = (u, v)$ là c nh c a th ta nói c nh này là liên thu c v i hai nh u và v, ho c c ng nói là n i nh u và nh v, ng th i các nh u và v s c g i là các nh u c a c nh (u, v) .*
- **B c c a nh:** *B c c a nh v trong th $G = (V, E)$, ký hi u $\deg(v)$ là s c nh liên thu c v i nó. N u c nh là khuyên thì c tính là 2.*



Hình 1. Đồ thị vô hướng

Thí d 1.

Xét th cho trong hình 1, ta có

$$\deg(a) = 1, \deg(b) = 4, \deg(c) = 4, \deg(f) = 3,$$

$$\deg(d) = 1, \deg(e) = 3, \deg(g) = 0$$

nh b c 0 g i là nh cô l p. nh b c 1 c g i là nh treo. Trong ví d trên nh g là nh cô l p, a và d là các nh treo.

nh lý 1.

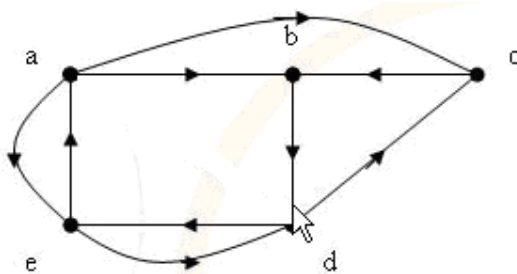
Gi s $G = (V, E)$ là th vô h ng v i m c nh. Khi ó tổng b c c a t t c các nh b ng hai l n s c nh.

Thí d 2.

th v i n nh có b c là 6 có bao nhiêu c nh?

Gi i: Theo nh lý 1 ta có $2m = 6n$. T ó suy ra t ng các c nh c a th là $3n$.

Ta g i bán b c ra (bán b c vào) c a nh v trong th có h ng là s c ung c a th i ra kh i nó (i vào nó) và ký hi u là $\deg^+(v)$ ($\deg^-(v)$)



Hình 2. Đồ thị có hướng

Thí d 3.

Xét th cho trong hình 2. Ta có

$$\deg^-(a)=1, \deg^-(b)=2, \deg^-(c)=2, \deg^-(d)=2, \deg^-(e) = 2.$$

$$\deg^+(a)=3, \deg^+(b)=1, \deg^+(c)=1, \deg^+(d)=2, \deg^+(e)=2.$$



nh lý 2.

Gi s $G = (V, E)$ là th có h ng. Khi ó

T ng t t c các bán b c ra b ng t ng t t c các bán b c vào b ng s c ung.

th vô h ng thu c b ng cách b qua h ng trên các cung c g i là th vô h ng t ng ng v i th có h ng ã cho.

- ng i, chu trình trên th

ng i dài n t nh u n nh v, trong ó n là s nguyên d ng, trên th vô h ng $G = (V, E)$ là dãy $x_0, x_1, \dots, x_{n-1}, x_n$

trong ó $u = x_0, v = x_n, (x_i, x_{i+1}) \in E, i = 0, 1, 2, \dots, n-1$.

ng i nói trên còn có th bi u di n d i d ng dãy các c nh:

$(x_0, x_1), (x_1, x_2), \dots, (x_{n-1}, x_n)$

nh u g i là nh u, còn nh v g i là nh cu i c a ng i. ng i có nh u trùng v i nh cu i (t c là $u = v$) c g i là **chu trình**. ng i hay chu trình c g i là n n u nh không có c nh nào b l p l i.

- **Tính liên thông c a th** - th vô h ng $G = (V, E)$ c g i là liên thông n u luôn tìm c ng i g i a hai nh b t k c a nó.

b. Bi u di n th trên máy tính

- Có nhi u cách khác nhau l u tr các th trong máy tính. S d ng c u trúc d li u nào thì tùy theo c u trúc c a th và thu t toán dùng thao tác trên th ó. Trên lý thuy t, ng i ta có th phân bi t g i a các c u trúc danh sách và các c u trúc ma tr n. Tuy nhiên, trong các ng d ng c th, c u trúc t t nh t th ng là k t h p c a c hai. Ng i ta hay dùng các c u trúc danh sách cho các th th a (sparse graph), do chúng òi h i ít b nh . Trong khi ó, các c u trúc ma tr n cho phép truy nh p d li u nhanh h n, nh ng l i c n l ng b nh l n n u th có kích th c l n.

- Các c u trúc danh sách

Danh sách liên thu c (Incidence list) - M i nh có m t danh sách các c nh n i v i nh ó. Các c nh c a th c có th c l u trong m t danh sách riêng (có th cài t b ng m ng (array) ho c danh sách liên k t ng (linked list)), trong ó m i ph n t ghi thông tin v m t c nh, bao g m: c p nh mà c nh ó n i (c p này s có th t n u th có h ng), tr ng s và các d li u khác. Danh sách liên thu c c a m i nh s chi u t i v trí c a các c nh t ng ng t i danh sách c nh này.

Danh sách k (Adjacency list) - M i nh c a th có m t danh sách các nh k nó (ngh a là có m t c nh n i t nh này n m i nh ó).



Trong th vô h ng, c u trúc này có th gây trùng l p. Ch ng h n n u nh 3 n m trong danh sách c a nh 2 thì nh 2 c ng ph i có trong danh sách c a nh 3. L p trình viên có th ch n cách s d ng ph n không gian th a, ho c có th li t kê các quan h k c nh ch m t l n. Bi u di n d li u này thu n l i cho vi c t m t nh duy nh t tìm m i nh c n i v i nó, do các nh này ã c li t kê t ng minh.

- Các c u trúc ma tr n

Ma tr n liên thu c (*Incidence matrix*) - th c bi u di n b ng m t ma tr n $[b_{ij}]$ kích th c $p \times q$, trong ó p là s nh và q là s c nh, $b_{ij} = 1$ ch a d li u v quan h gi a nh v_i và c nh x_j . n gi n nh t: $b_{ij} = 1$ n u nh v_i là m t trong 2 u c a c nh x_j , b ng 0 trong các tr ng h p khác.

Ma tr n k (*Adjacency matrix*) - m t ma tr n $N \times N$, trong ó N là s nh c a th . N u có m t c nh nào ó n i nh v_i v i nh v_j thì ph n t $M_{i,j}$ b ng 1, n u không, nó có giá tr 0. C u trúc này t o thu n l i cho vi c tìm các th con và o các th .

Ma tr n d n n p (*Admittance matrix*) ho c **ma tr n Kirchhoff** (*Kirchhoff matrix*) hay **ma tr n Laplace** (*Laplacian matrix*) - c nh ngh a là k t qu thu c khi l y ma tr n b c (*degree matrix*) tr i ma tr n k . Do ó, ma tr n này ch a thông tin c v quan h k (có c nh n i hay không) gi a các nh l n b c c a các nh ó.

c. Thu t toán tìm ki m trên th

* Thu t toán tìm ki m theo chi u r ng.

Trong [lý thuy t th](#), tìm ki m theo chi u r ng (BFS) là m t [thu t toán tìm ki m trong th](#) trong ó vi c tìm ki m ch bao g m 2 thao tác: (a) th m m t nh c a th ; (b) thêm các nh k v i nh v a th m vào danh sách có th th m trong t ng lai. Có th s d ng thu t toán tìm ki m theo chi u r ng cho hai m c ích: tìm ki m ng i t m t nh g c cho tr c t i m t nh ích, và tìm ki m ng i t nh g c t i t t c các nh khác. Trong th không có tr ng s , thu t toán tìm ki m theo chi u r ng luôn tìm ra ng i ng n nh t có th . Thu t toán BFS b t u t nh g c và l n l t th m các nh k v i nh g c. Sau ó, v i m i nh trong s ó, thu t toán l i l n l t th m các nh k v i nó mà ch a c th m tr c ó và l p l i. Xem thêm thu t toán [tìm ki m theo chi u sâu](#), trong ó c ng s d ng 2 thao



tác trên nh ng có trình t th m các nh khác v i thu t toán tìm ki m theo chỉ u r ng.

Thu t toán s d ng m t c u trúc d li u hàng i l u tr thông tin trung gian thu c trong quá trình tìm ki m:

1. Chèn nh g c vào hàng i
2. L y ra nh u tiên trong hàng i và th m nó
 - N u nh này chính là nh ích, d ng quá trình tìm ki m và tr v k t qu .
 - N u không ph i thì chèn t t c các nh k v i nh v a th m nh ng ch a c th m tr c ó vào hàng i.
3. N u hàng i là r ng, thì t t c các nh có th n c u ã c th m – d ng vì c tìm ki m và tr v "không th y".
4. N u hàng i không r ng thì quay v b c 2.

```

1  Th t c BFS( $G, v$ ):
2    t o hàng i  $Q$ 
3    chèn  $v$  vào  $Q$ 
4    ánh d u ã th m  $v$ 
5    while  $Q$  còn khác r ng:
6      l y ra ph n t  $t$  u tiên trong  $Q$ 
7      if  $t$  là nh ích:
8        tr v  $t$ 
9      for all cung  $e=(t, o)$  xu t phát t t do
10         if ch a th m  $o$ :
11           ánh d u ã th m  $o$ 
12           chèn  $o$  vào  $Q$ 
    
```

Thu t toán tìm ki m theo chỉ u r ng c dùng gi i nhi u bài toán trong lý thuy t th , ch ng h n nh :

- Tìm t t c các nh trong m t thành ph n liên thông
- Thu t toán Cheney cho vi c d n rác
- Tìm ng i ng n nh t gi a hai nh u và v (v i chỉ u dài ng i tính b ng s cung)



- Kiểm tra xem m có là hai phía
- Thuật toán Cuthill–McKee
- Thuật toán Ford–Fulkerson tìm luồng cực đại trong mạng

*** Thuật toán tìm kiếm theo chiều sâu**

Thuật toán chính của thuật toán là: Giả sử chúng ta đang xét trên đồ thị $G(V, E)$. Thuật toán như sau: Hình thức nào đó của thuật toán tìm kiếm và quá trình duyệt đồ thị. Thuật toán duyệt đồ thị, giả sử hình thức đang xét là u_0 , chúng ta sẽ có hai khả năng xảy ra:

- Nếu thuật toán tìm kiếm v_0 không có u_0 mà chỉ có u_0 thì thuật toán sẽ thành công và quá trình tìm kiếm kết thúc tại u_0 .

- Ngược lại, nếu thuật toán tìm kiếm u_0 thì thuật toán sẽ quay trở lại tìm kiếm.

Như vậy, trong quá trình thuật toán tìm kiếm theo chiều sâu, thuật toán sẽ duyệt theo chiều sâu (Cách Last In First Out - Vào sau ra trước). Do đó, ta có thể thực hiện quá trình này bằng một thuật toán quy hoạch sau:

Procedure DFS(u);

Begin

Visit(u);

Daxet[u]:=True;

For v \in K(u) **do**

if not Daxet[v] **then** DFS(v);

End;

Và thuật toán duyệt đồ thị hoàn chỉnh như sau:

Procedure Find;

Begin

Fillchar(Daxet, SizeOf(Daxet), False);

For u \in V **do**

If not Daxet[u] **then** DFS(u);

End;

Dựa trên lý thuyết, việc tìm kiếm DFS(u) thì toàn bộ các đỉnh cùng thành phần liên thông với u sẽ được duyệt. Thuật toán Visit(u) là thao tác trên đỉnh u trong từng bài toán cụ thể.



ph c t p không gian c a DFS th p h n c a BFS (tìm ki m u tiên chi u r ng). ph c t p th i gian c a hai thu t toán là t ng ng nhau và b ng $O(|V| + |E|)$.

Ý t ng thu t toán

1. DFS trên th vô h ng c ng gi ng nh khám phá mê cung v i m t cu n ch và m t thùng s n ánh d u, tránh b l c. Trong ó m i nh s trong th t ng tr ng cho m t c a trong mê cung.
2. Ta b t u t nh s, bu c u cu n ch vào s và ánh u nh này " ã th m". Sau ó ta ánh d u s là nh hi n hành u.
3. Bây gi , n u ta i theo c nh (u,v) b t k .
4. N u c nh (u,v) d n chúng ta n nh " ã th m" v, ta quay tr v u.
5. N u nh v là nh m i, ta di chuy n n v và l n cu n ch theo. ánh d u v là " ã th m". t v thành nh hi n hành và l p l i các b c.
6. Cu i cùng, ta có th i n m t nh mà t i ó t t c các c nh k v i nó u d n chúng ta n các nh " ã th m". Khi ó, ta s quay lui b ng cách cu n ng c cu n ch và quay l i cho n khi tr l i m t nh k v i m t c nh còn ch a c khám phá. L i ti p t c quy trình khám phá nh trên.
7. Khi chúng ta tr v s và không còn c nh nào k v i nó ch a b khám phá là lúc DFS d ng.

d. Bài t p áp d ng DFS và BFS

Bài toán 1. Bài toán tìm thành ph n liên thông c a th

Cho m t th $G=(V,E)$. Hãy cho bi t s thành ph n liên thông c a th và m i thành ph n liên thông g m nh ng nh nào.

G i ý làm bài:

i u ki n liên thông c a th th ng là m t yêu c u t t y u trong nhi u ng d ng, ch ng h n m t m ng giao thông hay m ng thông tin n u không liên thông thì xem nh b h ng, c n s a ch a. Vì th , vì c ki m tra m t th có liên thông hay không là m t thao tác c n thi t trong nhi u ng d ng khác nhau c a th . D i ây ta xét m t tình hu ng n gi n (nh ng c ng là c b n) là xác nh tính liên thông c a m t th vô h ng v i n i dung c th nh sau: “cho tr c m t th vô h ng, h i r ng nó có liên thông hay không?”.

tr l i bài toán, xu t phát t m t nh tùy ý, ta b t u thao tác tìm ki m t nh này (có th ch n m t trong hai thu t toán tìm ki m ã nêu). Khi



k t thúc tìm ki m, x y ra hai tình hu ng: n u t t c các nh c a th u c th m thì th ã cho là liên thông, n u có m t nh nào ó không c th m thì th ã cho là không liên thông. Nh v y, câu tr l i c a bài toán xem nh m t h qu tr c t i p c a thao tác tìm ki m. ki m tra xem có ph i t t c các nh c a th có c th m hay không, ta ch c n thêm m t thao tác nh trong quá trình tìm ki m, ó là dùng m t bi n m m s nh c th m. Khi k t thúc tìm ki m, câu tr l i c a bài toán s ph thu c vào vi c so sánh giá tr c a bi n m này v i s nh c a th : n u giá tr bi n m b ng s nh thì th là liên thông, n u trái l i thì th là không liên thông. Trong tr ng h p th là không liên thông, k t qu tìm ki m s xác nh m t thành ph n liên thông ch a nh xu t phát. B ng cách l p l i thao tác tìm ki m v i nh xu t phát khác, không thu c thành ph n liên thông v a tìm, ta nh n c thành ph n liên thông th hai, ..., c nh v y ta gi i quy t c bài toán t ng quát h n là xác nh các thành ph n liên thông c a m t th vô h ng b t k .

Nh ta ã bi t, các th t c DFS(u) và BFS(u) cho phép vi ng th m t t c các nh có cùng thành ph n liên thông v i u nên s thành ph n liên thông c a th chính là s l n g i th t c trên. Ta s dùng thêm bi n m Connect m s thành ph n liên thông.

Và vòng l p chính trong các th t c tìm ki m theo chi u sâu hay chi u r ng ch c n s a l i nh sau:

Procedure Find;

Begin

Fillchar(Daxet,SizeOf(Daxet),False);

Connect:=0;

For u V do

If not Daxet[u] then

Begin

Inc(Connect); DFS(u); (*BFS(u)*)

End;

End;

Th t c Visit(u) s làm công vi c ánh s thành ph n liên thông c a nh u:

LienThong[u]:=Connect;

Bài toán 2. Bài toán tìm ng i gi a hai nh c a th



Cho th $G=(V,E)$. V i hai nh s và t là hai nh nào ó c a th .
Hãy tìm ng i t s n t.

G i ý làm bài:

Do th t c DFS(s) và BFS(s) s th m l n l t các nh liên thông v i u
nên sau khi th c hi n xong th t c thì có hai kh n ng:

-N u Daxet[t]=True thì có ngh a: t n t i m t ng i t nh s t i nh t.

-Ng c l i, thì không có ng i n i gi a s và t.

V n còn l i c a bài toán là: *N u t n t i ng i n i nh s và nh t thì
làm cách nào v i t c hành trình (g m th t các nh) t s n t.* V k
thu t l y ng i là: Dùng m t m ng Truoc v i: Truoc[v] là nh tr c c a v
trong ng i. Khi ó, câu l nh If trong th t c DFS(u) c s a l i nh sau:

If not Daxet[v] then

Begin

DFS(v);

Truoc[v]:=u;

End;

Còn v i th t c BFS ta c ng s a l i trong l nh If nh sau:

If not Daxet[w] then

Begin

K t n p w vào Queue;

Daxet[w]:=True;

Truoc[w]:=v;

End;

V i c v i t ng i lên màn hình (ho c ra file) có th có 3 cách:

-V i t tr c t i p d a trên m ng Truoc: H i n nhiên ng i h i n th s ng c t
nh t tr v s nh sau:

$t \leftarrow p1 := \text{Truoc}[t] \leftarrow p2 := \text{Truoc}[p1] \leftarrow \dots \leftarrow s$

-Dùng thêm m t m ng ph P: cách này dùng o ng i t m ng Truoc
có ng i thu n t nh s n nh t.

-Cách th 3: là dùng ch ng trình quy v i t ng i.

Procedure Print_Way(i:Byte);

If i<>s then

Begin

Print_Way(Truoc[i]);



Write(' ',i);

End;

L i g i th t c quy nh sau:

Write(s);

Print_Way(s);

Các b n có th tu ch n cách mà mình thích nh ng thi t ngh ó ch a ph i là v n quan tr ng nh t. N u tính ý d a vào th t th m nh c a thu t toán tìm ki m theo chi u r ng BFS ta s có m t nh n xét r t quan tr ng, ó là: N u có ng i t s n t, thì ng i tìm c do thu t toán tìm ki m theo chi u r ng cho chúng ta m t hành trình c c ti u v s c nh.

Bài toán 3: Truy n tin

M t l p g m N h c viên, m i h c viên cho bi t nh ng b n mà h c viên ó có th liên l c c (chú ý liên l c này là liên l c m t chi u, ví d : B n An có th g i tin t i B n Vinh nh ng B n Vinh thì ch a ch c ã có th g i tin t i B n An). Th y ch nh i m ang có m t thông tin r t quan tr ng c n thông báo t i t t c các h c viên c a l p (tin này ph i c truy n tr c ti p). t i t ki m th i gian, th y ch nh n tin t i l s h c viên r i sau ó nh các h c viên này nh n l i cho t t c các b n mà các h c viên ó có th liên l c c, và c l n l t nh th làm sao cho t t c các h c viên trong l p u nh n c tin .

Câu h i

Có ph ng án nào giúp th y ch nh i m v i m t s ít nh t các h c viên mà th y ch nh i m c n nh n?

G i ý làm bài:

- Có th nh n th y bài toán này chính là bài toán 1 ã phát bi u phía trên. Có th coi m i h c sinh là m t nh c a th . Hai h c sinh có th liên l c c v i nhau là m t c nh. T ó suy ra bài toán này là . Bài toán tìm thành ph n liên thông c a th .

Bài toán 4: ng i n s 0

M i m t s nguyên d ng u có th bi u di n d i d ng tích c a 2 s nguyên d ng X, Y sao cho $X \leq Y$. N u nh trong phân tích này ta thay X b i $X-1$ còn Y b i $Y+1$ thì sau khi tính tích c a chúng ta thu c ho c là m t s nguyên d ng m i ho c là s 0.

Ví d : S 12 có 3 cách phân tích $1*12, 3*4, 2*6$. Cách phân tích th nh t cho ta tích m i là 0 : $(1-1)*(12+1) = 0$, cách phân tích th hai cho ta tích m i 10 :



$(3-1)*(4+1) = 10$, còn cách phân tích th ba cho ta $7 : (2-1)*(6+1)=7$. N u nh k t qu là khác không ta l i l p l i th t c này i v i s thu c. Rõ ràng áp d ng liên ti p th t c trên, cu i cùng ta s n c s 0, không ph thu c vào vi c ta ch n cách phân tích nào ti p t c

Yêu c u: Cho tr c s nguyên d ng $N (1 \leq N \leq 10000)$, hãy a ra t t c các s nguyên d ng khác nhau có th g p trong vi c áp d ng th t c ã mô t i v i N .

D li u: Vào t file Zeropath.Inp ch a s nguyên d ng N .

K t qu : Ghi ra file v n b n Zeropath.Out :

Dòng u tiên ghi K là s l ng s tìm c

Dòng ti p theo ch a K s tìm c theo th t t ng d n b t u t s 0.

L u ý: Có th có s xu t hi n trên nhi u ng bi n i khác nhau, nh ng nó ch c tính m t l n trong k t qu .

Ví d :

ZEROPATH.INP	ZEROPATH.OUT
12	6 0 3 4 6 7 10

G i ý làm bài:

n gi n là sau m i l n phân tích thì ch c ch n k t qu m i luôn nh h n s ó. Vì v y ta ch c n L u tr d i m ng $A: [0..10000]$ of boolean ; trong ó $A[i] = \text{true}$ n u nó xu t hi n trên ng i ó, ng c l i thì $A[i] = \text{false}$. B ng cách loang theo chi u sâu, chúng ta s ánh d u các s n u nó c dùng n, cho n khi không th nào loang c n a thì d ng.

Bài toán 5. Con ng a

M t bàn c hình ch nh t kích th c $M \times N$, M, N nguyên d ng không l n h n 100. Bàn c chia thành các ô vuông n v b ng các ng song song v i các c nh. Các dòng ô vuông ánh s t l n M t trên xu ng d i, các c t ánh s t l n N t trái sang ph i. Cho tr c m t s nguyên d ng $K \leq 1000$. M t con ng a ng ô $[u, v]$ và nh y không quá k b c.

Yêu c u: Hãy cho bi t con ng a có th nh y n bao nhiêu ô khác ô $[u, v]$ trên bàn c và ó là nh ng ô nào (khi ng t i m t ô, con ng a có th nh y t i ô i nh c a hình ch nh t kích th c 2×3).

D li u: Vào t file MA.INP trong ó :

Dòng u tiên ghi hai s M, N



Dòng th hai ghi s K

Dòng th ba ghi hai s U,V

K t qu : Ghi ra file MA.OUT :

Dòng u tiên ghi S là s ô con ng a có th nh y n

Ti p theo là S dòng, m i dòng ghi ch s dòng và ch s c t c a m t ô mà con ng a có th nh y n.

Ví d :

MA.INP

MA.OUT

5 5

6

1

1 1 1 5 3 1 3 5 4 2 4 4

2 3

G i ý làm bài

Chúng ta s loang theo chi u sâu, tìm ki m xem nh ng ô nào con mã có th t chân n trong vòng K b c nh y.

Bài toán 6: ng i trên l i ô vuông

Cho m t l i ô vuông kích th c $N \times N$. Các dòng c a l i c ánh s t l n N t trên xu ng d i, các c t c a l i c ánh s t l n N t trái qua ph i. Ô n m trên giao c a dòng i, c t j s c g i là ô (i, j) c a l i. Trên m i ô (i, j) c a l i ng i ta ghi m t s nguyên d ng a, $i, j = 1, 2, \dots, N$. T m t ô b t k c a l i c phép di chuy n sang ô có chung c nh v i nó. Th i gian di chuy n t m t ô này sang m t ô khác là 1 phút. Cho tr c th i gian th c hi n di chuy n là K (phút), hãy xác nh cách di chuy n b t u t ô (1, 1) sao cho t ng các s trên các ô di chuy n qua là l n nh t (M i ô c a l i có th di chuy n qua bao nhiêu l n c ng c).

D li u: Vào t file v n b n NETSUM.INP:

- ☐ Dòng u tiên ch a các s nguyên d ng N, K ($2 \leq N \leq 100$), $1 \leq K \leq 10000$).
- ☐ Dòng th i trong s N dòng ti p theo ch a các s nguyên ai1, ai2..., aiN, $0 < a_i \leq 10000$.

(Các s trên cùng m t dòng c ghi cách nhau b i ít nh t m t d u cách).

K t qu : Ghi ra file v n b n NETSUM.OUT:

- ☐ Dòng u tiên ghi t ng s các s trên ng di chuy n tìm c.
- ☐ K dòng ti p theo m i dòng ghi to c a m t ô trên ng di chuy n (b t u t ô (1, 1)).



Ví d :

NETSUM.INP	NETSUM.OUT
5 7	2 1
1 1 1 1 1	1 1
1 1 3 1 9	1 2
1 1 6 1 1	1 3
1 1 3 1 1	2 3
1 1 1 1 1	2 4
2 3	
2 4	

G ý làm bài:

Loang các ô có th n c a các ng i trên l i. Tìm cách i nào có ng i mà t ng l n nh t thì s l y.

Bài toán 7: Bàn c th



2793. Bàn c th

Mã bài: CHESSCBG

M t bàn c th là m t b ng g m 4 dòng, 4 c t. M i th c là m t cách s p x p 8 quân c , hai quân khác nhau hai ô khác nhau. Bài toán t ra là cho hai th c 1 và 2, hãy tìm m t s ít nh t b c di chuy n quân chuy n t th 1 sang th 2; m t b c di chuy n quân là m t l n chuy n quân c sang ô tr ng k c nh v i ô quân c ang ng.

D li u vào

T file v n b n g m 8 dòng, m i dòng là m t xâu nh phân dài 4 mà s 1/0 t ng ng v i v trí có ho c không có quân c . B n dòng u là th c 1, b n dòng sau là th c 2.

D li u ra

G m 1 dòng duy nh t là s b c chuy n quân ít nh t

Ví d

D li u vào:

1111
0000
1110



0010

1010

0101

1010

0101

D li u ra :

4

G i ý làm bài :

Chúng ta s gi i quy t nh m t ph ng pháp h t s c n gi n : Tìm ki m theo chi u r ng. Ta s coi m t tr ng thái l c a b ng là m t nh c a th m i. M i l n di chuy n m t quân c trên bàn thì nó s t o ra m t tr ng thái m i c a b ng, t c là s n m t nh m i. Trong bài toán này chúng ta s ch xét v i ($m=n=4$). T c là file input, không có dòng u tiên.

M i tr ng thái c a b ng là m t lo t các ô có giá tr 0 và 1. Chúng ta s tr i nó ra thành m t hàng thì s t o ra m t b ng m t chi u ch toàn các s 1 và 0. Vì có 16 ô, nên m i b ng nh v y s t ng ng v i h nh phân c a m t s nào ó n m trong word (16 bit). T c là s nh c a th có th có s là 216.

a1	a2	a3	a4
a5	a6	a7	a8
a9	a10	a11	a12
a13	a14	a15	a16

B ng 1

B ng m i sau khi tr i nh sau :

a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16

B ng 2

Khi di chuy n các quân thì các quân (b ng 1) có th i t i 4 ô bên c nh n u có th (tr tr ng h p i ra ngoài b ng). T c là t ng ng b ng 2, gi s t i v trí A_i thì nó có th có i n nh ng ô : A_i-1 , A_i+1 , A_i-4 , A_i+4 (Ph i tr nh ng tr ng h p nó i ra ngoài b ng). Quá trình di chuy n quân 1 nh v y t c là bít th i s c t t, còn các bít c n s c b t. Loang theo chi u r ng c a quá trình chuy n bit cho n khi chuy n n c tr ng thái



mong mu n. ng di chuy n ó chính là cách di chuy n c v i s b c ít
nh t n tr ng thái mong mu n.

Bài toán 8. S rở ràng

Các nhà toán h c a voà lý thuy t nhi u cách phân lo i s , ví d , v i
các s nguyên ta có s ch n và s l , s nguyên t và h p s , s chính
ph ng và không chính ph ng... Bob c ng mu n t d u n c a mình
trong l nh v c phân lo i s . Bob chia các s nguyên d ng thành 2 lo i: *rở
ràng là lu n qu n*. Vi c xác nh m t s thu c lo i nào c th c hi n theo
gi i thu t sau: V i s nguyên d ng n, ta t o s m i b ng cách l y t ng bình
ph ng các ch s c a nó, v i s m i này ta l p l i công vi c trên. N u
trong quá trình trên, ta nh n c s m i là 1, thì s n ban u c g i là s
rở ràng. Ví d v i n=19, ta có:

19 82(=1²+9²) 68 100 1

Nh v y, 19 là s rở ràng.

Không ph i m i s u rở ràng. Ví d , v i n=12 ta có:

12 5 25 29 85 89 145 42 20 4 16 37 58 89 145

R t thú v v i cách phân lo i c a mình, Bob mu n bi t, trong th c t , s
rở ràng nhi u hay ít?.

Yêu c u: Cho hai s nguyên d ng A và B (1 A B 10000000). Hãy xác nh
K-s l ng s rở ràng n m trong kho ng [A,B].

D li u: Vào t file CLEAR.INP g m 1 dòng ch a 2 s nguyên A và B

K t qu : a ra file v n b n CLEAR.OUT s nguyên K

Ví d :

CLEAR.INP

CLEAR.OUT

2 20

4

G i ý làm bài:

B ng cách loang theo chi u sâu, chúng ta s ánh d u các s n u nó
c dùng n là s rở ràng.

Bài toán 9:

T t p các bài có trên SPOJ (oi)

2195. i u ki n th i ti t

Mã bài: WEATHER

Hãng hàng không OlympAirways th c hi n các chuy n bay gi a n sân bay
c ánh s t 1 n n. H th ng các chuy n bay c thi t l p sao cho
gi a 2 sân bay b t k c ph c v b i hãng luôn có m t ng bay bao



g m m t ho c nhi u chuy n bay tr c ti p gi a hai sân bay. M i chuy n bay th c hi n vi c di chuy n gi a hai thành ph theo c hai chi u.

Trung tâm i u khi n c a hăng a ra khái ni m dính k t gi a c p hai sân bay A và B c xác nh nh là s l ng các chuy n bay mà vi c không th c hi n m t trong s chúng (các chuy n bay khác v n th c hi n bình th ng) d n n không th bay t sân bay A n sân bay B.

M t nghiê n c u cho bi tr ng, trong i u ki n th i ti t x u, t ng dính k t gi a các c p sân bay ph i t n m t giá tr nh t nh thì h th ng ng bay m i c g i là an toàn.

Yêu c u: Hãy giúp trung tâm i u khi n tính t ng dính k t gi a m i c p sân bay.

D li u

Dòng u tiên ch a s nguyên n (1 n 100)

Dòng th hai ch a s nguyên m (1 m 5000) - s l ng các chuy n bay

M i dòng trong s m dòng ti p theo ch a thông tin v m t chuy n bay, bao g m hai s nguyên d ng trong kho ng t 1 n n: ch s c a hai sân bay c n i b i chuy n bay.

K t q a

In ra 1 s nguyên duy nh t là t ng dính k t gi a m i c p sân bay (A, B) (v i $A < B$).

Ví d

D li u:

5

5

1 2

4 2

4 5

3 2

3 1

K t q a

10

G i ý làm bài:



Bài này v i m i c nh chúng ta c n ki m tra xem nó có là c u không (dùng DFS ho c BFS). N u là c u thì t ng k t đỉnh s t ng lên m t giá tr = tích c a các nh thu c hai mi n mà c nh ó làm c u.

Bài toán 10:

2719. Bãi c ngon nh t

Mã bài: VBGRASS

Bessie d nh c ngày s nhai c xuân và ng m nhìn c nh xuân trên cánh ng c a nông dân John, cánh ng này c chia thành các ô vuông nh v i R ($1 \leq R \leq 100$) hàng và C ($1 \leq C \leq 100$) c t. Bessie c gì có th m c s khóm c trên cánh ng.

M i khóm c trên b n c ánh d u b ng m t ký t '#' ho c là 2 ký t '#' n m k nhau (trên ng chéo thì không ph i). Cho b n c a cánh ng, hãy nói cho Bessie bi t có bao nhiêu khóm c trên cánh ng.

Ví d nh cánh ng d i đây v i $R=5$ và $C=6$:

.#....

..#...

..#..#

...##.

.#....

Cánh ng này có 5 khóm c : m t khóm hàng u tiên, m t khóm t o b i hàng th 2 và th 3 c t th 2, m t khóm là 1 ký t n m riêng r hàng 3, m t khóm t o b i c t th 4 và th 5 hàng 4, và m t khóm cu i cùng hàng 5.

D li u

Dòng 1: 2 s nguyên cách nhau b i d u cách: R và C

Dòng 2.. $R+1$: Dòng $i+1$ mô t hàng i c a cánh ng v i C ký t , các ký t là '#' ho c '.'.

K t qu

Dòng 1: M t s nguyên cho bi t s l ng khóm c trên cánh ng.

Ví d

D li u

5 6

.#....

..#....

..#...#

...##.



.#....

Kết quả

5

Gợi ý làm bài:

Coi mặt tích trên cánh cửa là mặt phẳng tọa độ Oxy . Số đo góc thu được tính toán theo chiều dương của trục Ox bắt đầu từ trục Oy .

Bài toán 11.

2969. Bin Laden

Mã bài: BINLADEN

Bin Laden

Trùm khủng bố Bin Laden trú ẩn trong 1 căn hầm có chiều sâu xuống mặt đất M mét, mặt đất có N phòng. Các phòng có cách bố trí các cửa ra vào khó phá. Các phòng có cửa xuống phòng ngay phía dưới và 2 phòng bên. Trên mặt đất có N cửa xuống N phòng tầng -1. Bin Laden trú ẩn dưới cùng (tầng - M) phòng tầng N (phòng bên phải nhất). Mỗi căn hầm làm bằng một kim loại khác nhau vì độ dày khác nhau nên việc phá cửa cần thời gian khác nhau.

Bạn hãy tìm cách đi từ mặt đất xuống phòng của Bin Laden nhanh nhất không hề thoát mặt đất.

Dữ liệu

Dòng 1 ghi M và N

Dòng 2 đến $2M + 1$, dòng chẵn ghi N số, dòng lẻ ghi $N - 1$ số là chi phí phá cửa.

Kết quả

Ghi ra 1 số là thời gian nhỏ nhất để vào phòng của Bin Laden

Ví dụ

Dữ liệu

4 2

99 10

1

10 99

1

99 10

1



10 99

1

K t qu

44

```

+--99--+--10--+
|         |         |
|         |         |
|         |         |
+--10--+--99--+
|         |         |
|         |         |
|         |         |
+--99--+--10--+
|         |         |
|         |         |
|         |         |
+--10--+--99--+
|         |         |
|         |         |
|         |         |
+-----+-----+

```

i theo ng zigzac

Gi i h n

- $1 \leq M \leq 2222$
- $1 \leq N \leq 10$
- Chi phí c a các cánh c a thu c $[0, 1000]$.

G i ý làm bài:

Coi m i phòng là m t nh c a th . Hai nh có ng n i n u các phòng k c nh, và có tr ng s b ng th i gian phá t ng ng n cách. Bài toán tr thành tìm ng i ng n nh t t l phòng nào ó c a t ng trên xu ng m t phòng cu i cùng c a t ng đ i.

Bài toán 12:

3892. Tr ng cây

Mã bài: GARDEN25

Nhà sherry có 1 khu v n r tr ng và tr ng nhi u lo i cây. ón t t n m 2010 sherry s tr ng th t nhi u mai và ào. Và ch có mai và ào mà thôi.



Khu v n nhà sherry có d ng hình ch nh t, kích th c $M \times N$. Trên ó có 1 s ô c ánh d u tr ng cây. t ng tính th m m c a khu v n sherry mu n s cây mai và ào trong khu v n chênh l ch nhau không quá 1. ng th i s cây mai, ào trên m i hàng, c t c a khu v n c ng chênh l ch nhau không quá 1.

Input

Dòng 1: ghi 2 s nguyên M, N ($1 \leq M, N \leq 250$)

M dòng ti p theo: M i dòng ghi N s , trong ó s th j c a hàng th i b ng 1/0 t ng ng v i ô (i, j) có/không tr ng cây.

Output

G m M dòng: M i dòng ghi N s nguyên, các ô không tr ng cây ghi ra 0, các ô tr ng cây có giá tr 1/2 t ng ng ó tr ng mai/ ào.

Example

Input:

```
4 4
1 0 1 0
0 1 0 1
1 0 1 0
0 1 0 1
```

Output:

```
2 0 1 0
0 2 0 1
1 0 2 0
0 1 0 2
```

G i ý làm bài:

Ta coi m i hàng , m i c t là m t nh c a th . N u ô (i, j) có giá tr $\langle \rangle$ 0 thì nh hàng i n i v i nh c t j. Bài toán tr thành :

Tìm các tô các c nh c a m t th b ng hai màu , sao cho :

- v i m i nh thì chênh l ch hai màu tô các c nh n i nó chênh l ch không quá 1 .
- V i c th chúng c ng chênh l ch nhau không quá 1 . Chúng ta có ph ng pháp gi i quy t bài toán này nh sau :
- Nh n xét 1 : N u xu t phát t m t nh b c l và i m t cách b t k theo các cung c a th , m i cung i qua ch m t l n thì tr ng thái t c ng ph i



x y ra t i m t nh b c l khác (s nh b c l n u có trong th là m t s ch n)

- Nh n xét 2 : N u xu t phát t m t nh b c ch n trong th không có nh b c l và i m t cách b t k các cung c a th , m i cung i qua ch m t l n thì tr ng thái t c ng ph i x y ra t i chính nh xu t phát . Tr ng thái t c ng là tr ng thái mà t i nh v a t i không còn cung nào ch a i qua . D a vào hai nh n xét chúng ta có :
- Tr ng h p 1 : Khi th còn nh b c l . Ch n m t nh l b t k xu t phát . B ng m t cách i b t k qua các cung c a th màu ch a c tô , m i cung i qua ta tô xen k b ng hai màu cho n khi t c ng . Trong tr ng h p này , t i nh b c l k t thúc ng i trên , không còn cung nào ch a nó ch a c tô , ng th i , t i nh xu t phát , s cung còn l i ch a tô (n u có) là m t s ch n , còn t i các nh còn l i trên ng i s cung c tô b ng các màu b ng nhau
- Tr ng h p 2 : Khi th ch còn nh b c ch n Trong tr ng h p này , t t c các cung k v i các nh b c l (n u có) c a th ban u u ã c tô . Ch n m t nh nào ó còn có cung ch a tô ch a nó làm nh xu t phát và c ng i m t cách b t k theo các cung ch a tô cho n khi t c tr ng thái k t thúc (t i nh xu t phát) . B ng cách tô màu các cung xen k trên l trình ã i qua . Khi ó số l ng các cung c tô hai màu c tô k v i m i nh trên l trình là b ng nhau .

2.4. K t qu thu c

- H c sinh sau khi h c chuyên này s h ng thú v i vi c h c lý thuy t th .
Khi g p m t bài toán v lý thuy t th s t tin làm bài. DFS và BFS còn là n n t ng d y các ph n lý thuy t khác trong chuyên th



3. Ph n k t lu n

- **BFS** và **DFS** là nh ng thu t toán tìm ki m c b n nh ng r t quan tr ng trên th . Nh ng thu t toán này s là n n móng quan tr ng có th xây d ng và thi t k nh ng thu t gi i khác trong lý thuy t th . Tuy nhiên có th th y r ng ph ng pháp này còn h n ch khi s l ng các ph n t c a t p D l n. Nó th hi n ch th i gian tính toán cho ra k t qu th ng không ch p nh n c. Do ó trong ph ng pháp Tìm ki m theo DFS và BFS c n ph i b sung các ph ng pháp cho phép b qua ho c g p m t s ph n t . i u này c i thi n áng k th i gian th c hi n ch ng trình.

- Ph ng pháp Tìm ki m theo DFS và BFS là m t trong nh ng ph ng pháp d hi u nh t v i h c sinh và có th áp d ng gi i r t nhi u bài toán t i u v i d li u nh (th ng t n 50% n 60% s tets c a m t bài thi).

4. Tài li u tham kh o:

1. C u trúc d li u và gi i thu t – Lê Minh Hoàng (DHSP Hà N i)
2. Tài li u t p hu n phát tri n chuyên môn giáo viên Tin h c - Nhi u tác gi
3. Tài li u h i th o phát tri n chuyên môn giáo viên Tin h c - Nhi u tác gi
4. Thu t toán quay lui – Lê S Hùng (H ng S n – Hà T nh)
5. Tài Li u sách giáo khoa chuyên tin t p 1,2 - Nhi u tác gi
6. VNOI - Olympic tin h c Vi t Nam - M c l c di n àn - Forum



Chuyên x p lo i B

Chuyên

M T S NG D NG C A DFS

Ngô Trung T ng-GV tr ng THPT chuyên Lê Hồng Phong-Nam nh

R t nhi u thu t toán trên th c xây d ng d a trên c s duy t qua t t c các nh c a th sao cho m i nh c a nó c th m úng m t l n. Vì v y, vì c xây d ng nh ng thu t toán cho phép duy t m t cách có h th ng t t c các nh c a th cùng các ng d ng c a nó là m t v n quan tr ng thu hút s quan tâm nghiên c u c a nhi u tác gi . Nh ng thu t toán nh v y g i là thu t toán tìm ki m trên th . Trong chuyên này tôi s gi i thi u m t s ng d ng c a thu t toán tìm ki m theo chi u sâu (DFS-Depth First Search) vào vi c gi i m t s bài toán trên th .

I. Th t duy t n và duy t xong:

Th t c: DFS(u)

- Khi b t u vào th t c DFS(u) ta nói nh u c duy t n hay c th m (discover), t c là t i th i i m ó quá trình tìm ki m theo chi u sâu b t u, t u s xây d ng nhánh cây DFS g c u.
- Khi chu n b thoát kh i th t c DFS(u) lùi v , ta nói nh u c duy t xong (finish), t c là t i th i i m ó quá trình tìm ki m theo chi u sâu k t thúc.

Trong th t c DFS ta thêm vào bi n m Time xác nh th i i m duy t n d[u] và th i i m duy t xong f[u]

- Mô hình cài t thu t toán DFS có thêm vào th t duy t n và duy t xong

Procedure DFS($u \in V$)

Begin

Time:=Time+1;

d[u]:=Time;

output $\leftarrow u$; // th m u

for $\forall v \in V: (u,v) \in E$ do // duy t m i nh n i t v t i u

// n u v ch a th m g i qui tìm ki m theo chi u sâu t v

If d[v]=0 then DSF(v)

Time:=Time+1;

f[u]:=Time;



end;

- Thuật duy t n và duy t xong có ý nghĩa r t quan tr ng trong nhi u thu t toán có s d ng DFS, nh tìm thành ph n liên thông m nh, tìm c u, kh p c a th ,...

II. M t s ng d ng

1. Tìm thành ph n liên thông m nh trên th có h ng (thu t toán Tarjan)

a.Ý t ng: Trong thu t toán Tarjan li t kê các thành ph n liên thông m nh trên th có h ng d a trên thu t toán tìm ki m theo chi u sâu DFS.

- Cài t thu t toán d a trên th t duy t n.

+ Number[u] là th t duy t n c a nh u

+ Color[u] là màu nh u, n u Color[u] là white (màu tr ng) thì nh u ch a c th m, n u là Gray(màu xám) thì nh u ã c th m nh ng ch a duy t xong, n u là Black (màu en) thì nh u ã b xóa kh i nhánh cây DFS.

+ Low[u] là giá tr Number[.] nh nh t trong các nh mà có th n c t m t nh v nào ó c a nhánh DFS g c u b ng m t cung. Tính Low[u] nh sau:

Kh i t o Low[u]:=+ , xét nh v n i t u có hai kh n ng

++ N u v có màu Gray (xám):

Low[u]:=min(Low[u],Number[v])

++ N u v có màu White (tr ng):

Th m V

Low[u]:=min(Low[u],Low[v])

+ Khi duy t xong m t nh u: so sánh Low[u] và Number[u], n u Low[u] >= Number[u], thì u là nh u tiên trong m t thành ph n liên thông m nh thu c cây DFS g c u, b i vì không có cung n i t nh DFS g c u t i m t nh th m tr c.

b. Mô hình cài t thu t toán Tarjan

Procedure Tarjan(u);

Begin

Time:=Time+1;

Number[u]:=Time;

Low[u]:=+

Color[u]:=Gray;

Push(u); // y u vào stack

For $\forall v \in V; (u,v) \in E$ do



```

If Color[v]=Gray then // nh v ã th m r i
    Begin
        Low[u]:=Min(Low[u],Number[v]);
    End
Else
    If Color[v]=White then // nh v ch a
    th m
    Begin
        Tarjan(v);
        Low[u]:=Min(Low[u],Low[v]);
    End;
If Low[u]>=Number[u] then //u là ch t
    Begin
        //thông báo thành ph n liên thông
        Repeat
            v:=pop;
            Output→v
            Color[v]:=Black;
            //xóa các nh trong m t tplt v a
            tìm c
        Until v=u;
    End;
End;
BEGIN
    Time:=0;
    For i:=1 to n do Number[i]:=0;
    For i:=1 to n do
        If Number[i]=0 then
            Tarjan(i);
END.

```

c. M t s ví d :

Bài Truy n tin (SPOJ)

M t l p g m N h c sinh, m i h c sinh cho bi t nh ng b n mà h c sinh ó có th liên l c c (chú ý liên l c này là liên l c m t chi u : u có th g i tin t i v nh ng v thì ch a ch c ã có th g i tin t i u).



Th y ch nhi m ang có m t thông tin r t quan tr ng c n thông báo t i t t c các h c sinh. t i t ki m th i gian, th y ch nh n tin t i l s h c sinh r i sau ó nh các h c sinh này nh n l i cho t t c các b n mà các h c sinh ó có th liên l c c, và c l n l t nh th làm sao cho t t c các h c sinh trong l p u nh n c tin .

Hãy tìm m t s ít nh t các h c sinh mà th y ch nhi m c n nh n.

Input

- Dòng u là N, M ($N \leq 800$, M là s l ng liên l c l chi u)
- M t s dòng ti p theo m i dòng g m 2 s u, v cho bi t h c sinh u có th g i tin t i h c sinh v

Output

- G m l dòng ghi s h c sinh c n th y nh n tin.

Example

Input	Output
12 15	2
1 3	
3 6	
6 1	
6 8	
8 12	
12 9	
9 6	
2 4	
4 5	
5 2	
4 6	
7 10	
10 11	
11 7	
10 9	

H ng d n:

- Li t kê các thành ph n liên thông m nh c a th
- Xây d ng th m i:
 - + M i nh là m t thành ph n liên thông m nh



+ M i cung là cung th ban u mà n i t thành ph n liên thông m nh này sang thành phân liên thông m nh kia.

- Trên th m i, ta tìm s nh không có cung i vào. ó chính là k t qu c a bài toán.

Ch ng trình

```
uses      math;
const
    fi='';
    fo='';
    maxN=800+5;
    oo=maxn+5;
type
    TColor=(White,Gray,Black);
    TEdge=record
        u,v:longint;
    end;
var
    top,ans,n,m,time,res:longint;
    a:array[0..maxN,0..maxN] of boolean;
    color:array[0..maxN] of TColor;
    dd,s,number,low:array[0..maxN] of longint ;
    e:array[0..maxN*maxN] of TEdge;
    count:array[0..maxN] of boolean;
procedure read_input;
var    i,u,v:longint;

begin
    fillchar(a,sizeof(a),false);
    assign(input,fi);
    reset(input);
    readln(n,m);
    for i:=1 to m do
        begin
            readln(u,v);
            a[u,v]:=true;
```



```
                                e[i].u:=u;
                                e[i].v:=v;

                                end;
                                close(input);
end;
procedure write_output;
begin
    assign(output,fo);
    rewrite(output);
    write(res);
    close(output);
end;
procedure Tarjan(u:longint);
var    v:longint;
begin
    time:=time+1;
    number[u]:=time;
    low[u]:=oo;
    color[u]:=gray;
    top:=top+1;
    s[top]:=u;//bo u vao ngan xep
    for v:=1 to n do
        if a[u,v] then
            begin
                if color[v]=Gray then
                    low[u]:=min(low[u],number[v])
                else
                    if color[v]=white then
                        begin
                            Tarjan(v);
                                low[u]:=min(low[u],low[v]);
                        end;
                    end;
                if low[u]>=number[u] then
                    begin
                        ans:=ans+1;//dem duoc 1 thanh phan lien thong manh
                        repeat
```



```
        v:=s[top]; //lay dinh v ra khoi ngan xep
        top:=top-1;
        color[v]:=black;
        dd[v]:=ans;
    until u=v;
end;
end;
procedure solve;
var    u,i:longint;
begin
    time:=0;
    for u:=1 to n do
        begin
            color[u]:=white;
            count[u]:=true;
        end;
    ans:=0;
    for u:=1 to n do
        if color[u]=white then
            begin
                top:=0;
                Tarjan(u);
            end;
    //danh dau dinh trong do thi moi co cung di vao
    for i:=1 to m do
        if dd[e[i].u]<>dd[e[i].v] then
            count[dd[e[i].v]]:=false;
    //dem so dinh trong do thi moi khong co cung di vao
    res:=0;
    for i:=1 to ans do
        if count[i] then
            res:=res+1;
    end;
BEGIN
    read_input;
    solve;
    write_output;
END.
```



Bì n i s (Mã bài: NUMBER)

Cho M máy bì n i s c ánh s t l n M và l s nguyên d ng N. Ho t ng c a máy i c xác nh b i c p s nguyên d ng (a_i, b_i) ($1 \leq a_i, b_i \leq N$). Máy nh n u vào là s nguyên d ng a_i và tr l i u ra s nguyên d ng b_i .

Ta nói m t s nguyên d ng X có th bì n i thành s nguyên d ng Y n u ho c $X=Y$ ho c t n t i d ã h u h n các s nguyên d ng $X = P_1, P_2, \dots, P_k = Y$ sao cho i v i 2 ph n t liên ti p P_i và P_{i+1} b t k trong d ã, luôn tìm c l trong s các máy ã cho bì n i P_i thành P_{i+1} .

Cho tr c l s nguyên d ng T ($T \leq N$). Hãy b sung thêm l s ít nh t các máy bì n i s b t kì s nguyên d ng nào t l n N u có th bì n i thành T

Input

- Dòng 1: 3 s nguyên d ng N, M, T ($1 \leq N, M, T \leq 10^4$)
- M dòng ti p theo m i dòng ch a l c p s t ng ng v i m t máy bì n i s .
Các s trên m t dòng cách nhau b i l d u cách

Output

Ghi ra 1 dòng duy nh t ch a l s nguyên d ng là s l ng máy bì n i s c n thêm

Example

Input	Output
6 4 5 1 3 2 3 4 5 6 5	1

H ng d n:

- Li t kê các thành ph n liên thông m nh c a th
- Xây d ng th m i:
+ M i nh là m t thành ph n liên thông m nh
+ M i cung là cung th ban u mà n i t thành ph n liên thông m nh này sang thành ph n liên thông m nh kia.
- Trên th m i, ta tìm s nh không có cung i ra. ó chính là k t qu c a bài toán.



Cài t gì ng bài truy n tin, ta s a m s cung i vào b ng m s cung i ra.

2. Li t kê các c nh c u, nh kh p c a th vô h ng

T ng t thu t toán Tarjan, ta nh ngh a thêm Low[u] và Number[u]. Hãy ý cung DFS(u,v) (u là nút cha c a v trên cây DFS).

a. Li t kê các c nh c u:

- N u t nhánh DFS g c v không có cung nào ng c lên phía trên v, có ngh a là t m t nh thu c nhánh DFS g c v i theo các cung nh h ng ch i c t i nh ng nh n i b trong nhánh DFS g c v mà thôi ch không th t i c u $\Rightarrow (u,v)$ là m t c u. V y (u,v) là m t c u n u và ch n u Low[v] \geq Number[v].
- Thu t toán li t kê các c u c a th : (ng d ng c ch tô màu cho các nh c a th : m i nh c tr ng b i 3 màu: ch a th m (màu White); ang th m (màu Gray); th m xong (màu Black).

- Cài t:

```
procedure DFS(u:PointType);
{Global: G, Color, Time, D (Number), L (Low)}
Var
    v : PointType;
    pq:List;
Begin{DFS}
    inc(Time);
    D[u]:=Time;
    L[u]:=0; //maxlongint
    Color[u]:=Gray;
    pq:=G[u];
    while pq<>nil Do
    begin
        v:=pq^.v;
        If Color[v]=White Then
        begin
            parent[v]:=u;
            DFS(v);
            if L[v]<L[u] then L[u]:=L[v];
        end
    end
```



```

else
  if
    Color[v]=Gray)and(parent[u]<>v)and(D[v]<L[u]) then
      L[u]:=D[v];
      pq:=pq^.link;
    end{while};
    if (u<>1)and(L[u]>=D[u]) then
      begin
        {parent[u]-u là m t c nh c u}
        inc(S);
        E[S].v:=u;
        E[S].u:=Parent[u];
      end;
      Color[u]:=Black;
    End {DFS};
  
```

- M t s ví d :

Nâng c p ng i. (thi HSG Nam nh)

Hì n nay nhi u thành ph có c s h t ng kém phát tri n cho nên c nh t c ng r t hay x y ra. Nhà n c ã có k ho ch nâng c p nhi u con ng trong thành ph g i m thi u n n t c ng. Hàng ngày m i ng i v n c n ph i i l i trên các con ng nên vi c nâng c p ng c n ph i nhanh chóng hoàn thành. Hì n t i, h th ng giao thông c a thành ph ND u áp ng c nhu c u i l i t a i m A n a i m B (A, B là hai a i m b t kì thu c thành ph ND). i t A n B có th b ng con ng n i t A n B ho c thông qua m t hay nhi u a i m khác. Không c i qua con ng n i t A n B n u con ng ang trong th i gian nâng c p. H th ng giao thông c a thành ph b ng ng tr n u t n t i hai a i m A và B mà không th i c t A n B.

Yêu c u: Cho bi t m ng l i giao thông c a thành ph ND có n a i m và m con ng n i tr c t i p gi a hai a i m. Hãy xác nh s l ng s các con ng mà khi nâng c p thì h th ng giao thông c a thành ph b ng ng tr (n g i n ta coi nh trong m t n v th i gian ch có không quá m t con ng c t i n hành nâng c p).

D li u vào: T t p v n b n SD.INP, có c u trúc:

- Dòng 1: ch a 2 s n và m u nguyên d ng (n 100000; m 200000).



- Trong m dòng ti p theo, m i dòng ch a hai s u và v; th h i n có con
ng n i tr c ti p t a i m u n a i m v.

D li u ra: a ra t p v n b n SD.OUT, ch a duy nh t m t s s tìm c theo
yêu c u.

Ví d v d li u vào /ra:

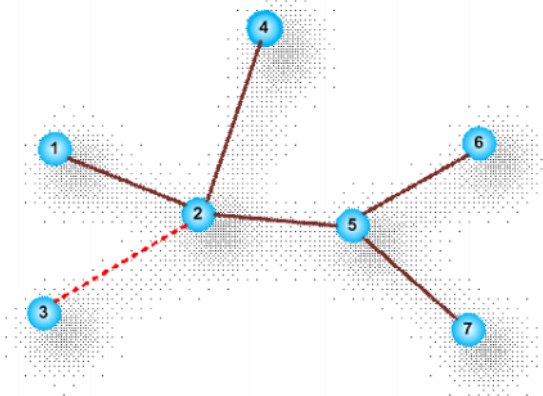
SD . INP		SD . OUT
5	5	2
1	2	
1	3	
1	4	
2	3	
4	5	

H ng d n: m s c nh c u c a th (cài t thu t toán nh trên)

TÀU I N

R ng ô ng là m t thành ph không l n nh ng có m t m ng giao thông công
c ng b ng tàu i n r t thu n ti n và h p lý. T hai b n b t k có th i t i
nhau b ng tàu i n và ch có m t cách i duy nh t. Nh v y m ng tàu i n
t o thành m t cây mà nút là các b n và c nh là tuy n ng tàu.

Ban u, gi a hai b n b t k có ít nh t m t tuy n tàu i n ch y. Nh ng v i
s phát tri n c a thành ph và các lo i ph ng ti n giao thông công c ng
khác m t s tuy n b h y b vì g n
nh không còn hành khách. i u
này d n n vi c m t s o n ng
s t không có tàu nào ch y
qua. Chính quy n thành ph quy t
nh tháo d nh ng o n ng
này.



Yêu c u: Cho s nguyên n ($2 \leq n \leq 100\,000$) – s b n . Các b n c

ánh s t 1 n n . Cho $(n-1)$ c p s b_i, e_i xác nh các c p b n có
ng tàu n i tr c ti p. Cho m – s tuy n ang ho t ng ($0 \leq m \leq 100\,000$)
và m c p s (x, y) , m i c p s xác nh m t tuy n i t x t i y theo ng
ng n nh t. Hãy xác nh s các o n ng c n tháo d .

D li u: Vào t file v n b n TRAM.INP:

- Dòng u tiên ch a s nguyên n ,



- Dòng th i trong $n-1$ dòng sau ch a 2 s nguyên b_i và e_i ,
- Dòng ti p theo ch a s nguyên m ,
- M i dòng trong m dòng sau ch a 2 s nguyên x và y .

K t qu : a ra file v n b n TRAM.OUT m t s nguyên – s các o n ng c n tháo d .

Ví d :

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

H ng d n: (bài này có th dùng thu t toán LCA-tìm cha chung g n nh t, ta s không bàn n)

- th ban u có n nh, $n-1$ c nh (th liên thông, không có chu trình).
- m c p (x,y), m i c p (x,y) thêm c nh (x,y) vào th ban u, ta s c m t chu trình.

=> Trên th này ta m s c nh c u chính là s tuy n ng ph i b .

Chú ý: m s c nh c u trên a th

b. Li t kê các nh kh p:

- N u t nhánh DFS g c v không có cung nào ng c lên phía trên u, t c là n u b u i thì t v không có cách nào lên c các t i n b i c a u. i u này ch ra r ng n u u không ph i là nút g c c a m t cây DFS thì u là kh p. V y n u u không ph i là nút g c c a m t cây DFS thì u là kh p n u và ch n u $Low[v] \geq Number[u]$.
- Thu t toán li t kê các nh kh p c a th : (ng d ng c ch tô màu cho các nh c a th : m i nh c tr ng b i 3 màu: ch a th m (màu White); ang th m (màu Gray); th m xong (màu Black).



Chú ý: g c c a cây DFS thì là kh p n u và ch n u có t hai nhánh con tr lên.

- Cài t (gi ng cài t tìm c nh c u, ta ch s a i u ki n)

```

procedure DFS(u:longint);
var  pq:Graph;
     v:longint;
begin
    Time:=Time+1;
    d[u]:=Time;
    l[u]:=maxlongint;
    Color[u]:=gray;
    pq:=G[u];
    while pq<>nil do
        begin
            v:=pq^.v;
            if p[u]<>v then
                begin
                    if color[v]=white then
                        begin
                            p[v]:=u;
                            con[u]:=con[u]+1;// m s con c a u
                            DFS(v);
                            l[u]:=min(l[u],l[v]);
                            if(p[u]<>-1)and(l[v]>=d[u])and      not
                                dd[u] then
                                    begin
                                        khop:=khop+1;
                                        dd[u]:=true;//u là
                                        kh p
                                    end;
                                end
                            else
                                if color[v]=gray then
                                    l[u]:=min(l[u],d[v]);
                                end;
                            pq:=pq^.link;

```



```
end;
if (p[u]=-1) and (con[u]>1) then
    khop:=khop+1;//n u nút cha có hai con tr
lên
    color[u]:=black;
end;
- Ví d : mã bài Graph_ tìm kh p và c u trên Spoj
Xét n th vô h ng  $G = (V, E)$  có  $n(1 \leq n \leq 10000)$  nh và
 $m(1 \leq m \leq 50000)$  c nh. Ng i ta nh ngh a m t nh g i là kh p n u nh xoá
nh ó s làm t ng s thành ph n liên thông c a th . T ng t nh v y, m t
c nh c g i là c u n u xoá c nh ó s làm t ng s thành ph n liên thông c a
th .
V n tra là c n ph i m t t c các kh p và c u c a th G.
```

Input

+Dòng u: ch a hai s t nhiên n, m .
 +M dòng sau m i dòng ch a m t c p s (u, v) ($u < v, 1 \leq u \leq n, 1 \leq v \leq n$) mô t
 m t c nh c a G.

Output

G m m t dòng duy nh t ghi hai s , s th nh t là s kh p, s th hai là s c u
 c a G

Example

Input	Output
10 12 1 10 10 2 10 3 2 4 4 5 5 2 3 6 6 7 7 3 7 8 8 9 9 7	4 3



- Ch ng trình

```
uses      math;
const
    fi='Graph_.inp';
    fo='graph_.out';
type
    Graph=^Node;
    Node=record
        v:longint;
        link:Graph;
    end;
    Tcolor=(white,gray,black);
var
    n,m,i,cau,khop,u,v,time:longint;
    G:array[0..10000+5] of Graph;
    dd:array[0..10000+5] of boolean;
    con,p,d,l:array[0..10000+5] of longint;
    color:array[0..10000+5] of TColor;
procedure add(u,v:longint);
var    q:Graph;
begin
    new(q);
    q^.v:=v;
    q^.link:=g[u];
    g[u]:=q;
end;
procedure DFS(u:longint);
var    q:Graph;
        v:longint;
begin
    Time:=Time+1;
    d[u]:=Time;
    l[u]:=maxlongint;
    Color[u]:=gray;
    q:=G[u];
    while q<>nil do
        begin
            v:=q^.v;
            if p[u]<>v then
                begin
```



```
        if color[v]=white then
            begin
                p[v]:=u;
                con[u]:=con[u]+1;
                DFS(v);
                l[u]:=min(l[u],l[v]);
                if (p[u]<>-1)and(l[v]>=d[u]) and
not dd[u] then
                    begin
                        khop:=khop+1;
                        dd[u]:=true;
                    end;
                end
                else
                    if color[v]=gray then
                        l[u]:=min(l[u],d[v]);
                    end;
                q:=q^.link;
            end;
        if (p[u]<>-1) and (l[u]>=d[u]) then cau:=cau+1;
        if (p[u]=-1) and (con[u]>1) then begin khop:=khop+1;
end;
        color[u]:=black;
end;
begin
    assign(input,fi);
    reset(input);
    assign(output,fo);
    rewrite(output);
    readln(n,m);
    for i:=1 to n do
        begin
            G[i]:=nil;
            color[i]:=white;
            dd[i]:=false;
            con[i]:=0;
        end;
    for i:=1 to m do
        begin
            readln(u,v);
```



```
add(u,v);
add(v,u);

end;

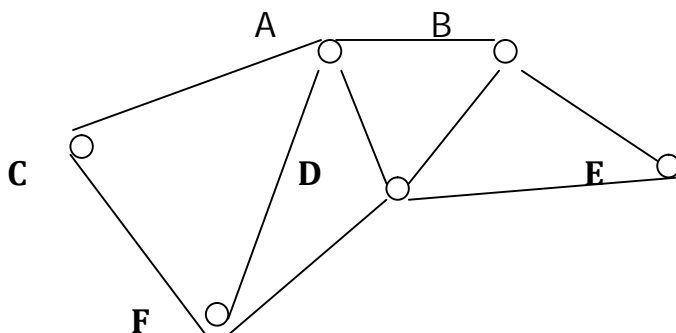
cau:=0;
khop:=0;
time:=0;
for i:=1 to n do
    if color[i]=white then
        begin
            p[i]:=-1;
            DFS(i);
        end;
    write(khop,' ',cau);
    close(input);
    close(output);
end.
```

III. K t l u n

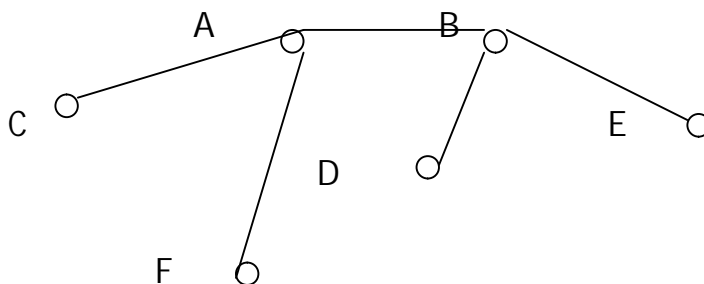
Hì u rõ c c ch ho t ng thu t toán tìm ki m theo chi u sâu (DFS) b ng quy cho ta cách cài t r t ng n g n, rõ ràng. Nh ng c i ti n nh trong thu t toán có th em lai nhi u i u thú v, gi i quy t c nhi u l p bài toán khác nhau. Trong ph m vi chuyên này tôi không th trình bày h t c nh ng ng d ng c a DFS, nh ng ph n nào cho th y c t m quan tr ng c a DFS.

Tài li u tham kh o:

- Tài li u chuyên tin quy n 1 – H S àm (ch biên)
- Toán r i r c – Nguy n c Ngh a – Nguy n Tô Thành
- Website <http://vn.spoj.com>



Rõ ràng là phải cắt đứt trên ít nhất n cạnh con đường đó là (A,C) ; (A,F) ; (A,B) ; (B,D) ; (B,E) . Đây là số biên độ nhỏ nhất cắt các con đường đó:



Sau đây cho ta hình minh họa cây, gồm tất cả các đỉnh của đồ thị và tất cả các cạnh của đồ thị. Minh họa có thể có hình minh họa cây khung.

Ta bài toán trên minh họa ra hai vấn đề:

Thứ nhất, tìm cây khung của nó.

Thứ hai, nếu muốn tìm các đỉnh của đồ thị có thể có một đồ thị thì hãy tìm cây khung có trọng số nhỏ nhất.

Trong khuôn khổ bài báo này, chúng tôi xin trình bày cách giải quyết của các vấn đề nêu trên.

1. CÂY KHUNG CỦA ĐỒ THỊ

1.1. Định nghĩa cây

Cây: là một đồ thị liên thông, vô hướng, liên thông và không có chu trình.

Rừng: là một đồ thị liên thông, vô hướng và không có chu trình.

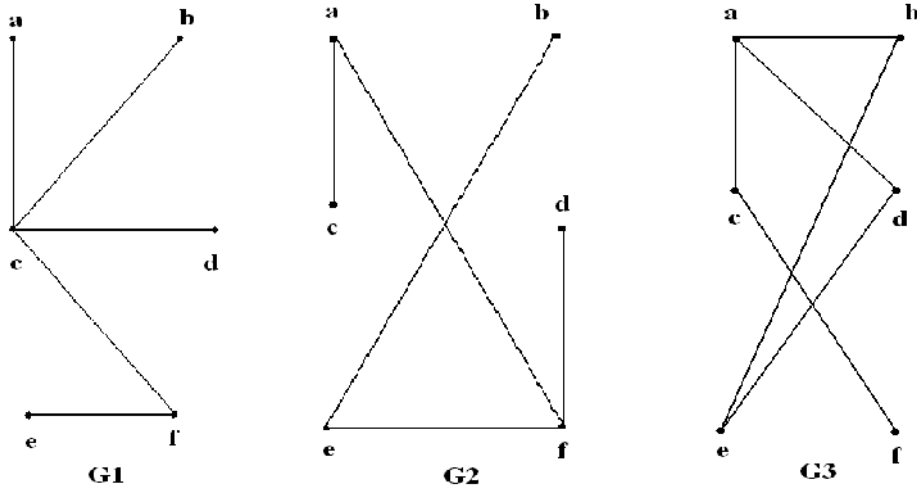
Đỉnh: là một đồ thị $G=(X,U)$ liên thông, có đỉnh là một tập hợp X và nó có ít nhất hai đỉnh và thỏa mãn 3 điều kiện sau:

- Mọi đỉnh khác x_1 đều là đỉnh cuối của một đường duy nhất.



- nh x_1 không là nh cụ i c a b t kì cung nào.
- th không có chu trình.

Ví d : Quan sát các th d i ây:



D a vào nh ngh a c a cây ta th y: G1, G2 là cây; G3 không là cây do có chu trình.

1.2. Tính ch t c a cây

Đ nh lý 1

N u T là th vô h ng, n nh ($n > 1$) và T có m t trong sáu tính ch t sau thì T là cây. M i tính ch t là m t m nh . Khi ó, các m nh sau là t ng ng:

- (1) T là cây.
- (2) T không có chu trình và có $(n-1)$ c nh.
- (3) T có $(n-1)$ c nh và liên thông.
- (4) T liên thông và m i c nh c a T u là c nh c t (c u).
- (5) Hai nh b t kì c a T c n i v i nhau b ng úng m t ng i n.
- (6) T không ch a chu trình nh ng n u thêm m t c nh b t kì vào T thì ta c thêm úng m t chu trình.

Ch ng minh đ nh lý:

(1) (2): T là cây T không ch a chu trình và có $(n-1)$ c nh.

- Hi n nhiên T không ch a chu trình (do T là cây).



- Tach c n ch ng minh T có (n-1) c nh.
- Xét T_n là cây có n nh. Ta s ch ng minh quy n p theo n:
 - $n = 2$. Cây có 2 nh thì có 1 c nh. úng.
 - Gi s cây có k nh thì có (k-1) c nh.
 - Xét T_{k+1} là cây có (k+1) nh. D th y trong cây này luôn t n t i ít nh t m t nh treo.
 - Lo i nh treo này (cùng v i c nh n i) ra kh i T_{k+1} ta c th T' có k nh. D th y, T' v n liên thông và không có chu trình (do T_{k+1} không có chu trình).
 - Suy ra T' là cây. Theo gi thi t quy n p, T' có k nh thì s có (k-1) c nh. V y T_{k+1} có k c nh (pcm).

(2) (3): T không có chu trình và có (n-1) c nh T liên thông và có (n-1) c nh.

- H i n nhiên T có (n-1) c nh (theo gi thi t).
- Tach c n ch ng minh T liên thông.
- Gi s T có k thành ph n liên thông v i s nh l n l t là n_1, n_2, \dots, n_k .
- Khi ó m i thành ph n liên thông c a T s là m t cây và s có s c nh l n l t là $n_1-1, n_2-1, \dots, n_k-1$.
- Suy ra, s c nh c a T s là: $n_1-1 + n_2-1 + \dots + n_k-1 = n-k$.
- Theo gi thi t, s c nh c a cây là (n-1). T ó suy ra $k=1$ hay T ch có m t thành ph n liên thông. Suy ra, T liên thông (pcm).

(3) (4): T có (n-1) c nh và liên thông T liên thông và m i c nh c a T đ u là c nh c t (c u).

- H i n nhiên T liên thông (theo gi thi t).
- Tach c n ch ng minh m i c nh c a T u là c nh c t.
- Xét (u,v) là c nh b t kì c a T. N u b (u,v) ra kh i T thì ta s c th T' có n nh và (n-2) c nh.



- Ta ã ch ng minh c th có n nh và $(n-2)$ c nh thì không th liên thông.

- V y n u b c nh (u,v) ra thì s làm m t tình liên thông c a th . Suy ra (u,v) là c nh c t (pcm).

(4) (5): *T liên thông và m i c nh c a T đ u là c nh c t (c u)*

Hai đ nh b t kì c a T đ c n i v i nhau b ng đúng m t đ ng đi đ n.

- Xét u,v là 2 nh b t kì trong T .
- Do T liên thông nên luôn t n t i ng i gi a u,v . Ta s ch ng minh ng i này là duy nh t.
- Gi s có hai ng i n khác nhau gi a u và v . Khi ó hai ng i này s t o thành chu trình.
- Suy ra các c nh trên chu trình này s không th là các c nh c t c.
- V y gi a u và v s ch c t n t i úng m t ng i n (pcm).

(5) (6): *Hai đ nh b t kì c a T đ c n i v i nhau b ng đúng m t*

đ ng đi đ n T không ch a chu trình nh ng n u thêm m t c nh b t kì vào T thì ta đ c thêm đúng m t chu trình.

- T không th có chu trình, vì n u có chu trình thì 2 nh trên chu trình này s có hai ng i n khác nhau Mâu thu n v i gi thi t.
- Gi s ta thêm vào T c nh (u,v) b t kì (tr c ó không có c nh này trong T).
- Khi ó c nh này cùng v i ng i duy nh t gi a u và v trong T s t o thành m t chu trình (vì n u t o hai chu trình thì ch ng t tr c ó có hai ng i khác nhau gi a u và v Mâu thu n v i gi thi t).

(6) (1): *T không ch a chu trình nh ng n u thêm m t c nh b t kì vào T thì ta đ c thêm đúng m t chu trình T là cây.*

- H i n nhiên T không ch a chu trình.
- Gi s T không liên thông. Khi ó T s có nhi u h n m t thành ph n liên thông.



- Suy ra, n u thêm vào m t c nh b t kì g i a hai nh thu c hai thành ph n liên thông khác nhau s không t o thêm m t chu trình nào. Mâu thu n v i g i thi t.
- V y T ph i liên thông T là cây (pcm).

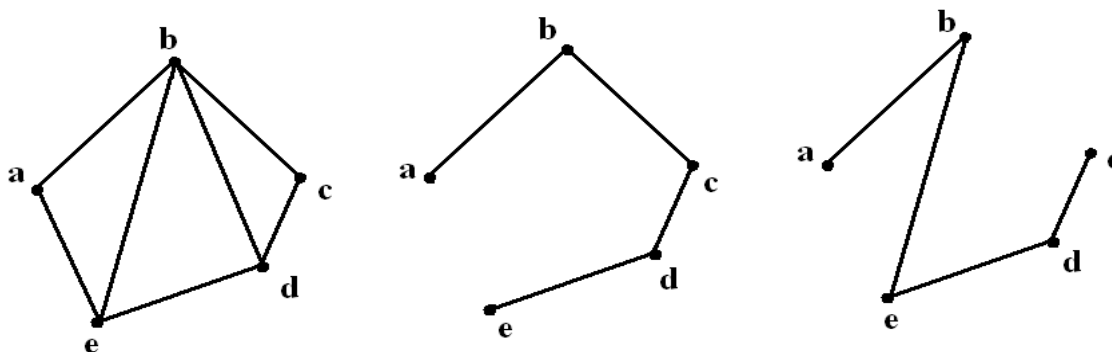
Đ nh lí 2

M t b i n u thay các cung b ng c nh thì thành cây.

1.3. Cây khung c a đ th

Đ nh nghĩa cây khung: Cho đ th vô h ng $G=(X,E)$ liên thông, có n nh ($n>1$). M i đ th b ph n c a G n u là cây thì c g i là cây khung c a đ th G (ho c cây bao trùm).

Ví d :



đ th và các cây khung c a nó.

Đ nh lí: M i đ th vô h ng có s đ nh $n>1$ liên thông khi và ch khi nó có cây khung.

Ch ng minh đ nh lí:

- N u G có ch a cây khung thì do tính ch t c a cây khung là liên thông và cây khung ch a t t c các nh c a G. Suy ra các nh c a G luôn c n i v i nhau hay G liên thông.
- Xét G liên thông. G i s trong G còn t n t i chu trình, xoá b t m t c nh trong chu trình này, khi ó đ th v n còn liên thông. N u v n còn chu trình thì l p l i b c trên. C th cho n khi không còn chu trình n a. Khi ó ta s c cây khung.

1.4. Thu t toán tìm cây khung

1.4.1. Bài toán



Cho th G liên thông, vô h ng, hãy tìm m t cây khung c a nó.

- D li u: s nh và danh sách các c nh c a th .
- K t qu : các c nh c a cây khung

CK.inp	CK.out
6	1 6
1 6	2 3
2 3	4 5
4 5	2 5
2 5	1 5
1 5	
1 2	
6 5	
5 3	

1.4.2.Thu t toán

Thu t toán 1:

Ý t ng: Duy t và th m các nh, m i nh m t l n. Vì th liên thông nên th m n nh (cùng lúc v i ã qua n-1 c nh, ta c cây khung). Có th dùng thu t toán DFS ho c BFS th m các nh.

Cài đ t:

```
program tim_caykhung_bang_DFS;
const   fi='CayKhung.inp';
        fo='CKhung.DFS.out';
        MN=1000;
var     A:array[1..MN,1..MN] of longint;
        vs:array[1..MN] of boolean;
        n,m:integer;

procedure nhap;
var i,j:integer;
begin
    assign(input,fi);
    reset(input);
    readln(n,m);
```



```
while not seekeof(input) do //doc
cac canh cua do thi
begin
    readln(i,j);
    a[i,j]:=1;
    A[j,i]:=1;
end;
close(input);
end;
//-----
procedure DFS_VS(i:integer);
var j:integer;
begin
    vs[i]:=true;
    for j:=1 to n do
        if not vs[j] and (a[i,j]=1) then
            begin
                writeln(i,' ',j);
                DFS_VS(j);
            end;
    end;
end;
//-----
Begin
    nhap;
    fillchar(vs, sizeof(vs),false);
    assign(output,fo); rewrite(output);
    DFS_VS(1);
    close(output);
end.
```

Thu t toán 2: H p nh t d n các vùng liên thông

Ý t ng: M i l n h p nh t hai vùng liên thông khác nhau b ng m t c nh n i hai vùng này thì n p c nh ó vào cây khung ang hình thành. Quá trình ch m d t khi n p (n-1) c nh.

Th c hi n c th :



B c 1: Coi m i nh thu c m t vùng có mã vùng là $v[i]=i$, s c nh ã n p vào cây khung là $sl=0$.

B c 2: Duy t t t c các c nh c a th :

- N u $sl=n-1$ thì d ng vòng l p duy t.
- N u c nh (i,j) có nh i và j khác mã vùng ($v[i] \neq v[j]$) thì:
 - N u $v[i]<v[j]$: t t c các nh cùng mã vùng v i j c gán l i mã vùng là $v[i]$, n p vào cây khung c nh (i,j) , t ng bi n sl m t n v .
 - N u $v[i]>v[j]$: t t c các nh cùng mã vùng v i i c gán l i mã vùng là $v[j]$, n p vào cây khung c nh (i,j) , t ng bi n sl m t n v .

Cài đ t:

```

program CayKhung;
const   fi='CayKhung.inp';
        fo='CK.out';
var     b,dau,cuoi:array[1..10000] of longint;
        i,j,k,n,t,sc:longint;
        f:text;
procedure nhap;
begin
    assign(f,fi);
    reset(f);
    readln(f,n);
    for i:=1 to n do
        b[i]:=i;
    sc :=0;
    while not eof(f) do {doc cac canh
cua do thi}
        begin
            readln(f,i,j);
            if b[i] <> b[j] then {khac ma vung
lien thong}
                begin
                    inc(sc); {tang so
canh}

```



```
        dau[sc] := i;           {nap them mot
canh vao 2 mang dau va cuoi}
        cuoi[sc] := j;
        if b[i]<b[j] then
            for t:=1 to n do {hop nhat 2
vung lien thong}
                if      b[t]=b[j]      then
b[t]:=b[i];
            if b[i]>b[j] then
                for t:=1 to n do
                    if      b[t]=b[i]      then
b[t]:=b[j];
            end;
        if sc = n-1 then break;      {nap du n-1
canh thi dung lai}
        end;
        close(f);
end;
procedure xuat;
begin
    assign(f,fo);
    rewrite(f);
    for i := 1 to n -1 do
        writeln(f,dau[i], ' ',cuoi[i]);
    close(f);
end;
begin
    nhap;
    xuat;
end.
```

Thu t toán 3: H p nh t d n các cây

Ý t ng: M i l n h p nh t hai cây có g c khác nhau b ng m t c nh c a th (n i hai nh thu c hai cây này) thì c nh ó c xác nh n là m t c nh c a cây khung ang hình thành. Quá trình k t thúc khi n p n-1 c nh c a cây khung.



Cài đặt:

```
program CayKhung; {su dung thuat toan hop nhat hai
cay}
const    fi='Caykhung.inp';
          fo='CKhung.out';
          MN=5000;

var      cha,dau,cuoi:array[1..MN] of integer;
          n,m,socanh:longint;
//-----
function root(x:integer):integer;      //tim goc cay
chua dinh x
var      i:integer;
begin
    i:=x;
    while cha[i]>0 do i:=cha[i];
    exit(i);
end;
//-----
procedure Union(x,y:integer);          // hop nhat hai
cay oc x, goc y
var      temp:integer;
begin
    temp:=cha[x]+cha[y];
    if cha[x]>cha[y] then                // cay chua
dinh x co it nut hon
        begin
            cha[x]:=y;                  //tam coi y la
cha cua x trong cay hop nhat
            cha[y]:=temp;               //goc moi cua 2
cay la y cay chua dinh x co it nut hon
        end
    else
        begin                          //cay chua dinh
y co it nut hon
            cha[y]:=x;

```



```
        cha[x]:=temp;
    end;
end;
//-----

procedure nhap_taocay;
var    i, x, y,r1,r2:longint;
begin
    assign(input,fi);
    reset(input);
    readln(n);
    for i:=1 to n do cha[i]:=-1;           // moi dinh la
cay co goc la chinh no
    socanh :=0;
    while not seekeof(input) do           //doc cac canh
cua do thi
        begin
            if socanh=n-1 then exit;       //la cay khung,
ket thuc
            readln(x,y);
            r1:=root(x);
            r2:=root(y);
            if r1 <> r2 then                 //hai cay co
goc khac nha
                begin
                    inc(socanh);           //tang so canh
                    dau[socanh] := x;      //nap them mot
canh vao 2 mang dau va cuoi
                    cuoi[socanh] := y;
                    union(r1,r2);         //hop nhat hai
cay
                end;
            end;
        end;
    close(input);
end;
procedure xuat;
```



```
var i:integer;  
begin  
    assign(output,fo);  
    rewrite(output);  
    writeln(socanh);  
    for i := 1 to n -1 do  
        writeln(dau[i], ' ',cuoi[i]);  
    close(output);  
end;  
begin  
    nhap_taocay;  
    xuất;  
end.
```

1.5. Thu t toán tìm cây khung ng n nh t

1.5.1. Bài toán

Đ nh nghĩa: Cho th G vô h ng, liên thông và có tr ng s không âm. Cây khung ng n nh t c a th G là cây khung có t ng tr ng s trên các c nh c a nó nh nh t (g i là tr ng s c a cây khung ng n nh t).

Bài toán: Cho th G vô h ng, liên thông và không có tr ng s âm. Tìm cây khung ng n nh t c a th G.

D li u: s nh, danh sách các c nh kèm theo tr ng s c a c nh (m i dòng mô t c nh g m 3 s i, j, l có ngh a c nh n i nh i và j thì có tr ng s là l).

K t qu : các c nh c a cây khung ng n nh t và tr ng s c a cây khung ng n nh t.

Ck.inp	Ck.out
5	Cay khung la:
1 2 2	(1,4)
1 3 2	(3,4)
1 4 1	(1,2)
1 5 4	(2,5)
2 3 5	Tong trong so : 8
2 4 3	
2 5 3	
3 4 2	
3 5 4	
4 5 8	



gi i bài toán cây khung ng n nh t có 2 thu t toán thông d ng: Kruskal và Prim

1.5.2. Thu t toán Kruskal

Ý t ng: N p d n các c nh ng n nh t và cây khung n u c nh y không t o thành chu trình v i các c nh ã n p.

Thu t toán:

- S p x p các c nh t ng d n (th ng dùng Quicksort)
- Lân l t k t n p các c nh có tr ng s nh nh t trong các c nh còn l i vào cây n u sau khi k t n p c nh này không t o thành chu trình trong cây. th c hi n yêu c u này, ta có th s d ng thu t toán h p nh t các vùng liên thông tr n. Quá trình này d ng khi k t n p c n-1 c nh vào cây.

Cài đ t:

```
program Kruskal;  
const fi='ck.inp';  
      fo='ck.out';  
type canh = record  
      d,c,l : longint;  
end;  
var b:array[1..10000] of longint;  
    a,ck : array[1..10000] of canh;  
    i,j,k,n,m,t,sc,sum:longint;  
    f:text;  
Procedure QuickSort(dau, cuoi : longint);  
var x, L, R : longint;  
    tmp : canh;  
begin  
    x := a[(dau+cuoi) div 2].l; L := dau; R :=  
    cuoi;  
    repeat  
        while a[L].l < x do inc(L);  
        while a[R].l > x do dec(R);  
        if L <= R then  
            begin
```



```
tmp := a[L];a[L] := a[R]; a[R]
:= tmp;
inc(L);dec(R);
end;
until L > R;
if R > dau then QuickSort(dau,R);
if L < cuoi then QuickSort(L,cuoi);
end;
procedure nhap;
begin
    assign(f,fi); reset(f);
    readln(f,n);
    m := 0;
    while not eof(f) do {doc cac canh cua do
thi}
        begin
            inc(m);
            readln(f,a[m].d,a[m].c,a[m].l);
        end;
    close(f);
end;
procedure xuli;
begin
    QuickSort(1,m);
    for i:=1 to n do
        b[i]:=i;
    sc :=0;sum := 0;
    for k := 1 to m do
        begin
            i := a[k].d;j:= a[k].c;
            if b[i] <> b[j] then {khac ma vung lien
thong}
                begin
                    inc(sc);          {tang so canh}
                    ck[sc] := a[k];    {them canh vao cay
khung}
```



```
sum := sum + a[k].l;  
if b[i]<b[j] then  
    for t:=1 to n do {hop nhat 2 vung  
        lien thong}  
        if b[t]=b[j] then b[t]:=b[i];  
    if b[i]>b[j] then  
        for t:=1 to n do  
            if b[t]=b[i] then b[t]:=b[j];  
    end;  
    if sc = n-1 then break; {nap du n-1 canh  
        thi dung lai}  
    end;  
end;  
procedure xuat;  
begin  
    assign(f,fo);  
    rewrite(f);  
    writeln(f,'Cay khung la: ');  
    for i := 1 to n -1 do  
        writeln(f,  
            (' ,ck[i].d,',',ck[i].c,')');  
    writeln(f,'Tong trong so : ',sum);  
    close(f);  
end;  
begin  
    nhap;  
    xuli;  
    xuat;  
end.
```

1.5.3. Thu t toán Prim

Ý t ng: N p d n t p cách nh vào cây khung. M i l n ch n m t nh ch a n plà nh k và g n các nh ã n p nh t.

Thu t toán:

B c 1: N p m t nh u tiên vào cây khung (th ng là nh 1)



B c 2: L n l t n p n-1 nh còn l i (t ng ng v i n-1 c nh) vào cây khung b ng cách: m i l n ch n m t c nh có tr ng s nh nh t mà m t u c a c nh ã thu c cây, u kia ch a thu c cây (ngh a là ch n m t nhg n các nh ã n p nh t)

Cài t

program Prim;

const max=100;

f1='ck.inp';

f2='ck.out';

var a: array[1..max,1..max] of integer;

d1,d2,d:array[1..max] of integer;

n: integer;

procedure nhap;

var g:text;

i,j,x:integer;

begin

assign(g,f1); reset(g);

readln(g,n);

while not seekeof(g) do

begin

readln(g,i,j,x);

a[i,j]:=x; a[j,i]:=x;

end;

close(g);

end;

procedure timcanh(var i,j:integer);

{Tim canh i, j ngan nhât}

var x,y,min:integer;

begin

min:=maxint;

for x:=1 to n do

if d[x]=1 then

for y:=1 to n do

if d[y]=0 then



```
        if (a[x,y]>0) and (a[x,y]<min)
then
        begin
            i:=x;
            j:=y;
            min:=a[x,y];
        end;
end;
procedure prim;
var i,j,k:integer;
    begin
        for i:=1 to n do d[i]:=0;
        d[1]:=1;
        for k:=1 to n-1 do
            begin
                timcanh(i,j);
                d[j]:=1;
                d1[k]:=i;
                d2[k]:=j;
            end;
        end;
procedure ghi;
var g:text; i,tong: integer;
    begin
        assign(g,f2);      rewrite(g);
        tong:=0;
        writeln(g,'Cay khung la : ');
        for i:=1 to n-1 do
            begin
                writeln(g,d1[i],' ',d2[i]);
                tong:=tong+a[d1[i],d2[i]];
            end;
        writeln(g,'Tong trong so: ',tong);
        close(G);
    end;
begin
    nhap; prim; ghi;
```




end.

2. M T S BÀI TOÁN NG D NG

Bài toán 1. M ng rút g n

M t h th ng g m N máy tính c n i thành m t m ng có M kênh n i, m i kênh n i hai máy tính trong m ng, gi a hai máy tính có không quá m t kênh n i. Các máy tính c ánh s t 1 n N, các kênh n i ánh s t 1 n M. Vì c truy n tin tr c ti p có th th c hi n c i v i hai máy có kênh n i. Các kênh n i trong m ng c chia thành ba lo i 1, 2 và 3. Ta nói gi a hai máy a và b trong m ng có ng truy n tin lo i k ($k=1$ ho c $k=2$) n u tìm c dãy các máy ($a=v_1, v_2, \dots, v_p=b$) tho mãn i u ki n: gi a hai máy v_i và v_{i+1} ho c có kênh n i lo i k ho c có kênh n i lo i 3 ($i=1, 2, \dots, p-1$).

Yêu c u: C n tìm cách lo i b kh i m ng m t s nhi u nh t kênh n i nh ng v n m b o luôn tìm c c ng truy n lo i 1 l n ng truy n tin lo i 2 gi a hai máy b t k trong m ng. **D li u** vào t t p v n b n MRG.INP nh sau:

- Dòng u tiên ch a hai s N, M ($N \leq 500$); $M \leq 10000$).
- Dòng th i trong M dòng ti p theo ch a ba s nguyên d ng u, v, s cho bi t kênh th i n i hai máy u và v thu c lo i s.

K t qu ghi ra t p v n b n MRG.OUT g m:

- Dòng u tiên ghi s r là s kênh c n lo i b .
- N u $r=-1$ thì có ngh a là trong m ng ã cho t n t i hai máy không có ng truy n lo i 1 ho c lo i 2.
- n u $r=0$ có ngh a là m ng có ng truy n tho mãn nh ng s kênh lo i b b ng 0.
- N u $r>0$ thì r dòng ti p theo, m i dòng ghi ch s c a m t kênh c n lo i b .



Các s trên cùng m t dòng c a các t p d li u và t p k t qu cách nhau ít nh t m t d u cách.

Ví d :

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

Cách gi i

+ Xây d ng th vô h ng v i m i nh là m t máy tính, có N nh. M i c nh là m t kênh trong M kênh, có M c nh; tr ng s trên c nh là lo i kênh (1, 2, 3).

+ Dùng thu t toán h p nh t d n các cây (Thu t toán 3) tìm cây khung. C th nh sau:

- Vì ng i lo i 1 và 2 theo yêu c u ph i ch a kênh lo i 3 nên ta tìm r ng cây ch g m nh ng c nh lo i 3, g i là (R3).
- N u r ng cây ó là cây khung (t c là có n-1 c nh) thì bài toán có nghi m và lo i b t t c các c nh lo i 1 và lo i 2.
- N u r ng cây ó ch a là cây khung thì ph i xem xét b sung c nh lo i 1 ho c lo i 2 vào r ng cây ó m ng có ng truy n lo i 1 ho c lo i 2. Ti n hành các vi c sau:

1. Cùng v i R3, xét thêm các c nh lo i 1, th c hi n l i thu t toán 3 xem có t o thành cây khung (ch g m c nh lo i 1 và 3) không. N u không là cây khung thì vô nghi m ($r=-1$); n u có thì th c hi n ti p b c 2:
2. Cùng v i R3, xét thêm các c nh lo i 2, th c hi n ti p thu t toán 3 xem có t o thành cây khung (ch g m c nh lo i 3 và 2) không. N u không thì vô nghi m ($r=-1$); N u có cây khung thì



bài toán có nghi m, các c nh c n lo i b là các c nh lo i 1 và 2 không thu c cây khung tìm th y trên. th c hi n vi c này c n ánh d u các c nh ã c n p vào cây khung.

Văn b n ch ng trình

```
program mang_rut_gon;
const   fi='MRG.IN2';
        fo='MRG.OUT';
        MN=500;
        MM=10000;
type    canh=record   u,v:integer; w:shortint end;
var      m,n:integer;
        socanh, lsc: integer;           //so canh,
luu so canh
        l, ll:array[1..MN] of integer; //nhan cua
dinh, luu nhan ddinh
        ds:array[1..MM] of canh;       //danh sach
canh
        caykhung:boolean;              //co la cay
khung hay khong
//-----
procedure readf;
var      i:integer;
begin
    assign(input,fi); reset(input);
    readln(n,m);
    for i:=1 to m do
        with ds[i] do readln(u,v,w);
    close(input);
    for i:=1 to n do l[i]:=-1;           // moi cay
co goc la chinh no

end;
//-----
function root(u:integer):integer;       //tra
ve goc cay chua u
```



```
begin
  while l[u]>=0 do u:=l[u];
  exit(u);
end;
//-----
procedure union(r1,r2:integer);           //hop nhat
hai cay co goc la r1, r2
var    x:integer;
begin
  x:=l[r1]+l[r2];                         //nhan cua
goc cay hop nhat
  if l[r1]>l[r2] then
    begin
      l[r1]:=r2;
      l[r2]:=x
    end
  else
    begin
      l[r2]:=r1;
      l[r1]:=x;
    end;
end;
//-----
function KRUSKAL(k:integer):boolean;     //co la cay
khung khi them canh loai k khong
var    i, r1,r2:integer;
begin
  for i:=1 to m do
    with ds[i] do if w=k then
      begin
        r1:=root(u);                     //goc cua cay
chua dinh u
        r2:=root(v);                     //goc cay
chua dinh v
        if r1<>r2 then
          begin
```



```
w:=-w; // danh dau
da canh da nap vao cay
    inc(socanh);
    if socanh=n-1 then exit(true); //la
cay khung
        union(r1,r2); // chua la
cay thi hop nhat hai cay
    end;
end;
exit(false);
end;
//-----
procedure writef;
var i:integer;
begin
    assign(output,fo); rewrite(output);
    if not caykhung then writeln(-1) //khong co
duong truyen thoa man
    else
        begin
            writeln(m-n+n-2-lsc);
            for i:=1 to m do
                if ds[i].w>0 then writeln(i);
            end;
        close(output);
    end;
//-----
begin
    readf;
    socanh:=0;
    caykhung:=KRUSKAL(3);
    if not caykhung then
        begin
            lsc:=socanh;
            ll:=1;
            caykhung:=KRUSKAL(1);
```



```
socanh:=lsc;
l:=11;
caykhung:=caykhung and KRUSKAL(2);
end;
writef;
end.
```

Bài toán 2. Mạng giao thông

Theo thị t k , m t m ng giao thông g m N nút có s hi u t 1 n N (N 1000).

Chi phí xây d ng ng hai chi u tr c ti p t nút i n nút j b ng $A[i,j]=A[j,i]$. Hai tuy n ng khác nhau không c t nhau t i các i m không là u nút. Hi n ã xây d ng c K tuy n ng.

Bài toán t ra nh sau: H th ng ng ã xây d ng có b o m s i l i gi a hai nút bất k ch a? N u ch a, hãy ch n m t s tuy n ng c n xây d ng thêm sao cho:

1. Các tuy n ng s xây d ng thêm cùng v i K tuy n ng ã xây d ng b o m s i l i gi a hai nút b t k .
2. T ng kinh phí xây d ng thêm các tuy n ng là ít nh t.

D li u vào t t p v n b n MGT.INP nh sau:

- Dòng u tiên ch a hai s N, K (N 500); M 10000).
- Trong K dòng ti p theo m i ch a hai s nguyên d ng là s hi u hai nút, ó là các tuy n ng ã xây d ng.
- Cu i cùng là N dòng, d ng th i ghi N s $A[i,1], A[i,2], \dots, A[i,N]$.

K t qu ghi ra t p v n b n MGT.OUT g m:

- Dòng u tiên ghi s CP là chi phí xây d ng thêm.
- N u $CP > 0$ thì trong N dòng ti p theo, m i dòng ghi hai s là s hi u hai nút, ó là hai u c a tuy n ng c n xây d ng thêm.

Các s trên cùng m t dòng c a các t p d li u và t p k t qu cách nhau ít nh t m t d u cách.

Ví d :



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

Cách gi i:

- + D a vào m ng giao thông xây d ng th vô h ng, có tr ng s :
 - M i nh c a th là m t nút giao thông (N nh);
 - M i c nh là o n ng tr c ti p n i 2 nút;
 - Tr ng s trên c nh t ng ng v i o n ng ã xây d ng b ng 0; trên c nh ch a xây d ng b ng chi phí xây d ng quăng ng t ng ng.
- + Tìm cây khung ng n nh t trên th . N u tr ng s c a cây b ng 0, có ngh a là K o n ng ã xây d ng ã m b o s i l i g i a hai nút b t k (th ã liên thông). Ng c l i, n u tr ng s khác 0, thì trên cây có nh ng o n ng ch a xây d ng (là nh ng c nh có tr ng s khác 0). ó chính là nh ng o n ng c n xây d ng thêm.

+ V n b n ch ng trình

```

Program          MangGiaoThong;
Const    Fi='MGT.INP' ;
          Fo='MGT.OUT' ;
          nm=100;
Var       f:text;
          n:integer;
          a:array[1..nm,1..nm] of longint;
          tr:array[1..nm] of integer;
          vs:array[1..nm] of boolean;
          res:longint;
  
```



```
Procedure      Nhap;  
var      i,j:integer;  
         k:longint;  
begin  
    assign(f,fi);  
    reset(f);  
    readln(f,n,k);  
    fillchar(a,sizeof(a),255);  
    while k>0 do  
        begin  
            readln(f,i,j);  
            a[i,j]:=0; a[j,i]:=0;  
            dec(k);  
        end;  
    for i:=1 to n do  
        for j:=1 to n do  
            begin  
                read(f,k);  
                if a[i,j]=-1 then a[i,j]:=k;  
            end;  
    close(f);  
end;  
  
Procedure      Prim;  
var      i,j,sc:integer;  
         min:longint;  
begin  
    for i:=1 to n do tr[i]:=1;  
    fillchar(vs,sizeof(vs),false);  
    vs[1]:=true; res:=0;  
    for sc:=1 to n-1 do  
        begin  
            min:=High(longint);  
            for i:=1 to n do  
                if not vs[i] and (a[tr[i],i]<min) then
```




```
begin
    min:=a[tr[i],i];
    j:=i;
end;
vs[j]:=true;
res:=res+a[tr[j],j];
for i:=1 to n do
    if not vs[i] and (a[j,i]<a[tr[i],i])
then tr[i]:=j;
    end;
end;

Procedure      Xuat;
var      i:integer;
begin
    assign(f,fo);
    rewrite(f);
    writeln(f,res);
    for i:=1 to n do
        if a[tr[i],i]<>0 then writeln(f,tr[i], '
',i);
    close(f);
end;

Begin
    Nhap;
    Prim;
    Xuat;
End.
```

Bài toán 3. Tìm cây khung dài nhất

Các thuật toán Kruskal và Prim không đòi hỏi việc đưa ra các cấu trúc. Vì vậy ta có thể áp dụng với cây có các cạnh có trọng số dương. Do đó, tìm cây khung dài nhất ta chỉ cần đưa tất cả các trọng số và áp dụng thuật toán trong hai thuật toán trên tìm cây khung nhỏ nhất trên đồ thị mà xây dựng. Cụ thể



cùng ch vì c i d u trong s c a cây khung. Tính úng n c a thu t toán là hi n nhiên vì m ts l n nh t thì d n ns i c a nó ph i nh nh t.

Bài toán 4. Tìm m ng đi n v i s tin c y l n nh t

Bài toán: Cho l i i n có N nút. ng dây n i nút i v i nút j có tin c y là m ts th c $0 < P_{ij} < 1$. tin c y c a toàn b l i i n b ng tích tin c y trên t t c các ng dây. Hãy tìm cây khung v i tin c y l n nh t.

Cách gi i:

- + Xây d ng th vô h ng, có tr ng s v i s nh là s nút c a l i i n, m i c nh (i,j) c a th là o n ng dây n i hai nút i và j. Tr ng s trên c nh (i,j) c gán b ng $-\ln(P_{ij})$.
- + Tìm cây khung nh nh t c a th v a xây d ng b ng m t trong hai thu t toán Kruskal ho c Prim.
- + Tính úng n c a thu t toán c ch ng minh nh tính ch t c a logarit nh sau:

Ta có công th c toán h c: $\ln(x_1.x_2...x_N) = \ln(x_1) + \ln(x_2) + ... + \ln(x_N)$. Vì th nên th c hi n tìm tin c y l n nh t c a m ng i n thay vì tính tích c a các tr ng s trên cây khung T, ta a v tính t ng c a c a các tr ng s m i. Khi ó ta có t ng tr ng s trên cây khung ng n nh t T là m ts âm nh nh t. Khi i d u ó là s l n nh t.

Bài toán 5. Tìm cây khung ng n nh t trên đ th , s d ng câu trúc HEAP.

- + Bi u di n th ban u b i danh sách k v i tr ng s . Khi ó có th cài t th v i s nh r t l n (10000 nh).
- + Áp d ng thu t toán Prim tìm cây khung ng n nh t trên th v a xây d ng, v i thao tác tìm nh g n nh t, t c là c nh liên thu c có tr ng s nh nh t b ng cách s d ng c u trúc HEAP. Vì v y m c dù v i th có s nh r t l n ch ng trình v n áp ng c yeu c u v th i gian.

+ Cài t c th

```
Program CayKhungNhoNhat_Heap;
Const Fi='caykhung.INP';
      Fo='CK_HEAP.INP';
      nm=10000;
      mm=15000;
      vc=10001;
Var f:text;
```



```
n,m,nH:longint;  
ke,t:Array[1..mm*2] of longint;  
index:array[0..nm] of longint;  
minc:array[2..nm] of longint;  
c1,c2,c,info,pos:array[1..mm] of longint;  
res:longint;  
Procedure      Nhap;  
var      i:longint;  
begin  
    assign(f,fi);  
    reset(f);  
    readln(f,n,m);  
    for i:=1 to m do readln(f,c1[i],c2[i],c[i]);  
    close(f);  
end;  
Procedure      TaoKe(x,y,c:longint);  
begin  
    ke[index[x-1]]:=y;  
    t[index[x-1]]:=c;  
    dec(index[x-1]);  
end;  
Procedure      Chuyen;      {Chuyen tu Ds canh -> Ds  
    ke}  
var      i:longint;  
begin  
    fillchar(index,sizeof(index),0);  
    for i:=1 to m do  
        begin  
            inc(index[c1[i]-1]);  
            inc(index[c2[i]-1]);  
        end;  
    for i:=1 to n do index[i]:=index[i-1]+index[i];  
    for i:=1 to m do  
        begin  
            TaoKe(c1[i],c2[i],c[i]);
```



```
TaoKe(c2[i],c1[i],c[i]);
end;

end;
Procedure      Swap(var a,b:longint);
var      tg:longint;
begin
      tg:=a; a:=b; b:=tg;
end;

Procedure      UpHeap(i:longint);
var      c,r:longint;
begin
      c:=pos[i];
      while c>1 do
      begin
            r:=c div 2;
            if minc[info[c]]<minc[info[r]] then
            begin
                  Swap(info[c],info[r]);
                  pos[info[c]]:=c;
                  pos[info[r]]:=r;
                  c:=r;
            end else break;
      end;
end;

end;
Procedure      DownHeap(i:longint);
var      c,r:longint;
begin
      r:=pos[i];
      while r*2<=nH do
      begin
            c:=r*2;

            if (c<nH)and(minc[info[c]]>minc[info[c+1]])then
            inc(c);

            if minc[info[c]]<minc[info[r]] then
```



```
begin
    Swap(info[c],info[r]);
    pos[info[c]]:=c;
    pos[info[r]]:=r;
    r:=c;
end else break;
end;
end;
Procedure      Insert(i:longint);
begin
    inc(nH);
    info[nH]:=i;
    pos[i]:=nH;
    UpHeap(i);
end;
Procedure      Del;
begin
    pos[info[1]]:=0;
    info[1]:=info[nH];
    pos[info[1]]:=1;
    dec(nH);
    DownHeap(info[1]);
end;
Procedure      Prim;
var      i,j,sc:longint;
begin
    for i:=2 to n do minc[i]:=vc;
    for      i:=index[0]+1      to      index[1]      do
minc[ke[i]]:=t[i];
    nH:=0;
    for i:=2 to n do Insert(i);
    res:=0;
    for sc:=2 to n do
        begin
            i:=info[1];
            res:=res+minc[i];
```



```

Del;
for j:=index[i-1]+1 to index[i] do
    if (pos[ke[j]]<>0) and
(t[j]<minc[ke[j]]) then
        begin
            minc[ke[j]]:=t[j];
            UpHeap(ke[j]);
        end;
    end;
end;
Procedure      Xuat;
begin
    assign(f,fo);
    rewrite(f);
    writeln(f,res);
    close(f);
end;
Begin
    Nhap;
    Chuyen;
    Prim;
    Xuat;
End.

```

L I K T

Trên ây là k t qu c a vi c tìm hi u, nghiên c u a vào gi ng d y c a nhóm giáo viên tin h c tr ng THPT Chuyên Thái Nguyên v cây khung và cây khung ng n nh t trên th . ó m i ch là nh ng ki n th c mang tính c s . Vi c cài t m c dù ã có s tìm tòi, k t h p vi c s d ng thu t toán v i c u trúc d li u tiên ti n song ch c ch n v n ch a áp ng c yêu c u cao trong vi c b i d ng h sinh gi i. R t mong c ng nghi p các n i óng góp ý ki n chúng tôi hoàn thi n ki n th c v v n này.

Xin trân tr ng c m n!



Chuyên x p lo i B

M T S NG D NG THU T TOÁN DIJKSTRA

Bùi Thu Hiền – THPT chuyên Thái Bình

L IM Đ U

Lý thuyết đồ thị là một phần quan trọng trong nội dung chương trình chuyên môn Tin học tại các trường chuyên. Hiện nay trong các kỳ thi học sinh giỏi đều có các bài toán liên quan đến lý thuyết đồ thị, do đó học sinh có kiến thức cơ bản về đồ thị sẽ có lợi thế trong việc giải quyết các bài toán liên quan đến lý thuyết đồ thị.

Trong tham luận của mình tôi xin đề cập đến **M t s ng d ng c a thu t toán Dijkstra** - tìm đường đi ngắn nhất giữa hai đỉnh trong đồ thị có trọng số không âm.

THU T TOÁN DIJKSTRA

Bài toán: Cho $G = (V, E)$ là đồ thị có hướng, mỗi cạnh $(u, v) \in E$ có trọng số $w(u, v)$ không âm. Yêu cầu tìm đường đi ngắn nhất từ đỉnh $s \in V$ đến đỉnh $t \in V$.

Thuật toán Dijkstra (E.Dijkstra - 1959) có thể mô tả như sau:

B c 1: Kh i t o

Với mỗi $v \in V$, gán nhãn $d[v]$ là độ dài đường đi ngắn nhất từ s đến v . Ban đầu $d[s] = 0$ và $d[v] = \infty$ với $v \neq s$. Nhãn của mỗi đỉnh có hai trạng thái: đã xử lý hoặc chưa xử lý. Nếu nhãn của đỉnh v là đã xử lý thì $d[v]$ đã là độ dài đường đi ngắn nhất từ s đến v nên không cần cập nhật thêm. Ngược lại, nếu nhãn của đỉnh v là chưa xử lý thì ta cần cập nhật $d[v]$ nếu có đường đi ngắn hơn từ s đến v qua các đỉnh đã xử lý. Việc cập nhật này được thực hiện bằng cách kiểm tra các đỉnh kề với v đã xử lý. Nếu $d[u] + w(u, v) < d[v]$ thì cập nhật $d[v] = d[u] + w(u, v)$. Quá trình này lặp lại cho đến khi nhãn của tất cả các đỉnh đều đã xử lý.

B c 2: L p

Bộ lập trình có hai thao tác:



- C nh nhn: Ch n trong các nh có nhn t do, l y ra nh u là nh có d[u] nh nh t, và c nh nhn nh u.
- S a nhn: Dùng nh u, xét t t c nh ng nh v và s a l i các d[v] theo công th c:

$$d[v] := \min(d[v], d[u] + c[u, v])$$

B c l p s k t thúc khi mà nh ích f c c nh nhn (tìm c ng íng n nh t t s t i f); ho c t i thao tác c nh nhn, t t c các nh t do u có nhn là + (không t n t i ng i). Có th t câu h i, thao tác 1, t i sao nh u nh v y c c nh nhn, gi s d[u] còn có th t i u thêm c n a thì t t ph i có m t nh t mang nhn t do sao cho $d[u] > d[t] + c[t, u]$. Do tr ng s c[t, u] không âm nên $d[u] > d[t]$, trái v i cách ch n d[u] là nh nh t. T t nhiên trong l n l p u tiên thì s là nh c c nh nhn do $d[s] = 0$.

B c 3: K t h p v i v i c l u v t ng i trên t ng b c s a nhn, thông báo ng íng n nh t tìm c ho c cho bi t không t n t i ng i ($d[f] = +$).

for ($\forall v \in V$) do $d[v] := +$;

$d[s] := 0$;

repeat

u := arg min($d[v] | \forall v \in V$); {L y u là nh có nhn d[u] nh nh t}

if (u = f) or ($d[u] = +$) then Break; {Ho c tìm ra ng íng n nh t t s t i f, ho c k t lu n không có ng}

for ($\forall v \in V: (u, v) \in E$) do {Dùng u t i u nhn nh ng nh v k v i u}

$d[v] := \min(d[v], d[u] + c[u, v]);$

until False;

Chú ý: N u th th a (có nhi u nh, ít c nh) ta có th s d ng danh sách k kèm tr ng s bi u di n th, tuy nhiên t c c a thu t toán Dijkstra v n khá ch m vì trong tr ng h p x u nh t, nó c n n l n c nh nhn và m i l n tìm nh c nh nhn s m t m t o n ch ng trình



v i ph c t p $O(n)$. thu t toán làm vi c hi u qu h n, ng i ta th ng s d ng c u trúc d li u Heap l u các nh ch a c nh nh n.

Bài t p

Bài 1: Ông Ngâu bà Ngâu

H n các b n ã bi t ngày "ông Ngâu bà Ngâu" hàng n m, ó là m t ngày y m a và n c m t. Tuy nhiên, m t ngày tr c ó, nhà Tr i cho phép 2 "ông bà" c oàn t . Trong v tr vùng thiên hà n i ông Ngâu bà Ngâu ng tr có N hành tinh ánh s t l n N, ông hành tinh Adam (có s hi u là S) và bà hành tinh Eva (có s hi u là T). H c n tìm ng p nh au.

N hành tinh c n i v i nh au b i m t h th ng c u v ng. Hai hành tinh b t k ch có th không có ho c duy nh t m t c u v ng (hai chi u) n i gi a chúng. H luôn i t i m c tiêu theo con ng ng n nh t. H i v i t c không i và nhanh h n t c ánh sáng. i m g p m t c a h ch có th là t i m t hành tinh th 3 nào ó.

Yêu c u: Hãy tìm m t hành tinh sao cho ông Ngâu và bà Ngâu cùng n ó m t lúc và th i gian n là s m nh t. B i tr ng, hai ng i có th cùng i qua m t hành tinh n u nh h n hành tinh ó vào nh ng th i i m khác nh au

D li u: vào t file v n b n ONGBANGAU.INP:

- Dòng u là 4 s N, M, S, T ($N \leq 100, 1 \leq S \leq T \leq N$), M là s c u v ng.
- M dòng ti p, m i dòng g m ba s nguyên I, J, L th hi n có c u v ng n i gi a hai hành tinh i và J có dài là L ($1 \leq I \leq J \leq N, 0 < L \leq 200$).

K t qu : ghi ra file v n b n ONGBANGAU.OUT: do tính ch t c u v ng, m i n m m t khác, nên n u nh không t n t i hành tinh nào tho m n yêu c u thì ghi ra m t dòng ch CRY. N u có nhi u hành tinh tho m n thì ghi ra hành tinh có ch s nh nh t.

Ví d :

ONGBANGAU . INP	ONGBANGAU . OUT
4 4 1 4	2
1 2 1	
2 4 1	
1 3 2	
3 4 2	



Thu t toán:

Ta có nh n xét:

- + Hai hành tinh b t kì ch c n i n nhau b i nhi u nh t m t c u v ng
- + Ông Ngâu và bà Ngâu luôn i t i m c tiêu theo con ng ng n nh t
- + H i v i v n t c không i và nhanh h n v n t c ánh sáng

Th c ch t ây là m t bài toán th , ta có thu t toán nh sau:

T hành tinh S (n i ông Ngâu) ta xây d ng b ng SP, trong ó SP[i] là ng i ng n nh t t hành tinh S n hành tinh i (do ông Ngâu luôn i t i m c tiêu theo con ng ng n nh t). SP[i] = 0 t c là không có ng i t hành tinh S n hành tinh i.

T ng t ta s xây d ng b ng TP, trong ó TP[i] là ng i ng n nh t t hành tinh T n hành tinh i. Và TP[i] = 0 t c là không có ng i t hành tinh T n hành tinh i.

Do yêu c u c a bài toán là tìm hành tinh khác S và T mà 2 ông bà Ngâu cùng n m t lúc và trong th i gian nhanh nh t. T c là ta s tìm hành tinh h sao cho (h khác S và T) và (SP[h] = ST[h]) t giá tr nh nh t khác 0. N u không có hành tinh h nào tho mẫn thì ta thông báo CRY

xây d ng m ng SP và ST ta ch n gi i thu t Dijkstra tìm ng i ng n nh t gi a 2 nh th .

Bài 2: Đôi b n

Tr c kia Tuấn và Mai là hai b n cùng l p còn bây gi hai b n h c khác tr ng nhau. C m i sáng, úng ó gi c hai u i t nhà t i tr ng c a mình theo con ng m t ít th i gian nh t (có th có nhi u con ng i m t th i gian b ng nhau và u ít nh t). Nh ng hôm nay, hai b n mu ng p nhau bàn vi c h p l p c nhân ngày 20-11.

Cho bi t s giao thông c a thành ph g m N nút giao thông c ánh s t 1 n N và M tuy n ng ph (m i ng ph n i 2 nút giao thông). V trí nhà c a Mai và Tuấn c ng nh tr ng c a hai b n u n m các nút giao thông. C n xác nh xem Mai và Tuấn có cách nào i tho mẫn yêu c u nêu trên, ng th i h i có th g p nhau nút giao thông nào ó trên con ng t i tr ng hay không? (Ta nói Tuấn và Mai có th g p nhau t i m t nút giao thông nào ó n u h n nút giao thông này t i cùng m t th i



i m). N u có nhi u ph ng án thì hãy ch ra ph ng án Mai và Tu ng p nhau s m nh t.

D li u: vào t file v n b n FRIEND.INP

- Dòng u tiên ch a 2 s nguyên d ng N, M ($1 \leq N \leq 100$);
- Dòng ti p theo ch a 4 s nguyên d ng H_a, S_a, H_b, S_b l n l t là s hi u các nút giao thông t ng ng v i: Nhà Tu n, tr ng c a Tu n, nhà Mai, tr ng c a Mai.
- Dòng th i trong s M dòng ti p theo ch a 3 s nguyên d ng A, B, T . Trong ó A và B là hai u c a tuy n ng ph i. Còn T là th i gian (tính b ng giây ≤ 1000) c n thi t Tu n (ho c Mai) i t A n B c ng nh t B n A.

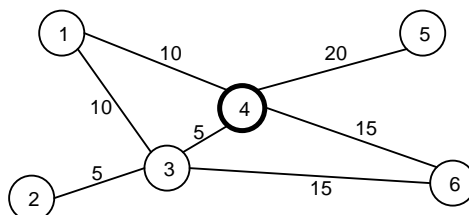
Gi thi t là s giao thông trong thành ph m b o có th i t m t nút giao thông b t k n t t c các nút còn l i.

K t qu : ghi ra file v n b n FRIEND.OUT

- Dòng 1: Ghi t YES hay NO tu theo có ph ng án giúp cho hai b n g p nhau hay không. Trong tr ng h p có ph ng án:
- Dòng 2: Ghi th i gian ít nh t Tu n t i tr ng
- Dòng 3: Ghi các nút giao thông theo th t Tu n i qua
- Dòng 4: Ghi th i gian ít nh t Mai t i tr ng
- Dòng 5: Ghi các nút giao thông theo th t Mai i qua
- Dòng 6: Ghi s hi u nút giao thông mà hai b n g p nhau
- Dòng 7: Th i gian s m nh t tính b ng giây k t 6 gi sáng mà hai b n có th g p nhau.

Ví d : V i s giao thông sau: ($N=6, M=7, H_a=1, S_a=6, H_b=2, S_b=5$)

Dòng	FRIEND.I	FRIEND.OU
g	NP	T
1	6 7	YES
2	1 6 2 5	25
3	1 3 10	1 4 6
4	1 4 10	30
5	2 3 5	2 3 4 5
6	3 4 5	4
7	3 6 15	10
8	4 5 20	





9	4 6 15	
---	--------	--

Thu t toán:

S d ng thu t toán Dijkstra, xây d ng th t c: Dijkstra(start:intger, var d: m ng_nhân); xây d ng m ng nhân d cho ng i ng n nh t t i m xu t phát start n m i nh (có th t i t xu t phát). Sau ó g i th t c này 4 l n b ng các l i g i:

Dijkstra(ha,d1); s c d1 cho bi t các ng i ng n nh t xu t phát t nhà Tu n

Dijkstra(sa,d2); s c d2 cho bi t các ng i ng n nh t xu t phát t nhà Mai

Dijkstra(hb,d3); s c d3 cho bi t các ng i ng n nh t xu t phát t tr ng Tu n

Dijkstra(sb,d4); s c d4 cho bi t các ng i ng n nh t xu t phát t tr ng Mai

i m h n là nút u c n th a m n các i u ki n sau:

$d1[u] + d3[u] = d1[sa]$ {th i gian Tu n i t nhà t i i m h n + Tu n}

$d2[u] + d4[u] = d2[sb]$ {th i gian Mai i t nhà t i i m h n + t i m h n t i tr ng Mai}

$d1[u] = d2[u]$ {th i gian i t nhà t i i m h n c a Tu n và Mai b ng nhau}

$d1[u]$ nh nh t {th i gian Tu n i t nhà t i i m h n s m nh t}

ghi k t qu vào file FRIENDS.OUT, c n g i th t c Dijkstra m t l n n a:
Dijkstra(u,d); s c m ng d(N) cho bi t nh n ng i ng n nh t

Bài 3: Đ ng đi gi i h n

M t m ng giao thông g m N nút giao thông ánh s t 1 n N. V i m i c p nút i, j có ng i hai chi u và trên o n ng ó, ng i ta quy nh m t chi u cao nguyên không âm $c[i,j]$ không l n h n 6000 là chi u cao t i a cho m i xe i trên o n ng ó ($c[i,j]=0$ có ngh a là không có ng i t i n j). Cho hai nút s và t. Hãy tìm m t hành trình t s n t qua các nút khác nhau sao cho chi u cao cho phép t i a v i xe ch y trên hành trình ó là l n nh t có th c.

D li u: vào t file v n b n HIGHT.INP :

- Dòng th nh t ghi 3 s N, s, t ($N \leq 100$)



- Tìm theo là m t s dòng, m i dòng ghi 3 s i, j, m v i ý nghĩa có ng i hai chi u t i n j v i chi u cao cho phép h.

K t qu : ghi ra file v n b n HIGHT.OUT

- Dòng th nh t ghi s h là chi u cao cho phép, n u $h > 0$ trong m t s dòng tìm theo, m i dòng ghi m t nh trên hành trình l n l t t s n t v i chi u cao t i a cho phép là h.

Thu t toán:

G i $H[i]$ là chi u cao l n nh t có th c a x e i t s n i

Kh i t o gán $H[s] := +\infty$ và $H[i] = 0$ v i i s

Thu t toán s a nh n t ng t thu t toán Dijkstra

Repeat

$u := 0$; $max := 0$;

 for $v := 1$ to n do

 if $free[v]$ and $(h[v] > max)$ then

 begin

$max := h[v]$; $u := v$;

 end;

 if $u = 0$ then break;

$free[u] := false$;

 for $v := 1$ to n do

 if $a[u, v]$ then

 if $h[v] < \min(h[u], c[u, v])$ then

 begin

$h[v] := \min(h[u], c[u, v])$;

$trace[v] := u$;

 end;

until false;



Bài 4: T ng s đ ng đi ng n nh t

Cho th vô h ng G g m N nh, M c nh, m i c nh có 1 tr ng s nguyên d ng, gi a hai nh b t kì có không quá m t c nh n i. Cho tr c hai nh s và t, hãy tính s ng i ng n nh t t s n t. Hai ng i khác nhau n u th t các nh trên 2 ng i khác nhau.

Thu t toán:

K t h p Dijkstra v i quy ho ch ng

- Theo thu t toán Dijkstra g i $d[i]$ là dài ng i ng n nh t t nh s n nh i.

Kh i t o $d[i]=+$ v i m i i s và $d[s]=0$

- Quy ho ch ng g i $f[i]$ là s ng i ng n nh t t nh s n nh i.

Kh i t o $f[i]=0$ v i m i i s và $f[s]=1$

Trong ch ng trình Dijkstra:

- M i khi tìm c ng i m i có dài ng n h n ($d[v]>d[u]+c[u,v]$) ta t i n hành thay i $d[v]:=d[u]+c[u,v]$ ng th i $f[v]:=f[u]$.

- M i khi tìm c 2 ng i có dài b ng nhau ($d[v]=d[u]+c[u,v]$) ta thay i $f[v]:=f[v]+f[u]$.

K t qu c n tìm là $f[t]$

o n ch ng trình Dijkstra k t h p quy ho ch ng

```
For v:=1 to n do d[v]:=maxlongint; d[s]:=0;
```

```
For v:=1 to n do f[v]:=0; f[s]:=1;
```

```
Fillchar(Free,sizeof(Free),true);
```

```
Repeat
```

```
    U:=0; mi:=maxlongint;
```

```
    For v:=1 to n do
```

```
        If (Free[v]) and (d[v]<mi) then
```

```
            Begin
```

```
                mi:=d[v];
```

```
                u:=v;
```

```
            end;
```

```
    If u=0 then break;
```

```
    Free[u]:=false;
```

```
    For v:=1 to n do
```

```
        If Free[v] then
```



```

If d[v]>d[u] + c[u,v] then
  Begin
    d[v]:=d[u] + c[u,v];
    f[v]:=f[u];
  end
Else if d[v] = d[u] + c[u,v] then
  f[v]:=f[v] + f[u];
Until false;

```

S d ng Dijkstra đ đ t c n cho m t s bài toán duy t

Bài 5: ROADS

N thành ph c ánh s t 1 n N n i v i nhau b ng các con ng m t chi u. M i con ng có hai giá tr : dài và chi phí ph i tr i qua. Bob thành ph 1. B n hãy giúp Bob tìm ng i ng n nh t n thành ph N, bi t r ng Bob ch có s ti n có h n là K mà thôi.

D li u: vào t file v n b n ROADS.INP

Dòng u tiên ghi t là s test. V i m i test:

- Dòng u ghi s nguyên K (0 K 10000) là s ti n t i a mà Bob còn có th chi cho l phí i ng.
- Dòng 2 ghi s nguyên N (2 N 100) là s thành ph .
- Dòng 3 ghi s nguyên R (1 R 10000) là s ng n i.
- M i dòng trong N dòng sau ghi 4 s nguyên S, D, L, T mô t m t con ng n i gi a S và D v i dài L (1 L 100) và chi phí T (0 T 100).

K t qu : ghi ra file v n b n ROADS.OUT

V i m i test, in ra dài ng i ng n nh t t 1 n N mà t ng chi phí không quá K. N u không t n t i, in ra -1.



Ví d :

ROADS . INP	ROADS . OUT
2	11
5	-1
6	
7	
1 2 2 3	
2 4 3 3	
3 4 2 4	
1 3 4 1	
4 6 2 1	
3 5 2 0	
5 4 3 2	
0	
4	
4	
1 4 5 2	
1 2 1 0	
2 3 1 1	
3 4 1 0	

Thu t toán:

S d ng thu t toán Dijkstra:

- L n 1: tìm ng i ng n nh t (v kho ng cách) ng c t nh N v các nh khác t o m ng mindist.
- L n 2: tìm ng i ng n nh t (v chi phí ti n) ng c t nh N v các nh khác t o m ng mincost.

Hai m ng mindist và mincost s c dùng làm c n cho quá trình duy t sau:

Th c hi n duy t theo các nh t nh 1. Gi s ã duy t t i nh i, và ã i c quăng ng là d và s t i n ã tiêu là t. Ngay u th t c Duyet(i,d,t) t c n:

N u $(d + \text{mindist}[i]) \geq$ ng i c a ph ng án t t nh t) thì không c n duy t t i p ph ng án hi n th i n a.

N u $(t + \text{mincost}[i]) > s$ t i n có c a Bob là k) thì không c n duy t t i p ph ng án hi n th i n a.

Trong ch ng trình chính g i th t c Duyet(1,0,0).

Chú ý: quá trình tìm nh duy t t i p theo c nhanh chóng ta c n t ch c danh sách k .



Ch ng trình tham kh o:

```
const    fi          = 'ROADS.INP';
         fo          = 'ROADS.OUT';
         maxn        = 100;
         infinity     = 20000;
         maxtime      = 180;

type     pt          = ^tnode;
         tnode       = record
                           v          : byte;
                           l, t      : byte;
                           next      : pt;
                        end;
         m1          = array[1..maxn] of word;
         m2          = array[1..maxn, 1..maxn] of word;

var
  list           : array[1..maxn] of pt;
  dd             : array[1..maxn] of boolean;
  cost, dist     : m2;
  mincost, mindist : m1;
  k             : word;
  n             : byte;
  best          : word;
  f,g           : text;
  t,test: longint;

procedure init;
var i, r, u, v, l, t : word;
    tmp              : pt;
begin

  readln(f, k);    {so tien cua Bob}
  readln(f, n);    {so thanh pho}
  readln(f, r);    {so con duong}

  for u:=1 to n do {khoi tri nhan gia tien , nhan khoang cach}
    for v:=1 to n do
      begin
        cost[u, v]:=infinity;
        dist[u, v]:=infinity;
      end;
  end;
```



{to chuc cac danh sach lien ket 1 chieu cua cac con duong. Moi danh sach

```
list[i] cho biet cac thanh pho co duong truc tiep tu i sang}
for i:=1 to n do {khởi tri cac nút góc của các danh sách liên kết
list[i]}
    list[i]:=nil;
for i:=1 to r do
    begin
        readln(f, u, v, l, t);
        new(tmp);
        tmp^.v:=v;
        tmp^.l:=l;
        tmp^.t:=t;
        tmp^.next:=list[u];
        list[u]:=tmp;

        {so gian lai du lieu}
        if l < dist[u, v] then
            dist[u, v]:=l;
        if t < cost[u, v] then
            cost[u, v]:=t;
    end;
end;
```

```
procedure dijkstra(var a : m2; var dist : m1);
{Thuật toán dijkstra tìm khoảng cách ngắn nhất từ thành phố i tới thành phố N}
var    chua           : array[1..maxn] of boolean;
        min           : word;
        i, j, last    : byte;
begin
    fillchar(chua, sizeof(chua), true);    {mảng đánh dấu thành phố đã xét}
    for i:=1 to n do
        dist[i]:=infinity;    {khởi tri mảng nhận}
    dist[n]:=0;                {nhận của đỉnh N}
    chua[n]:=false;           {đánh dấu đã xét N}

    last:=n;                  {last: đỉnh chưa xét có nhận nhỏ nhất}
    for i:=2 to n do    {n-1 lần sửa nhận thì xong}
        begin
            {sửa nhận cho các đỉnh j chưa xét đưa vào nhận của last}
            for j:=1 to n do
                if chua[j] and (a[j, last] + dist[last] < dist[j]) then
```



```
dist[j]:=dist[last] + a[j,
last];
    {tim dinh chua xet o nhan nho nhat}
    min:=infinity+1;
    for j:=1 to n do
    if chua[j] and (dist[j] < min) then
    begin
        min:=dist[j];
        last:=j;
    end;
    {danh dau da xet xong dinh last}
    chua[last]:=false;
end;
end;

procedure try(last : byte; l, t : word);
{Duyet tiep khi Bob da toi thanh pho last, da di doan duong l, da
tieu t xu}
var tmp : pt;
begin
    if (l + mindist[last] >= best) {dk can ve duong di}
    or (t + mincost[last] > k) then {dk can ve tien}
        exit;
    if last = n then {ve toi dich: Diem dung de quy}
    begin
        best:=l;
        exit;
    end;

    tmp:=list[last]; {tmp: thanh pho last}
    while tmp <> nil do {duyet chon cac de cu cho thanh pho tiep
theo last}
    begin
        if not dd[tmp^.v] then {thanh pho v chua qua}
        begin
            dd[tmp^.v]:=true; {danh dau da qua v}
            try(tmp^.v, l+tmp^.l, t+tmp^.t); {di tiep tu v}
            dd[tmp^.v]:=false; {quay lui}
        end;
        tmp:=tmp^.next; {de cu thanh pho khac}
    end;
end;

procedure process;
begin
```



```
{xay dung cac mang cost va dist de lam can phuc vu duyét de quy}
dijkstra(cost, mincost);
dijkstra(dist, mindist);
{khoi tri}
best:=infinity;
fillchar(dd, sizeof(dd), false);
try(1, 0, 0); {duyet tu thanh pho 1 (duong da di =0, tien da
tieu=0}
end;

procedure done;
begin
    if best = infinity then writeln(g, -1)
    else writeln(g, best);
end;

BEGIN
    assign(f, fi); reset(f);
    assign(g, fo); rewrite(g);
    readln(f, test);
    for t:=1 to test do
        begin
            init;
            process;
            done;
        end;
    close(f); close(g);
END.
```

Bài 6: Du lịch hè

Cho N thành phố đánh số từ 1 đến N . Một người muốn đi du lịch thành phố A đến thành phố B và sau đó quay lại A . Người đó muốn đi trên đường đi từ B về A không quay lại những thành phố đã qua trên đường đi từ A đến B . Hãy tìm cho người đó một hành trình với chi phí ít nhất.

Dữ liệu: vào tệp văn bản TOURIST.INP

- Dòng thứ t ghi 3 số nguyên dương N, A, B ($N \leq 100, 1 \leq A, B \leq N$) trong đó N là số thành phố, A là thành phố xuất phát, B là thành phố cần đến.
- Các dòng tiếp theo mỗi dòng ghi 3 số nguyên dương i, j, k với ý nghĩa: giữa thành phố i và thành phố j có đường đi trực tiếp và chi phí k đi quãng đường đó là k .

Kết quả: ghi ra tệp văn bản TOURIST.OUT

- Dòng thứ t ghi chi phí tối thiểu đi trên hành trình tìm được.



- Dòng thứ hai ghi các thành phố đi qua trên hành trình tìm kiếm theo đúng thứ tự, cách nhau 1 dấu cách. Nếu không có cách đi nào như vậy thì ghi thông báo "No Solution"

Ví dụ :

TOURIST.INP	TOURIST.OUT
10 1 5	14
1 2 2	1 2 3 4 5 6 7 8 9 10 1
2 3 2	
3 4 2	
4 5 2	
5 6 1	
6 7 1	
7 8 1	
8 9 1	
9 10 1	
10 1 1	
1 9 5	
9 3 5	
3 7 5	
7 5 5	

Thuật toán:

Dùng thuật toán duy nhất có quay lui và đánh giá cận tìm kiếm ngắn nhất thành phố xuất phát A đến thành phố đích B.

Tìm kiếm, thử chọn một thành phố vào hành trình đó, ta đánh dấu tất cả các thành phố đã đi qua giữa thành phố A và thành phố j, sau đó dùng thuật toán Dijkstra tìm đường ngắn nhất (là chi phí) ngắn nhất từ thành phố j quay về thành phố A (không đi qua các thành phố đã đánh dấu). Nếu không tìm thấy đường thì gán chi phí là +

Nếu chi phí từ A đến thành phố j (tìm kiếm của quá trình duy nhất) bằng với chi phí cho đường ngắn nhất từ j về A không tồn tại giá trị nhỏ nhất thì tìm kiếm tiếp có thể loại bỏ những án chọn thành phố j và thử sang những án khác.

Tập hợp dữ liệu:

- Gọi $X[0..N]$ là mảng lưu các thành phố đi qua trong quá trình duy nhất, $X[i]$ sẽ là thành phố đi qua tại bước thứ i trong tiến trình duy nhất.



- bi t $X[0] := A$. có nghi m t i u, dùng m ng $LX[0..N]$ l u l i hành trình t t nh t khi duy t.
- G i D là m ng ánh d u, ta s ánh d u b ng các s 0, 1, 2. Kh i t o ban u các nh u ch a ánh d u ngo it tr nh xu t phát A : $D[i] = 0$ v i m i i A ; $D[A] := 1$;
 - M ng Tien[0..N] có ý ngh a: Tien[i] cho ta bi t chi phí khi n thành ph th i trong duy t (là thành ph $X[i]$). Kh i t o Tien[i] := 0
 - M i b c th ch n thành ph j vào hành trình t i b c th i ($D[j] = 0$), ta t chi phí t i thành ph j là Tien[i] b ng chi phí cho n thành ph tr c ó là Tien[i-1] c ng v i chi phí t thành ph tr c ó (là $X[i-1]$) t i thành ph j v a ch n. ng th i ánh d u thành ph j ã i qua là $D[j] := 1$;
 - Vì t m t hàm Dijkstra(j) cho ta chi phí ít nh t t j v A
 - o Tr c h t xóa ánh d u cho 2 nh j và A: $D[j] = D[A] = 0$
 - o Sau ó áp d ng thu t toán Dijkstra trên t p các nh i có $D[i] = 0$. M i l n c nh nh n cho nh i ta t $D[i] = 2$
 - o Tr c khi k t thúc, ánh d u l i 2 nh j và A, ng th i t l i t t c các $D[i] = 2$ tr v 0 (ngh a là ph c h i l i m ng ánh d u D nh c không làm h ng t i n trình duy t ti p).
 - o t ng t c , hàm này không c n l u v t ng i mà ch c n tr l i dài ng i ng n nh t (hàm này tr v + n u không có ng quay v .
 - T i m i nút th i c a duy t, ta ánh giá c n: Tien[i] + Dijkstra($X[i]$) là dài ng i t A n $X[i]$ c ng v i dài ng i ng n nh t t $X[i]$ quay v A. N u con s này nh h n chi phí ng i tr c ó là MinT thì ta t i p t c tìm ki m, ng c l i thì không duy t ti p n a. Khi n c B thì ghi nh n ng i
 - K t thúc duy t, n u không ghi nh n c ng nào (MinT = +) thì ghi "No Solution". Ng c l i, tìm c ng i t A n B (và có ng quay v A không i l p l i b t c m t thành ph nào) và chi phí trên c chu trình là t i thi u thì in ra ng i t A n B (d a vào m ng LX) và áp d ng thu t toán Dijkstra (l n này có l u v t ng i) in ra ng quay v t B n A.



Bài 7: Cuộc đua tỉ p s c

Vùng t Alpha có N thành ph c ánh s t 1 n N. Giả hai thành ph có th có ng n i tr c tỉ p ho c không. Các con ng này c ánh s t 1 t i M. Ban lãnh o th d c th thao vùng Alpha t ch c cu c ch y ua tỉ p s c “thông minh” theo quy lu t sau:

- Thành ph xu t phát là thành ph 1, thành ph ích là thành ph N
- M i i thi u có K ng i d thi. L n l t t ng ng i ch y t thành ph 1 v thành ph N
- Khi ng i th nh t n c thành ph N thì ng i th hai m i b t ur i kh i thành ph 1, khi ng i th hai n c thành ph N thì ng i th ba m i b t ur i kh i thành ph 1, ..., ng i th i n c thành ph N thì ng i th i+1 m i b t ur i kh i thành ph 1, ..., (i<K). Ng i th K v t i ích t i th i i m nào thì th i i m ó c coi là th i i m v ích c a toàn i.
- ng ch y c a các i viên không c gi ng nhau hoàn toàn.
- Có th ch y l i o n ng ã ch y.

Hãy vi t ch ng trình tính th i gian nh nh t m t i hoàn thành cu c ch y ua tỉ p s c nêu trên n u các v n ng viên có t c ch y nh nhau.

D li u: vào t file v n b n RELAY.INP

- Dòng u tiên ghi 3 s nguyên d ng K, N, M ($2 \leq K \leq 40$; $4 \leq N \leq 800$; $1 \leq M \leq 4000$)
- M dòng tỉ p theo, m i dòng ch a 3 s nguyên i, j, w th hi n m t ng i tr c tỉ p gi a hai thành ph i và j m t th i gian ch y là w (n v th i gian) ($1 \leq i, j \leq N$; $1 \leq w \leq 9500$).

K t qu : ghi ra file v n b n RELAY.OUT:

- Dòng th nh t ch a m t s nguyên duy nh t là th i gian ch y nh nh t c a m t i
- K dòng tỉ p theo, m i dòng th hi n hành trình ch y c a m t v n ng viên trong i là dãy s hi u các thành ph liên tỉ p trên hành trình ó.

Ví d :



RELAY . INP	RELAY . OUT
4 5 8	23
1 2 1	1 3 2 5
1 3 2	1 3 5
1 4 2	1 2 1 2 5
2 3 2	1 2 5
2 5 3	
3 4 3	
3 5 4	
4 5 6	

Thu t toán:

C i ti n t thu t toán Dijkstra c i n

G i $L[i,j]$ là dài ng i th j trong K ng i ng n nh t t nh 1 n nh i ($i=1, 2, \dots, N; j=1, 2, \dots, k$). Kh i t o $L[i,j]$ b ng vô cùng v i m i i, j và $L[1,1]=0$.

- M i l n tìm m t c p (i, j) ch a ánh d u có nhấ n $L[i, j]$ nh nh t
- T (i, j) ta t i n hành s a nhấ n cho các c p (i, j) th a mấ n: i k v i i i , c p (i, j) ch a c ánh d u và $L[i, j] \geq L[i, j] + C[i, i]$ (*)

Kh i i u k i n (*) x y ra thì ng i ng n nh t th j t i nh i s thành ng i ng n nh t th j+1 t i i và ng ng n nh t th j t i i s thành ng i qua i i tr c, r i t i i.

Do ó v i m i c p (i, j) th a mấ n (*) ta s s a nhấ n cho c p (i, j) và các c p có liên quan nh sau: $L[i, j+s] := L[i, j+s-1]$ v i m i s=1 n k-j và $L[i, j] = L[i, j] + C[i, i]$

T ng t c p nh t l i v t ng i c a các c p (i, j)

- ánh d u c p (i, j) ã c nh nhấ n

Quá trình l p l i cho n khi không còn c p (i, j) nào ch a c nh nhấ n ho c c p (n, k) ã c c nh nhấ n.

Sau cùng ta tính t ng dài t i u c a toàn i K v n ng viên

$$\text{Minpath} = L[N, 1] + L[N, 2] + \dots + L[N, K]$$

và tìm hành trình c a t ng v n ng viên d a vào m ng theo dõi v t ng i.

V i s nh và s c nh c a th t ng i l n, c n t ch c danh sách k .



LUY N T P

Bài 1: Đ n tr ng

Ngày 27/11 t i là ngày t ch c thi h c k l tr ng H BK. Là sinh viên n m th nh t, Hi u không mu n vì i mu n mà g p tr c tr c phòng thi nên ã chu n b khá k càng. Ch còn l i m t công vi c khá gay go là Hi u không bi t i ng nào t i tr ng là nhanh nh t.

Th ng ngày Hi u không quan tâm t i v n này l m cho nên bây gi Hi u không bi t ph i làm sao c . B n thành ph là g m có N nút giao thông và M con ng n i các nút giao thông này. Có 2 lo i con ng là ng 1 chi u và ng 2 chi u. dài c a m i con ng là m t s nguyên d ng.

Nhà Hi u nút giao thông 1 còn tr ng H BK nút giao thông N. Vì m t l trình ng i t nhà Hi u t i tr ng có th g p nhi u y u t khác nh là g p nhi u ền , i qua công tr ng xây d ng, ... ph i gi m t c cho nên Hi u mu n bi t là có t t c bao nhiêu l trình ng n nh t i t nhà t i tr ng. B n hãy l p trình giúp Hi u gi i quy t bài toán khó này.

D li u: vào t file v n b n ROADS.INP

- Dòng th nh t ghi hai s nguyên N và M.
- M dòng ti p theo, m i dòng ghi 4 s nguyên d ng K, U, V, L. Trong ó:
K = 1 có ngh a là có ng i m t chi u t U n V v i dài L.
K = 2 có nghĩa là có ng i hai chi u gi a U và V v i dài L.

K t qu : ghi ra file v n b n ROADS.OUT hai s là dài ng i ng n nh T và s l ng ng i ng n nh t. Bi t tr ng s l ng ng i ng n nh t không v t quá ph m vì int64 trong pascal hay long long trong C++.

Ví d :

ROADS . INP	ROADS . OUT
3 2	4 1
1 1 2 3	
2 2 3 1	

Gi i h n:

- 1 N 5000
- 1 M 20000
- dài các con ng 32000



Bài 2: HIWAY

M t m ng giao thông g m N nút giao thông, và có M ng hai chi u n i m t s c p nút, thông tin v m t ng g m ba s nguyên d ng u, v là tên hai nút u nút c a ng, và w là dài o n ng ó. Bì t r ng hai nút giao thông b t kì có không quá 1 ng hai chi u nh n chúng làm hai u nút. Cho hai nút giao thông s và f, hãy tìm hai ng i n i gi a s v i f sao cho hai trên hai ng không có c nh nào c i qua hai l n và t ng dài 2 ng i là nh nh t.

D li u: vào t file v n b n HIWAY.INP

- Dòng u ghi N, M (N 100)
- Dòng th 2 ghi hai s s, f.
- M dòng ti p theo, m i dòng mô t m t ng g m ba s nguyên d ng u, v, w.

K t qu : ghi ra file v n b n HIWAY.OUT

- Dòng u ghi T là t ng dài nh nh t tìm c ho c -1 n u không tìm c.
- N u tìm c, hai dòng sau, m i dòng mô t m t ng i g m: s u là s nút trên ng i này, ti p theo là dãy các nút trên ng i b t u t s, k t thúc t i f.

(Ph m vi tính toán trong vòng Longint)

Ví d :

HIWAY.INP	HIWAY.OUT
5 8	5
1 5	3 1 3 5
1 2 1	4 1 2 4 5
1 4 8	
2 3 5	
2 4 1	
3 5 1	
4 3 8	
4 5 1	
1 3 1	



Bài 3: SHORTEST

M t h th ng giao thông g m N thành ph và M o n ng m t chi u. Các thành ph có s hi u t 1 n N. M i o n ng ta bi t thành ph xu t phát và thành ph ích và dài. Ta nói r ng o n ng F là *ti p n i c a* o n ng E n u thành ph ích c a o n ng E là thành ph xu t phát c a o n ng F. M t *hành trình* t thành ph A n thành ph B là m t dãy liên ti p các o n ng sao cho thành ph xu t phát c a o n ng u tiên là A, m i o n ng khác là ti p n i c a m t o n ng tr c ó và thành ph ích c a o n ng cu i cùng là thành ph B. dài c a hành trình là t ng dài c a các o n ng trong hành trình. M t hành trình t A n B là *hành trình ng n nh t* n u không có hành trình nào t A n B có dài ng n h n.

Yêu c u: V i m i o n ng, cho bi t có bao nhiêu hành trình ng n nh t ch a o n ng ó.

D li u: Cho trong t p SHORTEST.INP g m có:

- Dòng u ghi hai s nguyên N và M ($1 \leq N \leq 1500$, $1 \leq M \leq 5000$), là s thành ph và s o n ng.
- Dòng th i trong M dòng ti p ch a ba s nguyên U_i, V_i, L_i t ng ng là thành ph xu t phát, thành ph ích và dài c a o n ng th i (các o n ng u là m t chi u; các s U_i, V_i là khác nhau và giá tr L_i t i a là 10000).

K t qu : Ghi ra t p SHORTEST.OUT g m có M dòng, trong ó dòng th i dòng ghi m t s nguyên C_i là s hành trình ng n nh t khác nhau ch a o n ng th i (vì s C_i có th là r t l n nên b n hãy vi t nó d i d ng s d c a 1 000 000 007).



Ví dụ :

SHORTEST . INP	SHORTEST . OUT
4 3	3
1 2 5	4
2 3 5	3
3 4 5	
4 4	2
1 2 5	3
2 3 5	2
3 4 5	1
1 4 8	
5 8	0
1 2 20	4
1 3 2	6
2 3 2	6
4 2 3	6
4 2 3	7
3 4 5	2
4 3 5	6
5 4 20	

L I K T

Bài tập này đã giúp thu thập toán Dijkstra vô cùng phong phú và đa dạng, từ cơ bản đến nâng cao. Phần lớn các ví dụ nêu ra trong tham luận của tôi không phải là những tài liệu tham khảo khác nhau.. Tôi rất mong nhận được ý kiến đóng góp của các quý thầy cô tham luận hoàn thiện hơn.

Rất mong các đồng nghiệp các bạn đóng góp ý kiến chúng tôi hoàn thiện khi cần thiết và cần này.

Xin trân trọng cảm ơn!



Chuyên x p lo i B

TÊN CHUYÊN :
PHÉP DUY T M T TH

GIÁO VIÊN: NG TU N THÀNH
TR NG THPT CHUYÊN NGUY N T T THÀNH – YÊN BÁI

Các bài toán v th ngày càng c quan tâm nghiên c u, phát tri n, ng d ng trong khoa h c và cu c s ng. M t trong nh ng cách ti p c n các bài toán này là Phép duy t th . Trong ph m vi tham lu n c a mình tôi xin c p n m t s phép duy t th c b n, hi u qu .

Nh b n ã bi t: Khi bi t g c c a m t cây ta có th th c hi n phép duy t cây ó th m các nút c a cây theo th t nào y. V i th v n t ra c ng t ng t . Xét m t th không nh h ng $G(V,E)$ và m t nh v trong $V(G)$, ta c n th m t t c các nh thu c G mà có th v i t i c nh v (ngh a là th m m i nút liên thông v i v).

Ta có hai cách gi i quy t trên ây: Phép tìm ki m theo chi u sâu (Depth First Search-DFS) và phép tìm nhi u theo chi u r ng (Breadth First Search-BFS).

1. Tìm ki m theo chi u sâu.

nh xu t phát v c th m, ti p theo ó m t inh w ch a c th m, mà là lân c n c a v, s c ch n và m t phép tìm ki m theo chi u sâu xu t phát t w l i c th c hi n.

Khi m t nh u ã c v i t i mà m i nh lân c n c a nó u ã c th m r i, thì ta s quay ng c lên nh cu i cùng v a c th m (mà còn có nh w lân c n v i nó ch a c th m). Và m t phép tìm ki m theo chi u sâu xu t phát t w l i c th c hi n. Phép tìm ki m s k t thúc khi không còn m t nút nào ch a c th m mà v n có th v i t i c t nút ã c th m.

Gi i thu t c a phép duy t này:

Procedure DFS(v)

- 1) Visited(v) :=1; //Visited dùng ánh d u các nh ã c th m
- 2) **For** m i nh w lân c n c a v **Do**
If Visited(w)=0 then Call DFS(w);
- 3) **Return**



Ta th y: Trong tr ng ph p G c bi u di n b i m t danh sách lân c n thì nh w lân c n c a v s c xác nh b ng cách d a vào danh sách móc n i ng v i v. Vì gi i thu t DFS ch xem xét m i nút trong m t danh sách lân c n nhi u nh t m t l n mà thôi mà l i có 2e nút danh sách (ng v i e cung), nên th i gian hoàn thành phép tìm ki m ch là $O(e)$. Còn n u G c bi u di n b i ma tr n lân c n thì th i gian xác nh m i nh lân c n c a v là $O(n)$. Vì t i a có n nh c th m, nên th i gian tìm ki m t ng quát s là $O(n^2)$.

Gi i thu t DFS(V_1) s m b o th m m i nh liên thông v i V_1 . T t c các nh c th m cùng v i các cung liên quan t i các nh ó g i là m t b ph n liên thông (vùng liên thông) c a G. V i phép duy t DFS ta có th xác nh c G có liên thông hay không, ho c tìm c các b ph n liên thông c a G n u G không liên thông.

Áp d ng gi i thu t tìm ki m theo chi u sâu DFS gi i các bài toán sau, s giúp ta hi u h n v DFS.

Bài toán: **Bãi c ngon nh t - VBGRASS**

Bessie d nh c ngày s nhai c xuân và ng m nhìn c nh xuân trên cánh ng c a nông dân John, cánh ng này c chia thành các ô vuông nh v i R ($1 \leq R \leq 100$) hàng và C ($1 \leq C \leq 100$) c t. Bessie c gì có th m c s khóm c trên cánh ng.

M i khóm c trên b n c ánh d u b ng m t ký t '#' ho c là 2 ký t '#' n m k nhau (trên ng chéo thì không ph i). Cho b n c a cánh ng, ***hãy nói cho Bessie bi t có bao nhiêu khóm c trên cánh ng.***

Ví d nh cánh ng d i đây v i R=5 và C=6:

.#....

..#...

..#..#

...##.

.#....

Cánh ng này có 5 khóm c : M t khóm hàng u tiên, m t khóm t o b i hàng th 2 và th 3 c t th 2, m t khóm là 1 ký t n m riêng r hàng 3, m t khóm t o b i c t th 4 và th 5 hàng 4, và m t khóm cu i cùng hàng 5.

D li u

- Dòng 1: 2 s nguyên cách nhau b i d u cách: R và C



- Dòng 2..R+1: Dòng i+1 mô tả hàng i của cánh rừng với C ký tự, các ký tự là '#' hoặc '.'.

Kết quả

- Dòng 1: Mô tả số nguyên cho biết số lượng nhóm cây trên cánh rừng.

Ví dụ

Dữ liệu

5 6

.#....

..#...

..#..#

...##.

.#....

Kết quả

5

Nhận xét: Số lượng các nhóm cây có thể xem là số vùng liên thông trên đồ thị.

Trong đó, khi $a[i,j]$ là cây và 4 lân cận của nó, nếu có cây thì tính giá trị $a[i,j]$ như sau.

<pre> Uses math; Const fi = 'VBGRASS.INP'; fo = 'VBGRASS.OUT'; MAXN = 200; Var f : array [0..MAXN+1,0..MAXN+1] of boolean; m,n : longint; Res : longint; Procedure Init(); begin Fillchar(f,sizeof(f),false); Res := 0; end; Procedure ReadData(); var i,j : longint; c : char; begin Readln(m,n); for i:=1 to m do begin for j:=1 to n do begin read(c); f[i,j] := c = '#'; end; readln; end; end; </pre>	<pre> Procedure Dfs(x,y: longint); const tx : array [1..4] of longint = (1,-1,0,0); ty : array [1..4] of longint = (0,0,1,-1); var i,u,v: longint; begin f[x,y]:=false; for i:=1 to 4 do begin u:=tx[i] + x; v:=ty[i] + y; if f[u,v] then dfs(u,v); end; end; end; Procedure Solve(); var i,j : longint; begin for i:=1 to m do for j:=1 to n do if f[i,j] then begin dfs(i,j); inc(Res); end; end; end; end; BEGIN assign(input,fi); reset(input); assign(output,fo); rewrite(output); Init(); </pre>
--	--



end;	ReadData(); Solve(); Writeln(Res); close(input); close(output); END.
------	--

Bài Toán: V8ORG

m t t n c n , l c l ng an ninh v a phát hi n m t t ch c i l p. T ch c i l p này c t ch c ch t ch , bao g m m ng l i thành viên và ch huy các c p b c khác nhau. Các thành viên c a t ch c c ánh s t l n N. T ch c có m t ch huy t i cao, luôn c ánh s 1. M i thành viên ch b i t viên ch huy tr c t i p c a mình (có duy nh t m t viên ch huy tr c t i p) ch không b i t các ch huy c p cao h n.

Khi t i n hành vi c b t gi các thành viên, t ch c s b phân rã thành các nhóm nh không liên k t v i nhau, ví d sau khi b t gi thành viên s 2 (hình 1), t ch c b phân rã thành 4 nhóm. L c l ng an ninh kh ng nh, m t nhóm ch a ít h n K thành viên s không còn là m i e d a cho t n c. không làm gì m hình nh c a t n c tr c d lu n qu c t , các nhà lãnh o an ninh mu n b t gi m t s l ng ít nh t ph n t i l p, sao cho các nhóm b phân rã u không còn gây nguy h i cho t n c.

Cho b i t c u trúc c a t ch c i l p, vi c ch ng trình giúp các nhà lãnh o an ninh xác nh s l ng ph n t i l p ít nh t c n b t gi .

D l i u

- Dòng u tiên ch a s nguyên K ($1 \leq K \leq 10000$).
- Dòng th hai ch a s nguyên N ($1 \leq N \leq 10000$).
- Dòng th ba ch a N-1 s nguyên cách nhau b i kho ng tr ng, ch s c a ch huy tr c t i p c a m i ph n t c a t ch c (tr ch huy t i cao): S u tiên cho b i t ch huy c a ph n t th hai, s th hai cho b i t ch huy c a ph n t th ba,...

K t q a

In ra m t s nguyên duy nh t là s ph n t i l p ít nh t c n b t gi .



Ví d

D li u	K t qu	Mô t
<p>3 14 1 1 2 2 3 2 3 6 6 6 7 4 7</p> <p>Hình 1</p>	4	Có th b t gi 4 ph n t 6, 2, 7 và 8.

Ý t ng gi i thu t: G i $s[i]$ là s l ng ph n t d i quy n ch huy c a ph n t i (bao g m c chính i), d th y trong quá trình duy t Dfs, n u có m t ph n t có $s[i] \geq k$ thì ta s “b t gi ” ph n t này, t c là cho $s[i] = 0$, vì c tính các $s[u]$ (v i m i u nh n i là ch huy s c tính tr c khi tính $s[i]$);

<pre>Const fi ='V8ORG.INP'; fo ='V8ORG.OUT'; MAXN = 20000; type link = ^node; node = record v : longint; next: link; end; var a : array [0..MAXN] of link; s : array [0..MAXN] of longint; n,k : longint; Res : longint; Procedure Push(u,v: longint); var p: link; begin new(p); p^.v:=v; p^.next:=a[u]; a[u]:=p; end;</pre>	<pre>Procedure Dfs(x: longint); var p: link; begin p:=a[x]; s[x]:=1; while p<>nil do begin dfs(p^.v); inc(s[x],s[p^.v]); p:=p^.next; end; if s[x] >= k then begin inc(Res); s[x]:=0; end; end; BEGIN assign(input,fi); reset(input); assign(output,fo); rewrite(output); ReadData(); Dfs(1); Write(Res); close(input); close(output); END.</pre>
--	--



<pre> Procedure ReadData(); var i: longint; x : longint; begin Readln(k); Readln(n); for i:=1 to n-1 do begin read(x); push(x,i+1); end; end; </pre>	
---	--

Bài toán: D o ch i ng c

Có N con bò ($1 \leq N \leq 1,000$), thu n t i n ta ánh s t $1 \rightarrow N$, ang n c trên N ng c , thu n t i n ta c ng ánh s các ng c t $1 \rightarrow N$. B i t r ng con bò i ang n c trên ng c i.

M t vài c p ng c c n i v i nhau b i l trong $N-1$ con ng 2 chi u mà các con bò có th i qua. Con ng i n i 2 ng c A_i và B_i ($1 \leq A_i \leq N$; $1 \leq B_i \leq N$) và có dài là L_i ($1 \leq L_i \leq 10,000$).

Các con ng c thi t k sao cho v i 2 ng c b t k u có duy nh t l ng i gi a chúng. Nh v y các con ng này ã hình thành l c u trúc cây.

Các chú bò r t có tinh th n t p th và mu n c th m th ng xuyên. Vì v y l bò mu n b n giúp chúng tính toán dài ng i gi a Q ($1 \leq Q \leq 1,000$) c p ng c (m i c p c mô t là 2 s nguyên p_1, p_2 ($1 \leq p_1 \leq N$; $1 \leq p_2 \leq N$)).

D LI U

- Dòng 1: 2 s nguyên cách nhau b i d u cách: N và Q
- Dòng 2.. N : Dòng $i+1$ ch a 3 s nguyên cách nhau b i d u cách: A_i , B_i , và L_i
- Dòng $N+1$.. $N+Q$: M i dòng ch a 2 s nguyên khác nhau cách nhau b i d u cách mô t 1 yêu c u tính toán dài 2 ng c mà l bò mu n i th m qua l i p_1 và p_2 .

K T QU

- Dòng 1.. Q : Dòng i ch a dài ng i gi a 2 ng c yêu c u th i.

VÍ D

D li u

4 2

2 1 2

4 3 2



1 4 3

1 2

3 2

K t qu

2

7

GI I THÍCH

Yêu c u 1: Con ng gi a ng c 1 và 2 có dài là 2. Yêu c u 2: i qua con ng n i ng c 3 và 4, r i t i p t c i qua con ng n i 4 và 1, và cu i cùng là con ng n i 1 và 2, dài t ng c ng là 7.

Ý t ng gi i thu t: V i m i truy v n (p1,p2) ta th c hi n Dfs b t u t p1, trong quá trình dfs ta l u l i f[i] là dài trên ng i t i n p1, k t qu là f[p2];

<pre> Const fi = 'PWALK.INP'; fo = 'PWALK.OUT'; MAXN = 2000; type link = ^node; node = record v,w : longint; next: link; end; var a : array [0..MAXN] of link; n,q : longint; p1,p2 : longint; l : array [0..MAXN] of longint; free : array [0..MAXN] of boolean; Procedure push(u,v,w: longint); var p: link; begin new(p); p^.v:=v; p^.w:=w; p^.next:=a[u]; a[u]:=p; end; </pre>	<pre> Procedure Dfs(x: longint); var p: link; v : longint; begin free[x] := false; p:=a[x]; while p<>nil do begin v:=p^.v; if free[v] then begin l[v] := l[x] + p^.w; dfs(v); end; p:=p^.next; end; end; Procedure Solve(); begin Fillchar(l,sizeof(l),0); Fillchar(free,sizeof(free),true); Dfs(p1); end; BEGIN assign(input,fi); reset(input); assign(output,fo); rewrite(output); ReadData(); </pre>
--	--



<pre> Procedure ReadData(); var i : longint; u,v,w: longint; begin Readln(n,q); for i:=1 to n-1 do begin readln(u,v,w); push(u,v,w); push(v,u,w); end; end;</pre>	<pre> While q>0 do begin Readln(p1,p2); Solve(); Writeln(l[p2]); dec(q); end; close(input); close(output); END.</pre>
---	--

Bài toán: B o v nông trang

Nông trang có r t nhi u ng n i núi, b o v nông trang nông dân John mu n t ng i canh gác trên các ng n i này.

Anh ta b n kho n không bi t s c n bao nhiêu ng i canh gác n u nh anh ta mu n t l ng i canh gác trên nh c a m i i. Anh ta có b n c a nông trang là m t ma tr n g m N ($1 < N \leq 700$) hàng và M ($1 < M \leq 700$) c t. M i ph n t c a ma tr n là cao H_{ij} so v i m t n c bi n ($0 \leq H_{ij} \leq 10,000$) c a ô (i, j). Hãy giúp anh ta xác nh s l ng nh i trên b n .

nh i là l ho c nhi u ô n m k nhau c a ma tr n có cùng cao c bao quanh b i c nh c a b n ho c b i các ô có cao nh h n. Hai ô g i là k nhau n u chên l ch gi a t a X không quá 1 và chên l ch t a Y không quá 1.

D li u

* Dòng 1: Hai s nguyên cách nhau b i d u cách: N và M

* Dòng 2..N+1: Dòng i+1 mô t hàng i c a ma tr n v i M s nguyên cách nhau b i d u cách: H_{ij}

K t qu

* Dòng 1: M t s nguyên duy nh t là s l ng nh i.

Ví d

D li u:

8 7

4 3 2 2 1 0 1

3 3 3 2 1 0 1

2 2 2 2 1 0 0

2 1 1 1 1 0 0

1 1 0 0 0 1 0

0 0 0 1 1 1 0



0 1 2 2 1 1 0

0 1 1 1 2 1 0

K t qu : 3

Ý t ng gi i thu t : Ta s làm 2 b c:

B c 1: V i m i nh $[i,j]$ ch a th m, ta dfs ánh d u các nh có chi u cao $< a[i,j]$, ta s m b o r ng t nh có chi u cao $a[u,v]$ nào ó, th t c dfs1 s ánh d u nh ng nh có chi u cao $\leq a[u,v]$ l n c n;

Nh v y ch có các nh có chi u cao “ nh” còn l i;

B c 2: Dfs tìm các nhóm nh, công vi c này khá d dàng, cách làm t ng t v i bài VBGRASS.

<pre> Const fi ='NKGUARD.INP'; fo ="; MAXN = 1000; tx : array [1..8] of longint = (1,1,1,-1,-1,0,0); ty : array [1..8] of longint = (-1,0,1,-1,0,1,1,-1); Var a : array [0..MAXN+1,0..MAXN+1] of longint; m,n : longint; Res : longint = 0; free : array [0..MAXN+1,0..MAXN+1] of boolean; Procedure ReadData(); var i,j: longint; begin Readln(m,n); for i:=1 to m do for j:=1 to n do read(a[i,j]); end; end; Procedure Init(); var i: longint; begin Fillchar(free,sizeof(free),true); for i:=0 to m+1 do begin free[i,n+1]:=false; free[i,0]:=false; end; for i:=0 to n+1 do </pre>	<pre> Procedure Dfs1(x,y,s: longint); var i: longint; u,v : longint; begin for i:=1 to 8 do begin u:=x+tx[i]; v:=y+ty[i]; if (free[u,v]) and (a[u,v]<=a[x,y]) and (a[u,v]<s) then begin free[u,v]:=false; Dfs1(u,v,s); end; end; end; end; Procedure Dfs2(x,y: longint); var i: longint; u,v : longint; begin for i:=1 to 8 do begin u:=x+tx[i]; v:=y+ty[i]; if free[u,v] then begin free[u,v]:=false; Dfs2(u,v); end; end; end; end; Procedure Solve(); var i,j : longint; begin for i:=1 to m do for j:=1 to n do </pre>
--	---



<pre> begin free[m+1,i]:=false; free[0,i]:=false end; end;</pre>	<pre> if free[i,j] then Dfs1(i,j,a[i,j]); for i:=1 to m do for j:=1 to n do if free[i,j] then begin Dfs2(i,j); inc(Res); end; end; end; end; BEGIN assign(input,fi); reset(input); assign(output,fo); rewrite(output); ReadData(); Init(); Solve(); WriteLn(Res); close(input); close(output); END.</pre>
--	---

Bài toán: Leo núi

Cho m t b n kích th c $N \times N$ ($2 \leq N \leq 100$), m i ô mang giá tr là cao c a ô ó ($0 \leq \text{cao} \leq 110$). Bác John và bò Bessie ang ô trên trái (dòng 1, c t 1) và mu n i n cabin (dòng N, c t N). H có th i sang ph i, trái, lên trên và xu ng d i nh ng không th i theo ng chéo. Hãy giúp bác John và bò Bessie tìm ng i sao cho chênh l ch gi a i m cao nh t và th p nh t trên ng i là nh nh t.

D li u

- Dòng 1: N
- Dòng 2..N+1: M i dòng ch a N s nguyên, m i s cho bi t cao c a m t ô.

K t qu

M t s nguyên là chênh l ch cao nh nh t.

Ví d

D li u

5

1 1 3 6 8

1 2 2 5 5

4 4 0 3 3

8 0 2 3 4

4 3 0 2 1

K t qu



2

Ý t ng : Do gi i h n chi u cao c a nh i là 200 nên ta s th c hi n tìm ki m nh phân và Dfs;

B t u v i 2 bi n hmin là chi u cao nh nh t s xét, hmax là chi u cao l n nh t s xét, ta duy t hmin t 1 n 200 và dùng hàm ch t nh phân tìm hmax nh nh t sao cho n u o n ng t (1,1) n (n,n) ch có các nh có cao n m trong o n [hmin,hmax]

V i m i c p hmin, hmax tìm c, ta so sánh hi u v i k t qu và c p nh t.

<pre>{Thu t toán : DFS + ch t nh phân} uses Math; Const fi ='MTWALK.INP'; fo ='MTWALK.OUT'; MAXN =200; INF =99999; var a : array [1..MAXN,1..MAXN] of longint; n : longint; res : longint = INF; free : array [0..MAXN,0..MAXN] of boolean; hmin,hmax: longint; Procedure ReadData(); var i,j : longint; begin Readln(n); for i:=1 to n do for j:=1 to n do read(a[i,j]); end; Procedure Dfs(x,y: longint); const tx : array [1..4] of longint = (1,-1,0,0); ty : array [1..4] of longint = (0,0,1,-1); var i,u,v: longint; begin for i:=1 to 4 do begin u:=x+tx[i]; v:=y+ty[i]; if free[u,v] and (a[u,v] >= hmin) and (a[u,v] <= hmax) then begin free[u,v]:=false; Dfs(u,v); end; end; end; end;</pre>	<pre>Function ok():boolean; var i: longint; begin Fillchar(free,sizeof(free),true); for i:=1 to n do begin free[i,0]:=false; free[0,i]:=false; free[i,n+1]:=false; free[n+1,i]:=false; end; if (a[1,1] >= hmin) and (a[1,1] <= hmax) then Dfs(1,1); exit(not(free[n,n])); end; Function f():longint; var u,v,mid: longint; begin u:=hmin; v:=200; while u<v-1 do begin mid:= (u+v) div 2; hmax:=mid; if ok() then v:=mid else u:=mid; end; hmax:=u; if ok() then exit(u-hmin); hmax:=v; if ok() then exit(v-hmin); exit(INF); end; BEGIN assign(input,fi); reset(input); assign(output,fo); rewrite(output); ReadData(); For hmin:=0 to 200 do Res := min(Res,f()); Writeln(Res); close(input); close(output); END.</pre>
---	--



Bài toán: N c l nh

Mùa hè oi Wisconsin ã khi n cho l bờ ph i i tìm n c làm d u i c n khát. Các ng ng d n n c c a nông dân John ã d n n c l nh vào l t p N ($3 \leq N \leq 99999$; N l) nhánh (ánh s t $1..N$) t m t cái b m t chu ng bờ.

Khi n c l nh ch y qua các ng, s c nóng mùa hè s làm n c m lên. Bessie mu n tìm ch có n c l nh nh t cô bò có th t n h ng mùa hè m t cách tho i mái nh t.

Bessie ã v s toàn b các nhánh ng n c và nh n ra r ng nó là m t th d ng cây v i g c là chu ng bờ và các i m nút ng thì có chính xác 2 nhánh con i ra t nút ó. M t i u ng c nhiên là các nhánh ng này u có dài là 1.

Cho b n các ng n c, hãy cho bi t kho ng cách t chu ng bờ t i t t c các nút ng và các ph n cu i ng ng.

“Ph n cu i” c a m t ng ng, có th là i vào m t nút ng ho c là b b t, c g i theo s th t c a ng ng. B n có C ($1 \leq C \leq N$) nút ng, c mô t b ng 3 s nguyên: là “ph n cu i” c a ng E_i ($1 \leq E_i \leq N$) và 2 ng nhánh i ra t ó là $B1_i$ và $B2_i$ ($2 \leq B1_i \leq N$; $2 \leq B2_i \leq N$). ng ng s l n i v i chu ng bờ; kho ng cách t ph n cu i c a ng ng này t i chu ng bờ là 1.

D li u

- Dòng 1: 2 s nguyên cách nhau b i d u cách: N và C
- Dòng 2..C+1: Dòng i+1 mô t nút ng i v i ba S nguyên cách nhau b i d u cách: E_i , $B1_i$, và $B2_i$

K t qu

- Dòng 1..N: Dòng i ch a l s nguyên là kho ng cách t chu ng t i “ph n cu i” c a ng th i.

Ví d

D li u

5 2

3 5 4

1 2 3

Gi i thích:

D li u trên mô t b n ng n c sau:



```

+-----+
| Chu ng |
+-----+
| 1
*
2 /\ 3
*
4 /\ 5

```

K t qu

1
2
2
3
3

Gi i thích:

ng 1 luôn cách chu ng 1 o n là 1. ng 2 và 3 n i v i ng 1 nên kho ng cách s là 2. ng 4 và 5 n i v i ng 3 nên kho ng cách s là 3.

Ý t ng thu t toán: G i $h[i]$ là dài t ng i n chu ng, $r[i]$ là ng ph i c a i và $l[i]$ là ng trái, ta có $h[r[i]] = h[l[i]] = h[i] + 1$;

<pre> Const fi='VCOLDWAT.INP'; fo=''; mxF=100000; mxT=1000; Type Nut=record t,p:longint; end; Var n:longint; h:array [1..mxF] of longint; a:array [1..mxF] of Nut; Procedure Init; Var c,i,e:longint; Begin assign(input,fi); reset(input); </pre>	<pre> procedure DFS(u:longint); begin if u=1 then h[u]:=1; if a[u].t<>0 then begin h[a[u].t]:=h[u]+1; DFS(a[u].t); end; if a[u].p<>0 then begin h[a[u].p]:=h[u]+1; DFS(a[u].p); end; end; procedure GetOut; var i:longint; </pre>
---	--



<pre> readln(n,c); for i:=1 to c do readln(e,a[e].t,a[e].p); close(input); End;</pre>	<pre> begin assign(output,fo); rewrite(output); for i:=1 to n do DFS(i); for i:=1 to n do writeln(h[i]); close(output); end; BEGIN Init; GetOut; END.</pre>
---	--

2. Tìm kiếm theo chi u r ng

Trong phép duy t th BFS, nh xu t phát v ây c ng c th m u tiên, nh ng có khác v i DFS ch là: Sau ó các nh ch a c th m mà là lân c n c a v s c th m k ti p nhau, r i m i n các nh ch a c th m là lân c n l n l t c a các nh này và c t ng t nh v y. Sau ây là gi i thu t BFS:

Procedure BFS(v)

- 1) Visited(v) :=1; //Visited dùng ánh d u các nh ã c th m
- 2) Kh i t o queue v i v ã c n p vào
- 3) **While** Q không r ng **Do**

Begin

Call pop(v,Q); //L y nh v ra kh i Q

For m i ình w lân c n v i v **Do**

if Visited(w)=0 **then**

Begin

Call push(w,Q);

Visited(w) :=1;

End:

M i nh c th m s c n p vào queue ch m t l n v v y câu l nh while l p l i nh i u nh t n l n. N u G c bi u di n b i ma tr n lân c n thì câu l nh For s chi phí $O(n)$ th i gian i v i m i nh, do ó th i gian chi phí toàn



b s là $O(n^2)$. Còn tr ng h p G c bi u di n v i danh sách lân c n thì chi phí t ng quát chung là $O(e)$.

hi u rõ h n v BFS ta nghiên c u các toán sau:

Bài toán: G m c

Bessie r t yêu b i c c a mình và thích thú ch y v chu ng bò vào gi v t s a bu i t i. Bessie ã chia ng c c a mình là 1 vùng hình ch nh t thành các ô vuông nh v i R ($1 \leq R \leq 100$) hàng và C ($1 \leq C \leq 100$) c t, ng th i ánh d u ch nào là c và ch nào là á. Bessie ng v trí R_b, C_b và mu n n c theo cách c a mình, t ng ô vuông m t và tr v chu ng ô 1,1 ; bên c nh ó ng i này ph i là ng n nh t. Bessie có th i t 1 ô vuông sang 4 ô vuông khác k c nh.

D i ây là m t b n ví d [v i á (*), c (.), chu ng bò ('B'), và Bessie ('C')] hàng 5, c t 6] và m t b n cho bi t hành trình t i u c a Bessie, ng i c đánh d u b ng ch 'm'.

B n	ng i t i u
1 2 3 4 5 6 <-c t	1 2 3 4 5 6 <-c t
1 B . . . * .	1 B m m m * .
2 . . * . . .	2 . . * m m m
3 . * * . * .	3 . * * . * m
4 . . * * * .	4 . . * * * m
5 * . . * . C	5 * . . * . m

Bessie n c 9 ô c .

Cho b n , hãy tính xem có bao nhiêu ô c mà Bessie s n c trên con ng ng n nh t tr v chu ng (t t nhiên trong chu ng không có c âu nên ng có tính nhé)

D li u

- Dòng 1: 2 s nguyên cách nhau b i d u cách: R và C
- Dòng 2..R+1: Dòng i+1 mô t dòng i v i C ký t (và không có d u cách) nh ã nói trên.

K t qu

- Dòng 1: M t s nguyên là s ô c mà Bessie n c trên hành trình ng n nh t tr v chu ng.

Ví d

D li u

5 6



B...*.

..*...

***.

..***.

...C

K t qu

9

Ý t ng : Bfs b t u t nh B, v i b ng f[i,j] là dài ng i ng n nh t t
nh (i,j) n nh B, k t qu là f[cx,cy];

<pre> Const fi ='VMUNCH.inp'; fo ="; MAXN =1500; tx : array [1..4] of longint = (1,0,-1,0); ty : array [1..4] of longint = (0,1,0,-1); Type pos=record x,y : longint; end; Var a : array [0..MAXN,0..MAXN] of boolean; d : array [1..MAXN,1..MAXN] of longint; q : array [1..MAXN*MAXN] of pos; r,c,cx,cy : longint; Procedure nhap; var i,j : longint; t : char; begin assign(input,fi);reset(input); fillchar(a,sizeof(a),false); readln(r,c); for i:=1 to r do begin for j:=1 to c do begin read(t);a[i,j]:=t='.'; d[i,j]:=1; if t='B' then begin q[1].x:=i; q[1].y:=j; end; if t='C' then begin a[i,j]:=true; </pre>	<pre> Procedure xuly; var bot,top,x,y,i : longint; u : pos; begin bot:=1;top:=1; repeat u:=q[top];inc(top); a[u.x,u.y]:=false; for i:=1 to 4 do begin x:=u.x+tx[i];y:=u.y+ty[i]; if a[x,y] then begin a[x,y]:=false; inc(bot); q[bot].x:=x; q[bot].y:=y; d[x,y]:=d[u.x,u.y]+1; end; end; until top>bot; end; Procedure xuat; begin assign(output,fo);rewrite(output); writeln(d[cx,cy]-1); close(output); end; BEGIN NHAP; XULY; XUAT; END. </pre>
---	---

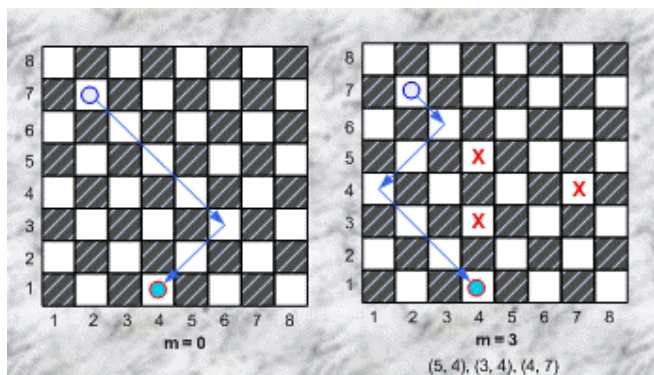


<pre> cx:=i; cy:=j; end; end; readln; end; close(input); end; </pre>	
--	--

Bài toán: VOI06 Quân t ng

Xét bàn c vuông kích th $c \times n$. Các dòng c ánh s t 1 n n, t d i lên trên. Các c t c ánh s t 1 n n t trái qua ph i.

Ô n m trên giao c a dòng i và c t j c g i là ô (i,j). Trên bàn c có m (0 m n) quân c . V i m > 0, quân c th i ô (r_i, c_i) , $i = 1, 2, \dots, m$. Không có hai quân c nào trên cùng m t ô. Trong s các ô còn l i c a bàn c , t i ô (p, q) có m t quân t ng. M i m t n c i, t v trí ang ng quân t ng ch có th di chuy n n c nh ng ô trên cùng ng chéo v i nó mà trên ng i không ph i qua các ô ã có quân



C n ph i a quân t ng t ô xu t phát (p, q) v ô ích (s,t). Gi thi t là ô ích không có quân c . N u ngoài quân t ng không có quân nào khác trên bàn c thì ch có 2 tr ng h p: ho c là không th t i c ô ích, ho c là t i c sau không quá 2 n c i (hình trái). Khi trên bàn c còn có các quân c khác, v n s không còn n g i n nh v y.

Yêu c u: Cho kích th c bàn c n, s quân c hi n có trên bàn c m và v trí c a chúng, ô xu t phát và ô ích c a quân t ng. Hãy xác nh s n c i ít nh t c n th c hi n a quân t ng v ô ích ho c a ra s -1 n u i u này không th th c hi n c.

Input

Dòng u tiên ch a 6 s nguyên n, m, p, q, s, t.



N u m > 0 thì m i dòng th i trong m dòng ti p theo ch a m t c p s nguyên ri , ci xác nh v trí quân th i.

Hai s liên ti p trên cùng m t dòng c ghi cách nhau ít nh t m t d u cách.

Output

G m l dòng duy nh t là s n c i tìm c

Example

Input:

8 3 7 2 1 4

5 4

3 4

4 7

Output:

3

H n ch :

Trong t t c các test: 1 n 200. Có 60% s l ng test v i n 20.

Ý t ng: gi ng nh bài VMUNCH, ch khác nhau cách thêm nh vào trong queue.

<pre> Const fi ='QBBISHOP'; fo =""; MAXN =300; tx : array [1..4] of longint = (1,1,-1,-1); ty : array [1..4] of longint = (1,-1,1,-1); Type pos = record x,y : longint; end; Var a : array [0..MAXN,0..MAXN] of longint; free,tick : array [0..MAXN,0..MAXN] of boolean; queue : array [1..MAXN*MAXN] of pos; n,s,t,p,q : longint; Procedure nhap; var i,u,v,m : longint; begin fillchar(free,sizeof(free),true); tick:=free; readln(n,m,p,q,s,t); for i:=0 to n+1 do begin tick[0,i]:=false; tick[i,0]:=false; tick[n+1,i]:=false; tick[i,n+1]:=false; end; for i:=1 to m do begin readln(u,v); tick[u,v]:=false; end; </pre>	<pre> Procedure BFS; var d,c,i,k : longint; u,v : pos; begin d:=1;c:=1; queue[d].x:=p; queue[d].y:=q; free[p,q]:=false; repeat u:=queue[d].inc(d); for i:=1 to 4 do begin k:=1; while (tick[u.x+k*tx[i],u.y+k*ty[i]]) do begin if free[u.x+k*tx[i],u.y+k*ty[i]] then begin free[u.x+k*tx[i],u.y+k*ty[i]]:=false; a[u.x+k*tx[i],u.y+k*ty[i]]:=a[u.x,u.y]+1; inc(c); queue[c].x:=u.x+k*tx[i]; queue[c].y:=u.y+k*ty[i]; end; inc(k); end; until d>c; end; Procedure xuất; begin if (s=p) and (t=q) then writeln(0) else if a[s,t]=0 then writeln(-1) else writeln(a[s,t]); </pre>
---	--



end;	end; BEGIN assign(input,fi);reset(input); assign(output,fo);rewrite(output); NHAP; BFS; XUAT; close(input); close(output); END.
------	--

Bài toán: Laser Phones

FJ mua m t h th ng liên l c m i cho àn bò chúng có th trò chuy n v i nhau trong khi n c . ng c c mô t b ng m t l i hình ch nh tkích th c $W \times H$ ($1 \leq W \leq 100$; $1 \leq H \leq 100$).

Hi n t i FJ m i cung c p i n tho i cho 2 con bò u àn. Tuy nhiên v n là liên l c gi a hai con bò ch th c hi n c n u ng truy n thông tin gi a chúng không b ch n. ây, thông tin ch c truy n theo các ng th ng và d ng l i n u nó b ch n b i nút á, cây to (kí hi u b ng các kí t '*').

Do ó, FJ ph i mua thêm m t s g ng (kí hi u b ng các kí t '/' và '\') i h ng ng i c a tia laser. Xét ví d minh h a d i ây:

Kích th c c a ng có là 8×7 , $H = 8$ và $W = 7$. Hai con bò u àn c kí hi u là 'C', á và cây to kí hi u là '*':

7.....	7.....
6.....C	6...../-C
5.....*	5..... *
4*****.*	4***** *
3....*..	3....* .
2....*..	2....* .
1.C..*..	1.C..* .
0.....	0.\-----/.
0 1 2 3 4 5 6	0 1 2 3 4 5 6

C n xác nh M - s l ng g ng ít nh t FJ c n mua có th m b o liên l c gi a hai con bò nói trên. D li u luôn m b o có ít nh t m t cách th c hi n.

INPUT

* Dòng 1: Ch a 2 s nguyên W và H cách nhau ít nh t 1 kí t .



* Dòng 2..H+1: Mô t cánh ng, m i dòng g m W kí t 'C' ho c '*', và '.'.
Thông tin không b ch n khi i qua các kí t '.' và ch có 2 ch 'C'.

Ví d :

7 8

.....

.....C

.....*

*****.*

....*..

....*..

.C..*..

.....

OUTPUT

* Dòng 1: M t s nguyên duy nh t ghi giá tr M - s g ng ít nh t c n mua.

Ví d :

3

Ý t ng: Gi ng nh bài QBBISHOP, ch khác cách thêm nh.

<pre>{ \$R+ } Const fi = ";//MLASERP.INP"; fo = " "; MAXN = 200; Var a,free : array [0..MAXN,0..MAXN] of boolean; f : array [0..MAXN,0..MAXN] of longint; queuex : array [0..MAXN*MAXN] of longint; queuey : array [0..MAXN*MAXN] of longint; cx,cy : array [1..2] of longint; n,m : longint; kq : longint; Procedure Nhap; var i,j,l : longint; c : char; begin Readln(n,m); l:=0;</pre>	<pre>Procedure Xuly; Const tx : array [1..4] of longint = (1,0,-1,0); ty : array [1..4] of longint = (0,1,0,-1); var d,c : longint; u,v,x,y,i : longint; begin d:=1; c:=1; a[cx[1],cy[1]]:=false; queuex[1]:=cx[1]; queuey[1]:=cy[1]; Repeat x:=queuex[d]; y:=queuey[d]; Inc(d); for i:=1 to 4 do begin u:=x+tx[i]; v:=y+ty[i]; While a[u,v] do begin if free[u,v] then begin f[u,v]:=f[x,y]+1; free[u,v]:=false;</pre>
---	--



<pre> for i:=1 to m do begin for j:=1 to n do begin Read(c); a[i,j] := c<>'*'; if upcase(c) = 'C' then begin Inc(l); cx[l]:=i; cy[l]:=j; end; end; end; Readln; end; end; Procedure KhoiTao; var i : longint; begin for i:=0 to m+1 do a[i,0]:=false; for i:=0 to m+1 do a[i,n+1]:=false; for i:=0 to n+1 do a[0,i]:=false; for i:=0 to n+1 do a[m+1,i]:=false; Fillchar(f,sizeof(f),0); Fillchar(free,sizeof(free),true); end; </pre>	<pre> Inc(c); QueueX[c]:=u; QueueY[c]:=v; end else begin if f[u,v]>f[x,y]+1 then f[u,v]:=f[x,y]+1; end; u:=u+tx[i]; v:=v+ty[i]; end; end; until d>c; Kq:=f[cx[2],cy[2]]-1; if (cx[2] = 0) and (cy[2] = 0) then kq:=0; end; BEGIN assign(input,fi); reset(input); assign(output,fo); rewrite(output); Nhap; KhoiTao; Xuly; Writeln(Kq); close(input); close(output); END. </pre>
---	---

Vì c n m v ng c ph ng pháp và cài t c thu t toán tìm ki m theo chi u r ng (DFS) và tìm ki m theo chi u sâu (BFS) là nh ng n i dung, k n ng quan tr ng i v i h c sinh trong i tuy n Tin h c. Tôi hi v ng, tham lu n này tr thành ngu n tài li u nh bé có ích trong vô vàn ngu n tài li u ã có h ng d n h c n i dung th . Tôi mong nh n c ý ki n óng góp c a các th y, cô tham lu n hoàn thi n h n.

Tác gi : *ng Tu n Thành*



M T S BÀI TOÁN V CÂY KHUNG NH NH T

Lê Th H i H ng

Tr ường THPT Chuyên Biên Hòa - Hà
Nam

Bài toán cây khung nh ỏ nh t là m t trong nh ững bài toán t i u thu c ph n lý thuy t ết . Nh ững chúng ta bi t, có 2 thu t toán ết i quy t bài toán này, ó là thu t toán Prim và thu t toán Kruskal, trong cu n *Tài li u Giáo khoa chuyên Tin (Quy n 2)* ã trình bày r t k thu t toán, h ướng d n cách cài t c ết và ánh giá ết c t p tính toán. Trong bài vi t này, tôi xin ả ra m t s bài t p áp d ng thu t toán.

i toán 1: Vòng ủa F1- Mã bài: NKRACING

Singapore s t ch c m t cu c ủa xe Công Th c 1 vào n m 2008. Tr c khi cu c ủa đi n ra, ã xu t hi n m t s cu c ủa v ẽm trái lu t. Chính quy n mu n thi t k m t h ệ th ng ki m soát giao thông ết t ết các tay ủa ph m lu t. H ệ th ng bao g m m t s camera t trên các tuy n ường khác nhau. ết m b o tính hi u qu cho h ệ th ng, c n có ít nh t m t camera d c theo m i vòng ủa.

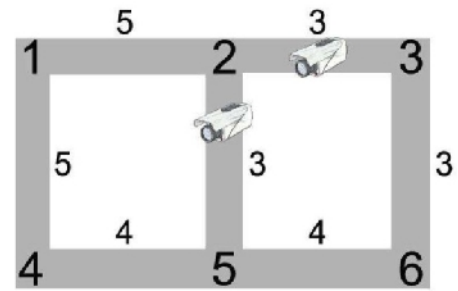
H ệ th ng ường Singapore có th c mô t b i m t dãy các nút giao thông và các ường n i hai chi u (xem hình v). M t vòng ủa bao g m m t nút giao thông xu t phát, ti p theo là ường i bao g m ít nh t 3 tuy n ường và cu i cùng quay tr l i i m xu t phát. Trong m t vòng ủa, m i tuy n ường ch c i qua úng m t l n, theo úng m t h ướng.

Chi phí t camera ph thu c vào tuy n ường c ch n. Các s nh trong hình v cho bi t chi phí t camera lên các tuy n ường. Các s l n xác nh các nút giao thông. Camera c t trên các tuy n ường ch không ph i t i các nút giao thông. B n c n ch n m t s tuy n ường sao cho chi phí l p t là th p nh t ể th i v n m b o có ít nh t m t camera d c theo m i vòng ủa.

Vi t ch ường trính tìm cách t các camera theo dõi giao thông sao cho t ổng chi phí l p t là th p nh t.

D li u

- Dòng ủa tiên ch ả 2 s nguyên n, m (1 n 10000, 1 m 100000) là



s nút giao thông và s ng n i. Các nút giao thông c ánh s t l n n.

- m dòng ti p theo mô t các ng n i, m i dòng bao g m 3 s nguyên d ng cho bi t hai u nút c a tuy n ng và chi phí l p t camera. Chi phí l p t thu c ph m vi $[1, 1000]$.

K t qu

In ra l s nguyên duy nh t là t ng chi phí l p t th t nh t tìm c.

Ví d

D li u:

6 7

1 2 5

2 3 3

1 4 5

4 5 4

5 6 4

6 3 3

5 2 3 *K t qu*

6

Thu t toán:

Ban u ta gi s ã t camera m i tuy n ng, nh v y c n tìm cách b i m t s các camera v i t ng chi phí gi m c là l n nh t.

T p h p các tuy n ng b i không c ch a chu trình vì n u ch a s t o ra m t vòng ua không c giám sát, suy ra ch có th b i nhi u nh t là $n-1$ camera $n-1$ tuy n ng và $n-1$ tuy n ng ó là m t cây khung c a th .

gi m c nhi u chi phí nh t thì c n tìm cây khung l n nh t c a th b camera trên các c nh c a cây khung ó.



Ch ng trình:

```
{ $mode objfpc }
const
fi='nkracing.inp';
fo='nkracing.out';
max=10000;
maxm=100000;
vc=100000000;
var f:text;
n,m,kq:longint;
x,y,c:array[0..maxm+1]of longint;
{a,ts:array[0..maxm*2+1]of longint;}
goc:array[0..max+1]of longint;
chon:array[0..maxm+1]of longint;
dd:array[0..max+1]of boolean;
procedure doc;
var i,j:longint;
begin
assign(f,fi);
reset(f);
readln(f,n,m);
kq:=0;
for i:=1 to m do
begin
read(f,x[i],y[i],c[i]);
kq:=kq+c[i];
end;
close(f);
end;
procedure viet;
var i,j:longint;
begin
assign(f,fo);
rewrite(f);
writeln(f,kq);
close(f);
end;
function laygoc(u:longint):longint;
begin
while goc[u]<>-1 do
u:=goc[u];
laygoc:=u;
end;
procedure doi(var i,j:longint);
var tg:longint;
begin
```



```
        tg:=i;
        i:=j;
        j:=tg;
    end;
procedure sort(dl,c1:longint);
    var i,j,gt:longint;
    begin
        if dl>=c1 then exit;
        i:=dl;
        j:=c1;
        gt:=c[(c1+dl)div 2];
        repeat
            while c[i]>gt do inc(i);
            while c[j]<gt do dec(j);
            if i<=j then
                begin
                    if i<j then
                        begin
                            doi(x[i],x[j]);
                            doi(y[i],y[j]);
                            doi(c[i],c[j]);
                        end;
                    dec(j);
                    inc(i);
                end;
            until i>j;
            sort(dl,j);
            sort(i,c1);
        end;
end;

procedure lam;
    var i,j,dem,u,v,i1,j1,p:longint;
    begin
        for i:=0 to n do
            goc[i]:=-1;
        sort(1,m);
        dem:=0;
        for i:=1 to m do
            begin
                u:=laygoc(x[i]);
                v:=laygoc(y[i]);
                if u<>v then
                    begin
                        inc(dem);
                        goc[u]:=x[i];
                        kq:=kq-c[i];
                        goc[x[i]]:=y[i];
                    end;
            end;
    end;
```



```

        chon[dem]:=i;
        if dem=n-1 then break;
        end;
    end;

end;

BEGIN
doc;
lam;
viet;
END.

```

Đề toán 2: Xây dựng thành phố - Mã bài: NKCITY

Nhà Anpha đang lập kế hoạch xây dựng một thành phố mới và hiện đại. Theo kế hoạch, thành phố sẽ có N vị trí quan trọng, được gọi là N trung tâm và các trung tâm này sẽ ảnh hưởng tới vị trí của N . Bộ giao thông đã lập ra một danh sách M tuyến đường hai chiều có thể xây dựng để nối hai trung tâm nào đó. Mỗi tuyến đường có một thời gian hoàn thành khác nhau.

Các tuyến đường phải được xây dựng sao cho N trung tâm liên thông với nhau. Nói cách khác, nối hai trung tâm bất kỳ cần phải đi chuyển qua ít nhất một tuyến đường. Bộ giao thông sẽ chọn ra một số tuyến đường trong danh sách ban đầu để xây dựng sao cho chi phí xây dựng là nhỏ nhất.

Do nhà nước ưu tiên đầu tư chính phủ, bộ giao thông sẽ thuê hẳn một đội thi công riêng cho mỗi tuyến đường cần xây dựng. Do đó, thời gian hoàn thành toàn bộ các tuyến đường cần xây dựng sẽ bằng thời gian lâu nhất hoàn thành một tuyến đường nào đó.

Yêu cầu: Giúp bộ giao thông tính thời gian hoàn thành các tuyến đường nhỏ nhất thỏa mãn yêu cầu đã nêu.

Dữ liệu

Dòng đầu tiên chứa N và M ($1 \leq N \leq 1000$; $1 \leq M \leq 10000$).

M tiếp theo, mỗi dòng chứa ba số nguyên u , v và t cho biết có thể xây dựng tuyến đường nối giữa trung tâm u và trung tâm v trong thời gian t . Không có hai tuyến đường nào nối cùng một cặp trung tâm.

Kết quả

Một số nguyên duy nhất là thời gian nhỏ nhất hoàn thành các tuyến đường thỏa mãn yêu cầu đã nêu.



Ví dụ

Dãy

5 7

1 2 2

1 5 1

2 5 1

1 4 3

1 3 2

5 3 2

3 4 4

Kết quả

Thuật toán:

Bài là tìm ra cây khung có chi phí nhỏ nhất là chi phí và chi phí của các chi phí nhỏ nhất đó, tuy nhiên tôi nghĩ rằng mọi cây khung nếu là chi phí nhỏ nhất thì chi phí nhỏ nhất của nó cũng là chi phí nhỏ nhất trong số các chi phí nhỏ nhất của các cây khung.

Vì vậy, tôi dùng thuật toán Kruskal tìm cây khung chi phí nhỏ nhất áp dụng cho bài toán này, cụ thể cùng với thêm vào là chi phí nhỏ nhất của cây khung.

Chương trình:

```
{ $mode objfpc }
const
  fi='nkcity.inp';
  fo='nkcity.out';
  max=1000;
  maxm=10000;
  vc=1000000000;
  var f:text;
  n,m,kq1,kq2:longint;
  x,y,c:array[0..maxm+1] of longint;
  {a,ts:array[0..maxm*2+1] of longint;}
  goc:array[0..max+1] of longint;
  chon:array[0..maxm+1] of longint;
  dd:array[0..max+1] of boolean;
  procedure doc;
    var i,j:longint;
    begin
      assign(f,fi);
      reset(f);
```



```
    readln(f,n,m);
    for i:=1 to m do
        begin
            read(f,x[i],y[i],c[i]);
        end;
    close(f);
end;
procedure viet;
var i,j:longint;
begin
    assign(f,fo);
    rewrite(f);
    writeln(f,kq1);
    close(f);
end;
function laygoc(u:longint):longint;
begin
    while goc[u]<>-1 do
        u:=goc[u];
    laygoc:=u;
end;
procedure doi(var i,j:longint);
var tg:longint;
begin
    tg:=i;
    i:=j;
    j:=tg;
end;
procedure sort(dl,cl:longint);
var i,j,gt:longint;
begin
    if dl>=cl then exit;
    i:=dl;
    j:=cl;
    gt:=c[(cl+dl)div 2];
    repeat
        while c[i]<gt do inc(i);
        while c[j]>gt do dec(j);
        if i<=j then
            begin
                if i<j then
                    begin
                        doi(x[i],x[j]);
                        doi(y[i],y[j]);
                        doi(c[i],c[j]);
                    end;
            end;
    until i>j;
end;
```




```
        dec(j);
        inc(i);
    end;
until i>j;
sort(d1,j);
sort(i,c1);
end;

procedure lam;
var i,j,dem,u,v,i1,j1,p:longint;
begin
    for i:=0 to n do
        goc[i]:=-1;
    sort(1,m);
    kq1:=0;
    dem:=0;
    for i:=1 to m do
        begin
            u:=laygoc(x[i]);
            v:=laygoc(y[i]);
            if u<>v then
                begin
                    inc(dem);
                    kq1:=c[i];
                    goc[u]:=x[i];
                    goc[x[i]]:=y[i];
                    chon[dem]:=i;
                    if dem=n-1 then break;
                end;
            end;
        end;

    end;
BEGIN
doc;
lam;
viet;
END.
```

Bài toán 3: Mạng truy n thông - Mã bài: COMNET (thi HSG QG 2013)

T ng công ty Z g m N công ty con, ánh s t 1-N. M i công ty con có m t máy ch . m b o truy n tin gi a các công ty, Z thuê M ng truy n tin k t n i N máy ch thành m t m ng máy tính c a T ng công ty. Không có 2 ng truy n n i cùng 1 c p máy ch . ng truy n i n i máy ch c a 2 công ty u_i, v_i có chi phí là w_i . M ng máy tính có tính *thông su t*, ngh a là t



m t máy ch có th truy n tin n m t máy ch b t kì khác b ng ng truy n tr c ti p ho c qua nhi u ng trung gian.

M t ng truy n g i là *không ti m n* ng n u nh : m t m t, vì c lo i b ng truy n này không làm m t tính thông su t; m t khác, nó ph i có tính không ti m n ng, ngh a là không thu c b t c m ng con thông su t g m N máy ch và N-1 ng truy n tin v i t ng chi phí thuê bao nh nh t nào c a m ng máy tính.

Trong th i gian t i, chi phí thuê bao c a m t s ng truy n tin thay i. T ng công ty mu n xác nh v i chi phí m i thì ng truy n th k có là ng không ti m n ng hay không xem xét ch m d t vì c thuê ng truy n này.

Yêu c u: Cho Q gi nh, m i gi nh cho bi t danh sách các ng truy n tin v i chi phí thuê m i và ch s k. V i m i gi nh v chi phí m i thuê ng truy n tin, hãy xác nh ng truy n tin th k có là ng truy n tin không ti m n ng trong m ng không.

Input

- Dòng u là T – s testcase. T nhóm dòng, m i nhóm cho thông tin v m t testcase.
- Dòng th nh t g m 3 s nguyên d ng N, M, Q ($Q \leq 30$).
- Dòng th i trong M dòng ti p theo ch a 3 s nguyên d ng u_i, v_i, w_i ($u_i, v_i, w_i < 10^9$).
- Dòng th j trong Q dòng ti p theo mô t gi nh th j:
 - S u tiên là ch s k_j c a ng truy n tin c n xem xét
 - Ti p theo là s_j ($s_j \leq 100$) cho bi t s l ng ng truy n có chi phí thuê m i
 - Cu i cùng là s_j c p s nguyên d ng t_p, c_p cho bi t ng truy n th t_p có chi phí thuê m i là c_p ($c_p < 10^9$).

Output

- G m T nhóm dòng, m i nhóm g m Q dòng. M i dòng là câu tr l i cho gi nh t ng ng trong input. Ghi YES n u câu tr l i là kh ng nh và NO trong tr ng h p ng c l i.

Example

<i>Input:</i>	<i>Output:</i>
---------------	----------------



1	NO
3 3 2	YES
1 2 1	
1 3 2	
2 3 3	
3 2 2 4 3 4	
1 1 1 4	

Giới hạn

- 30% s test u có 1 $N \leq 100$;
- 30% s test t p theo có 1 $N \leq 10^4$ và 1 $M \leq 10^5$;
- 40% s test còn lại có 1 $N \leq 10^5$ và 1 $M \leq 10^6$.

Thuật toán:

Ta tóm tắt bài nh sau: Cho th vô h ng N nh M c nh và Q truy v n. M i truy v n yêu c u thay i tr ng s S c nh c a th và h i xem c nh K có thu c m i cây khung nh nh t c a th hay không.

Nh n th y, n u sau khi b c nh K kh i th ta không tìm c cây khung ho c tìm c cây khung nh nh t nh ng có tr ng s l n h n ban u thì K s là c nh n m trên m i cây khung nh nh t. ph c t p $O(Q \times \text{ph c t p tìm cây khung nh nh t})$.

30% s test u: cài t thu t toán Prim ho c Kruskal thông th ng.

30% s test t p theo, ta c i t n thu t toán Prim s d ng c u trúc d li u Heap có ph c t p $O(Q \times N \log N)$, ho c dùng thu t toán Kruskal v i c u trúc d li u Disjoint-set forest- ph c t p $O(Q \times (O(M \log M) + O(N)))$, trong ó $O(M \log M)$ là chi phí s p x p M c nh và $O(N)$ là chi phí qu n lý Disjoint-set forest.

t 100% s test ta c ng dùng dùng thu t toán Kruskal v i c u trúc d li u Disjoint-set forest, duy t h t các c nh có tr ng s nh h n c nh K , khi duy t n c nh (u, v) thì ta h p t p ch a c nh u và t p ch a c nh v l i, Cu i cùng c nh K là c nh t i m n ng n u nó n i hai t p r i nhau.

Ch ng trình:

```
Program comnet;
const
  fi='comnet.inp';
  fo='comnet.out';
  mn=100000+100;
```



```
mm=1000000+1000;
type
  tedge=record
    u,v,w:longint;
  end;
Var
  edge:array[0..mm] of tedge;
  tmp:array[0..mm] of tedge;
  p:array[0..mn] of longint;
  n,m,q:longint;
  ntest:longint;
Function getRoot(u:longint):Longint;
begin
  if p[u]=u then exit(u);
  p[u]:=getRoot(p[u]);
  exit(p[u]);
end;
procedure union(u,v:longint);
begin
  u:=getRoot(u);
  v:=getRoot(v);
  if u=v then exit;
  p[u]:=v;
end;
procedure solve;
var i,k,s,t,c:longint;
begin
  readln(n,m,q);
  for i:=1 to m do
    with edge[i] do
      readln(u,v,w);
  while q>0 do
    begin
      dec(q);
      // dung mang tmp de luu trong so cac canh ban dau
      for i:=1 to m do
        tmp[i]:=edge[i];
      read(k,s);
      // thay doi s canh teo truy van
      for i:=1 to s do
        begin
          read(t,c);
          tmp[t].w:=c;
        end;
      //khoi tao disjoint set
      for i:=1 to n do
        p[i]:=i;
      //duyet qua cac canh co trong so nho hon canh K
```



```
for i:=1 to m do
  with tmp[i] do
    if w<tmp[k].w then union(u,v);
  // thu xem canh k co noi 2 dinh thuoc 2 tap roi nhau hay khong
  with tmp[k] do
    begin
      if getRoot(u)<>getRoot(v) then writeln('YES')
      else writeln('NO');
    end;
  end;
end;
end;

begin{mai}
  assign(input,fi);
  reset(input);
  assign(output,fo);
  rewrite(output);
  readln(ntest);
  while ntest>0 do
    begin
      dec(ntest);
      solve;
    end;
  end.
end.
```



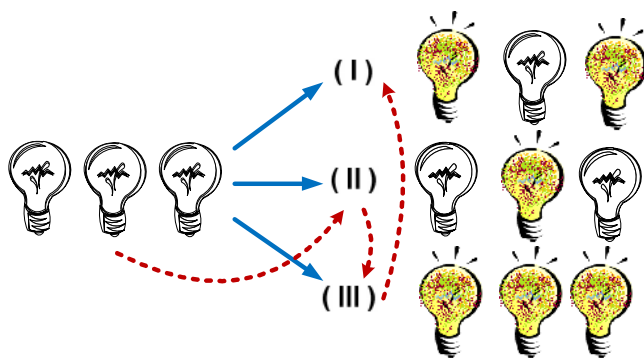
Rèn luyện kỹ năng duy trì trong thi

Nguyễn Quang Minh, trường THPT chuyên H Long

Rèn luyện kỹ năng duy trì trong thi, tôi xin giới thiệu cách giải một vài bài toán sau đây:

Bài 1 RÈN TRANG TRÍ

Rôn mua một bộ đèn trang trí gồm n đèn ($1 \leq n \leq 1000$). Mỗi đèn có một công tắc bật hay tắt riêng đèn đó. Mỗi giây Rôn có thể bật hoặc tắt một bóng đèn tùy chọn. Ban đầu tất cả các bóng đèn đều tắt. Mục tiêu của Rôn là trạng thái khi mà tất cả đèn nào đó bật sáng, những đèn còn lại tắt. Rôn thích một số cấu hình vì chúng có vẻ phù hợp với khung cảnh phòng của Rôn. Mỗi trạng thái của bộ đèn được biểu diễn bằng một chuỗi n ký tự thuộc tập $\{0, 1\}$. Ký tự thứ i xác định trạng thái đèn thứ i , 0 tắt, 1 bật. Ví dụ, với $n = 3$ và Rôn thích 3 cấu hình $\{1, 0, 1\}$, $\{0, 1, 0\}$, $\{1, 1, 1\}$. Kiểm tra xem cấu hình nào là thích hợp nhất Rôn phải làm việc để tắt tất cả đèn. Trong trường hợp này Rôn cần 4 giây để xem xét hết tất cả các cấu hình.



Yêu cầu: Cho biết n và m , trong đó $m - s$ cấu hình

khác nhau mà Rôn thích (1 ≤ m ≤ 15). Hãy xác định thời gian tối thiểu cần thiết để kiểm tra tất cả các trạng thái mà Rôn quan tâm.

Dữ liệu: Vào tệp file văn bản GARLAN.INP:

- Dòng đầu tiên chứa 2 số nguyên n và m ,
- Mỗi dòng trong m dòng tiếp theo chứa chuỗi n ký tự xác định một cấu hình Rôn yêu thích.

Kết quả: Tạo ra file văn bản GARLAN.OUT một số nguyên – thời gian tối thiểu để kiểm tra các cấu hình.

Ví dụ:

GARLAN.INP
3 3
101
010
111

GARLAN.OUT
4

Lưu ý: - Mỗi trạng thái coi như 1 nhà cửa (trạng thái ban đầu là nhà số 0)

- Trong số các nhà là chi phí chuyển trạng thái sang trạng thái kia
- Bài toán trở thành tìm ngắn nhất 0 qua tất cả các nhà với trạng thái ban đầu



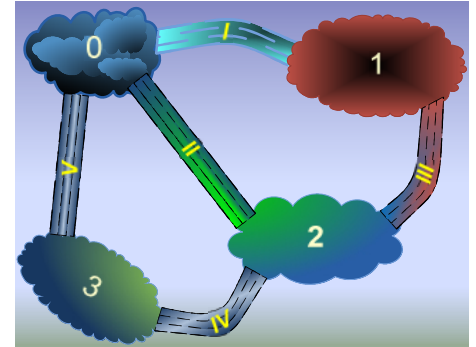
- G i m i cạnh là (u,v,w) , trong ó u,v là 2 nh , w là tr ng s .

Trong ví d trên ta có các c nh $(0,1,2)$, $(0,2,1)$, $(0,3,3)$, $(1,2,3)$, $(1,3,1)$, $(2,3,2)$

Ta c ng ít tnh t là 0-2-1-3 (ho c 0-2-3-1) v i chi phí = 4.

Bài 2 B C C U

Chính ph qu c o Oceani quy t nh xây d ng m chi c c u n i n o c a mình, t o m t m ng l i giao thông ng b cho phép ít d o b t k t i o khác b ng ng b (tr c ti p ho c qua m t s o trung gian). M i cây c u s n i 2 o khác nhau và cho phép i l i hai chi u. Các o c ánh s t 0 n $n-1$. B h n ch b i kinh phí và ngu n nhân l c, ng i ta quy t nh s xây d ng l n l t t ng chi c c u m t và lên k ho ch xác nh c u và trình t xây. M i cây c u c xác nh b i c p o u, v mà nó n i. Trong quá trình th c h i k ho ch có th n m t lúc nào ó t m t o ã có th i n b t k o khác b ng ng b .



Ví d , Oceani có 4 o và ng i ta quy t nh xây d ng 5 c u theo trình t l n l t là 0 – 1, 0 – 2, 1 – 2, 2 – 3, 3 – 0. Tuy v y, không c n ch i n khi hoàn thành k ho ch xây c u, sau khi c u th 4 c xây xong t t c các o ã c n i l n b ng ng b .

Yêu cầu: Cho n, m và các cây c u d i ki n xây. Thông tin v các cây c u a ra theo úng trình t xây d ng. Hãy xác nh s c u t i thi u c n xây theo k ho ch t m t o ã có th i n b t k o khác b ng ng b .

Dữ liệu: Vào t file v n b n BRIDGES.INP:

- Dòng u tiên ch a 2 s nguyên n và m ($1 \leq n \leq 10^6, 1 \leq m \leq 5 \times 10^6$),
- Dòng th i trong m dòng ti p theo ch a 2 s nguyên u và v xác nh cây c u th i c n xây.

Kết quả: a ra file v n b n BRIDGES.OUT k t qu tìm c d i d ng m t s nguyên.

Ví dụ:

BRIDGES.INP	
4	5
0	1
0	2
1	2
2	3
3	0

BRIDGES.OUT
4

L i g i i :- Tìm á p s b ng tìm ki m nh phân (Ds min = $n-1$, ds max = m)



- V i m i ds d oán , ta ch vi c kì m tra tính liên thông v i danh sách c nh t 1 n ds .

Bài 3 Đường đi của Robot

M t b ng hình ch nh t có kích th c $M \times N$ (M, N nguyên d ng và không l n h n 100) c chia thành các ô vuông n v b ng các ng th ng song song v i các c nh. M t s ô vuông nào ó có th t các v t c n. T m t ô vuông, Robot có th i n m t ô vuông k c nh v i nó n u ô vuông ó không có v t c n. H i r ng n u Robot b t u xu t phát t m t ô vuông không có v t c n thu c dòng K, c t L thì có th i n c ô vuông không có v t c n thu c dòng H, c t O hay không? N u có thì hãy ch ra ng i qua ít ô vuông nh t.

D li u vào là t p v n b n BAI3.INP có c u trúc:

- Dòng u tiên ghi các ch s M, N, K, L, H, O . Các s ghi cách nhau ít nh t m t ký t tr ng;
- M dòng ti p theo, m i dòng ghi N s 1 ho c 0 tu thu c vào ô vuông t ng ng trong b ng hình ch nh t nêu trên có v t c n hay không (ghi s 1 n u có v t c n); các s trên m i dòng ghi liên ti p nhau.

D li u ra là t p v n b n BAI3.OUT có c u trúc:

- N u Robot có th i c t ô vuông thu c dòng K, c t L n ô vuông thu c dòng H, c t O thì:
- Dòng u tiên ghi 'Co duong di ';
 - Các dòng ti p theo, m i dòng ghi 2 s là ch s dòng và ch s c t c a các ô vuông trong ng i tìm c t ô vuông thu c dòng K, c t L n ô vuông thu c dòng H, c t O mà qua ít ô vuông nh t. Hai s trên m i dòng ghi cách nhau ít nh t m t ký t tr ng;
 - Ng c l i, n u Robot không th i c t ô vuông thu c dòng K, c t L n ô vuông thu c dòng H, c t O thì ghi 'Khong co duong di'.



Ví d 1:

robot.inp:	robot.out:
4 7 3 4 2 6	Có dương di
1000000	3 4
0010100	3 5
0000000	3 6
1101000	2 6

Ví d 2:

robot.inp:	robot.out:
4 7 2 2 1 3	Không có dương di
1010000	
0010100	
0100000	
1101000	

Phân tích:



Yêu c u c a bài toán th c ch t là tìm ng i t ô $[K,L]$ n ô $[H,O]$ sao cho qua ít ô vuông nh t. Ta đ th y thu t toán x lý m t cách h p lý nh t là thu t toán Loang. Ta b t d u “loang” t ô $[K,L]$, n u “loang” n c ô $[H,O]$ thì có ng i, ng c l i không có ng i.

Hàng i ph c v “loang” c th hi n b i m ng 2 chi u $Q: \text{Array}[1..2, \text{Mmax} * \text{Max}]$ of Byte; hàng th 1 c a Q l u thông tin ch s hàng, hàng th 2 l u thông tin c a ch s c t c a các ô khi n p vào Q .

M ng A l u thông tin tình tr ng các ô - có v t c n hay không c a b ng hình ch nh t ch a các ô vuông.

M ng P dùng ánh d u nh ng ô ã “loang” n; ng th i ph c v cho vi c truy xu t ng i sau này nên khi ô $[i,j]$ c “loang” n thì $P[i,j]$ c gán giá tr là r (r là giá tr t ng ng v i h ng mà ô tr c ó “loang” n, h ng nào t ng ng v i giá tr k bao nhiêu tu theo quy nh, ví d $r = 1$ - sang ph i, 2 - i xu ng, 3 - sang trái, 4 - i lên).

Sau khi th c hi n xong vi c “loang”, n u $P[H,O] = 0$ thì i u có có ngh a là ô $[H,O]$ ch a c “loang” n (không có ng i), n u $P[H,O] = r$ ($r = 1..4$ - loang theo 4 h ng) thì d a vào h ng “loang” n mà ta tìm c ô tr c ó, r i t a l i d a vào giá tr k c a ô tìm c ta tìm c ô tr c ó n a ... quá trình trên k t thúc khi tìm c ô $[K,L]$.

Sau khi “loang” xong thì giá tr các ph n t trong m ng Q không còn giá tr s d ng n a nên ta có th dùng m ng Q ph c v cho vi c truy xu t k t qu .

Bài 4 Gặp gỡ của hai Robot.

Trên m t l i ô vuông $M \times N$ ($M, N < 100$), ng i ta t Robot A góc trái trên, Robot B góc ph i d i. M i ô c a l i ô có th t m t v t c n ho c không (ô trái trên và ô ph i d i không có v t c n). Hai Robot b t u di chuy n ng th i v i t c nh nhau và không Robot nào c d ng l i trong khi Robot kia di chuy n (tr khi nó không th i c n a). T i m i b c, Robot ch có th di chuy n theo 4 h ng - i lên, i xu ng, sang trái, sang ph i - vào các ô k c nh. Hai Robot g p nhau n u chúng cùng ng trong m t ô vuông. Bài toán t ra là tìm cách di chuy n ít nh t mà 2 Robot ph i th c hi n có th g p nhau.

D li u vào cho b i t p robot.inp:



- Dòng u ghi 2 s M, N cách nhau ít nh t m t ký t tr ng;
- M dòng ti p theo, m i dòng ghi N s 0 ho c 1 liên ti p nhau mô t tr ng thái c a các ô vuông: 1 - có v t c n, 0 - không có v t c n.

D li u ra ghi vào t p robot.out:

- N u 2 Robot không th g p nhau thì ghi ký t '#'.
- Ng c l i, ghi hai dòng, m i dòng là m t dãy các ký t vị t li n nhau mô t các b c i c a Robot: U - i lên, D - i xu ng, L - sang trái, R - sang ph i. Dòng u là các b c i c a Robot A, dòng sau là các b c i c a Robot B.

Ví d :

robot.inp	robot.out	robot.inp	robot.out
4 6	DRRR	3 4	#
011000	LULU	0000	
000001		0000	
001001		0000	
010100			

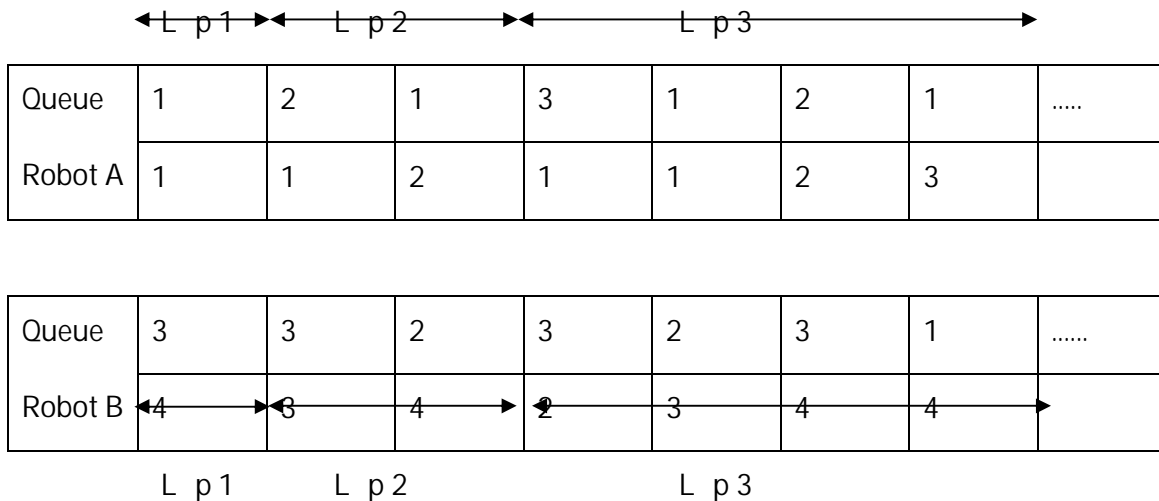
Phân tích:

V i d ng bài toán nh v y thì ta ngh ngay n thu t toán Loang tìm ng i cho 2 Robot. Nh v y là ph i “loang” t 2 phía (loang c a Robot A và loang c a Robot B). Nh ng vì 2 Robot di chuy n ng th i trong khi không cho phép ta cài t vị c “loang” song song t 2 phía nên ta ph i thì t k “loang” th nào cho h p lý.

Xin xu t m t ý t ng “loang” nh sau: C Robot A loang 1 l p thì d ng l i Robot B loang 1 l p, quá trình ó c l p i l p l i cho n khi 2 Robot g p nhau t i m t ô ho c 1 trong 2 Robot d ng “loang”. M t l p “loang” ây là “loang” t các ph n t hi n có trong hàng i (t ph n t Queue[dau] n ph n t Queue[cuoi]). Sau m i l p “loang”, bi n dau và bi n cuoi l i c i uch nh tr thành v trí u và v trí



củ i c a các ph n t m i trong Queue. Ta có th mô t c th các l p “loang” c a 2 Robot v i d li u vào là t probot.inp th 2 trên:



Q1, Q2 là 2 m ng dùng bi u di n c u trúc hàng i ph c v vì c “loang” c a 2 Robot. Trong quá trình “loang” ta ph i l u gi thông tin hàng, c t c a ô khi “loang” n, b i v y các ph n t c a Q1, Q2 là các record có ki u HC

HC = Record

$h, c: \text{Byte}; \{h: \text{lưu chỉ số hàng}, c: \text{lưu chỉ số cột}\}$

end;

Hai hàng i Q1, Q2 c kh i t o nh sau:

Procedure KT_Queue;

Begin

dau1:=1;

cuoi1:=1;

Q1[cuoi1]:=1;

Q1[cuoi1]:=1; {Robot A xuất phát từ ô [1,1]}

dau2:=1;



cuoi2:=1;

Q2[cuoi2]:=M;

Q2[cuoi2]:=N; {Robot B xuất phát từ ô [M,N]}

End;

Ngay sau khi kh i t o thì trong Q1 ch a ô [1,1], Q2 ch a ô [M,N]. ó là các ô xu t phát "loang" c a 2 Robot.

M i Robot t m t ô có th "loang" theo b n h ng: i xu ng, sang trái, i lên, sang ph i; nên thu n ti n cho vì c cài t ta s d ng k thu t "rào": M ng A[i,j] ch a thông tin các ô trong l i ô vuông c khai báo A:Array[0..Mmax + 1,0..Nmax + 1] of Byte (ch không ph i nh thông th ng là [1..Mmax,1..Nmax]) và c kh i t o FillChar(A,SizeOf(A),1) (nh v y là xung quanh l i ô vuông c "rào" b i s 1); ng th i s d ng 2 m ng h ng $H_i=(1,0,-1,0)$, $H_j=(0,-1,0,1)$.

Khi ó vì c "loang" theo l p c a Robot A c th ch i n nh sau:

Procedure LoangA;

Var

k:Byte;

Begin

j:=Cuoi1;

For i:=dau1 to cuoi1 do

For k:=1 to 4 do

Begin

h:= Q1[i].h + $H_i[k]$; { $k=1$ - đi xuống, 2 - sang trái, 3 - đi lên, 4 - sang phải}

c:= Q1[i].c + $H_j[k]$;

If A[h,c] = 0 then {ô [h,c] không có vật cản và chưa "loang" đến}

Begin



Inc(j);

Q1[j].h:= h;

Q1[j].c:= c; {Nạp ô [h,c] vào hàng đợi Q1}

A[h,c]:=k; {Đánh dấu ô bằng cách gán giá trị tương ứng với hướng loang}

B[h,c]:=True; {Dấu hiệu cho Robot B nhận biết đã gặp Robot A}

End;

End;

dau1:=cuoi1 + 1;

cuoi1:=j; {Điều chỉnh lại biến dau1, cuoi1 cho các phần tử mới trong Q1}

If dau1 > cuoi1 then ST:=True; {ST=True là Q1 rỗng, kết thúc "loang"}

End;

Vì c "loang" theo l p c a Robot B c ng t ng t nh Robot A nh ng ch khác ch khi
"loang" n m t ô [h,c] nào ó thì ph i xét d u hi u B[h,c] xem th ã g p Robot A
ch a:

.....

If B[h,c] then {Nếu tại ô [h,c] Robot B gặp Robot A thì}

Begin

lk:=k; {Luu lại giá trị tương ứng với hướng "loang" để lấy kết quả}

hm:=h; {Luu lại chỉ số hàng của ô mà 2 Robot gặp nhau để lấy kết quả}

cm:=c; {Luu lại chỉ số cột của ô mà 2 Robot gặp nhau để lấy kết quả}

TT:=True; {Dấu hiệu dừng "loang" của 2 Robot vì đã gặp nhau}

Exit;

End;



Số địa chỉ lưu trữ giá trị thông tin vị trí hàng "loang" ($lk:=k$) là vị trí ô g p nhau $[h,c]$ Robot A đã "loang" nên cần phải gán giá trị của $A[h,c]$ bằng giá trị thông tin vị trí hàng "loang" nên khi Robot B "loang" nên ô $[h,c]$ được địa chỉ lưu trữ giá trị thông tin vị trí hàng "loang" vào biến lk sau này truy xuất thông tin của Robot B.

Quá trình "loang" theo trình tự của 2 Robot được thể hiện như sau:

Procedure Loang_lop;

Begin

TT:=False;

ST:=False;

While (ST=False) and (TT=False) do

Begin

FillChar(B,SizeOf(B),False); {Đánh dấu theo từng lớp loang}

Loang1;

Loang2;

End;

End;

Lưu ý đánh dấu theo trình tự "loang" tại vị trí như trên: FillChar(B,SizeOf(B),False) là một thao tác vì Robot B gặp Robot A tại ô $[h,c]$ chỉ khi $B[h,c] = \text{True}$ thì mới lưu trữ "loang" của Robot A cùng trình tự "loang" với Robot B. Còn nếu $B[h,c] = \text{True}$ của trình tự "loang" trước đó của Robot A thì không thể lưu trữ 2 Robot gặp nhau vì khi đó 2 Robot sẽ di chuyển không thể khác nhau.

Vì cần kiểm tra dựa vào giá trị của biến TT: TT=True - Hai Robot gặp nhau, TT=False - Hai Robot không gặp nhau.

Trong trường hợp gặp nhau thì dựa vào vị trí lưu trữ thông tin ô g p nhau vào 2 biến hm , cm (hm - ch số hàng, cm - ch số cột) ta sẽ truy xuất thông tin của 2 Robot.



CÁC PHƯƠNG PHÁP TÌM KIẾM TRÊN ĐỒ THỊ

Lê Thị Tuyết Vân-Tổ Tin trường THPT chuyên Quốc Học, Huế

M t bài toán quan tr ng trong lí thuy t th là bài toán duy t t t c các nh có th n c t m t nh xu t phát nào ó. V n này a v m t bài toán li t kê mà yêu c u c a nó là không c b sót hay l p l i b t kì nh nào. Chính vì v y mà ta ph i xây d ng nh ng thu t toán cho phép duy t m t cách h th ng các nh, nh ng thu t toán nh v y g i là nh ng thu t toán tìm ki m trên th (graph traversal). Ta quan tâm n hai thu t toán c b n nh t: thu t toán tìm ki m theo chi u sâu và thu t toán tìm ki m theo chi u r ng.

1. Thu t toán tìm ki m theo chi u sâu:

a. Thu t toán tìm ki m theo chi u sâu:

Ý t ng:

T t ng c a thu t toán tìm ki m theo chi u sâu (Depth-First Search - DFS) có th trình bày nh sau: Tr c h t, d nhiên nh s n c t s, ti p theo, v i m i cung (s, x) c a th thì x c ng s n c t s. V i m i nh x ó thì t t nhiên nh ng nh y n i t x c ng n c t s...

i u ó g i ý cho ta vì t m t th t c quy DFSVisit(u) mô t vi c duy t t nh u b ng cách th m nh u và ti p t c quá trình duy t DFSVisit(v) v i v là m t nh ch a th m n i t u.

K thu t ánh d u c s d ng tránh vi c li t kê l p các nh: Kh i t o avail[v]:=true, $\forall v \in V$, m i l n th m m t nh, ta ánh d u nh ó l i (avail[v]:=false) các b c duy t quy k ti p không duy t l i nh ó n a.

Thu t toán:

```
procedure DFSVisit( $u \in V$ ); //Thu t toán tìm ki m theo chi u sâu t nh u
begin
    avail[u] := False; //avail[u]=False  $\Leftrightarrow$  u ã th m
    Output u; //Li t kê u
    for  $\forall v \in V: (u, v) \in E$  do //Duy t m i nh v ch a th m n i t u
        if avail[v] then DFSVisit(v);
    end;
begin //Ch ng trình chính
    Input th G
```




for $\forall v \in V$ do avail[v] := True; // ánh d um i nh uch ath m
DFSVisit(s);
end.

b. Thu t toán tìm ng i theo DFS:

Bài toán tìm ng i:

Cho th $G=(V,E)$ và hai nh $s, t \in V$.

Nh c l i nh ngh a ng i: M t dãy các nh:

$$P = \langle s = p_0, p_1, \dots, p_k = t \rangle (\forall i: (p_{i-1}, p_i) \in E)$$

c g i là m t ng i t st i t, ng i này g m k+1 nh p_0, p_1, \dots, p_k
và c nh $(p_0, p_1), (p_1, p_2), \dots, (p_{k-1}, p_k)$. nh s c g i là nh u và nh t
c g i là nh cu i c a ng i. N u t n t i m t ng i t st i t,
ta nói s n c t và t n \rightarrow c t s: s t.

Thu t toán:

L u l i ng i t nh xu t phát s, trong th t c DFSVisit(u), tr c
khi g i quy DFSVisit(v) v i v là m t nh ch a th m n i t uch a
ánh d u), ta l u l i v t ng i t u t i v b ng cách t trace[v]:=u,
t c là trace[v] l u l i nh l i n tr c v trong ng i t st i v. Khi
thu t toán DFS k t thúc, ng i t st i t s là: $\langle p_1 = t \leftarrow p_2 = \text{trace}[p_1] \leftarrow$
 $p_3 = \text{trace}[p_2] \leftarrow \dots \leftarrow s \rangle$

```
procedure DFSVisit(u ∈ V); //Thu t toán tìm ki m theo chi u sâu t nh u
begin
    avail[u] := False; //avail[u]=False ⇔ u ã th m
    for  $\forall v \in V: (u, v) \in E$  do //Duy t m i nh v ch a th m n i t u
        if avail[v] then
            begin
                trace[v] := u; //L u v t ng i, nh l i n tr c v tr n ng i t st i v l u
                DFSVisit(v); //G i quy tìm ki m theo chi u sâu t nh v
            end;
    end;
begin //Ch ng tr ình chính
    Input th G, nh xu t phát s, nh ích t;
    for  $\forall v \in V$  do avail[v] := True; // ánh d um i nh uch ath m
    DFSVisit(s);
    if avail[t] then //s i n c t
        «Truy theo v t t t tìm ng i t st i t»;
```



end.

Có thể không cần mảng đánh dấu avail[1 ... n] mà dùng luôn mảng trace[1 ... n] đánh dấu: Khi nào các phần tử mảng trace[1 ... n] là:

$$\begin{cases} \text{Trace}[s] = 0 \\ \text{Trace}[v] = 0, \forall v \neq s \end{cases}$$

Khi đó nếu khi nào phần tử mảng đánh dấu là trace[v] = 0, thì khi đó phần tử mảng đánh dấu v, phép gán trace[v] = u sẽ luôn luôn đúng vì v đánh dấu v đã là phần tử mảng (trace[v] = 0).

Tính chất của BFS

Nếu ta sắp xếp danh sách các đỉnh theo thứ tự tăng dần thì thuật toán DFS luôn trả về mảng đánh dấu có thứ tự tăng dần như thứ tự trong danh sách các đỉnh.

c. Thuật toán duyệt đồ thị theo DFS

Cài đặt trên đây là một thuật toán DFS liệt kê các đỉnh theo thứ tự tăng dần. Thuật toán DFS dùng để duyệt qua các đỉnh và các cạnh của đồ thị có hướng sau:

```
procedure DFSVisit(u ∈ V); //Thuật toán tìm kiếm theo chiều sâu từ đỉnh u
begin
    Time := Time + 1;
    d[u] := Time; //Thời gian duyệt từ u
    Output ← u; //Liệt kê u
    for v ∈ V: (u, v) ∈ E do //Duyệt tìm kiếm từ u
        if d[v] = 0 then DFSVisit(v); //Nếu v chưa thăm, gọi quy trình tìm kiếm theo chiều sâu từ v
    Time := Time + 1;
    f[u] := Time; //Thời gian duyệt xong u
end;
begin //Chương trình chính
    Input → đồ thị G
    for v ∈ V do d[v] := 0; //Mảng đánh dấu duyệt từ
    Time := 0;
    for v ∈ V do
        if d[v] = 0 then DFSVisit(v);
    end.
```

Thời gian thực hiện của thuật toán DFS có thể đánh giá bằng số lần gọi các DFSVisit(|V| lần) của vòng lặp for bên trong các DFSVisit. Chính vì vậy:

- Nếu đồ thị có hướng thì danh sách các đỉnh hoặc danh sách liên thuộc, vòng lặp for bên trong các DFSVisit (xét thuật toán chương trình) sẽ duyệt



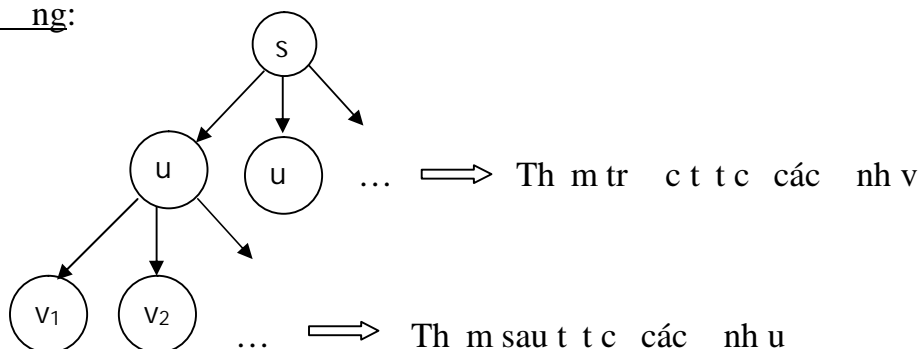
qua t t c các c nh c a th (m i c nh hai l n n u là th vô h ng, m i c nh m t l n n u là th có h ng). Trong tr ng h p này, th i gian th c hi n gi i thu t DFS là $(|V| + |E|)$

- N u th c bi u đi n b ng ma tr n k , vòng l p for bên trong m i th t c DFSVisit s ph i duy t qua t t c các nh $1 \dots n$. Trong tr ng h p này th i gian th c hi n gi i thu t DFS là $(|V| + |V|^2) = (|V|^2)$.
- N u th c bi u đi n b ng danh sách c nh, vòng l p for bên trong th t c DFSVisit s ph i duy t qua t t c danh sách c nh m i l n th c hi n th t c. Trong tr ng h p này th i gian th c hi n gi i thu t DFS là $(|V||E|)$.

2. Thu t toán tìm ki m theo chi u r ng:

a. Thu t toán tìm ki m theo chi u r ng

Ý t ng:



T t ng c a thu t toán tìm ki m theo chi u r ng (Breadth-First Search – BFS) là “l p l ch” duy t các nh. Vi c th m m t nh s lên l ch duy t các nh n i t nó sao cho th t duy t là u tiên chi u r ng (nh nào g n nh xu t phát s h n s c duy t tr c). u tiên ta th m nh s. Vi c th m nh s s phát sinh th t th m nh ng nh u_1, u_2, \dots n i t s (nh ng nh g n s nh t). T p theo ta th m nh u_1 , khi th m nh u_1 s l i phát sinh yêu c u th m nh ng nh r_1, r_2, \dots n i t u_1 . Nh ng rõ ràng các nh r này “xa” s h n nh ng nh u nên chúng ch c th m khi t t c nh ng nh u ã th m. T c là th t duy t nh s là: $s, u_1, u_2, \dots, r_1, r_2, \dots$

Thu t toán tìm ki m theo chi u r ng s d ng m t danh sách ch a nh ng nh ang “ch ” th m. T i m i b c, ta th m m t nh u danh sách, lo i nó ra khi danh sách và cho nh ng nh ch a “x p hàng” k v i nó x p hàng thêm vào cu i danh sách. Thu t toán s k t thúc khi danh sách r ng.

Vì nguyên t c vào tr c ra tr c, danh sách ch a nh ng nh ang ch th m c t ch c d i d ng hàng i (Queue): N u ta có Queue là m t hàng i v i th t c Push(r) y m t nh r vào hàng i và hàm Pop tr v m t nh l y ra t hàng i thì thu t toán BFS có th vi t nh sau:

Thu t toán:

```
Queue := (s); //Kh i t o hàng i ch g m m t nh s
for v ∈ V do avail[v] := True;
```



```

avail[s] := False; // ánh d uch có nh s cx phàng
repeat //L pt i khi hàng ir ng
u := Pop; //L yt hàng iram t nhu
Output u; //Li t kê u
for v ∈ V: avail[v] and (u, v) ∈ E do //Xét nh ng nh v k uch a c
// y vào hàng i
begin
Push(v); // y v vào hàng i
avail[v] := False; // ánh d uch v ã x phàng
end;
until Queue = ∅;

```

2. Thu t toán tìm ng i theo BFS:

```

Queue := (s); //Kh i t o hàng ich g m m t nh s
for v ∈ V do avail[v] := True;
avail[s] := False; // ánh d uch có nh s cx phàng
repeat //L pt i khi hàng ir ng
u := Pop; //L yt hàng iram t nhu
for v ∈ V: avail[v] and (u, v) ∈ E do //Xét nh ng nh v k uch a c
// y vào hàng i
begin
trace[v] := u; //L uch t ng i
Push(v); // y v vào hàng i
avail[v] := False; // ánh d uch v ã x phàng
end;
until Queue = ∅;

```

if avail[t] then //s i t i ct
«Truy theo v t t t tìm ng i t s t i t»;

T ng t nh thu t toán tìm ki m theo chi u sâu, ta có th dùng m ng Trace[1...n] kiêm luôn ch c n ng ánh d uch.

Tính ch t c a BFS

Thu t toán BFS luôn tr v ng i qua ít c nh nh t trong s t t c các ng i t s t i t. N u ta s p x p các danh sách k c a m i nh theo th t t ng d n và n u có nhi u ng i t s t i t u qua ít c nh nh t thì thu t toán BFS s tr v ng i có th t t i n nh nh t trong s nh ng ng i ó.

c. Thu t toán duy t th theo BFS

T ng t nh thu t toán DFS, trên th c t , thu t toán BFS c ng dùng xác nh m t th t trên các nh c a th và c vi t theo mô hình sau:

```

procedure BFSVisit(s ∈ V);
begin

```



```

Queue := (s); //Kh i t o hàng i ch g m m t n h s
Time := Time + 1;
d[s] := Time; //Duy t n n h s
repeat //L p t i k h i hàng i r ng
    u := Pop; //L y t hàng i r a m t n h u
    Time := Time+1;
    F[u]:=Time; //Ghi n h n t h i i m duy t x o n g n h u
    Output ← u; //L i t k ê u
    for v ∈ V: (u, v) ∈ E do //Xét n h ng n h v k u
        if d[v] = 0 then //N u v ch a duy t n
            begin
                Push(v); // y v vào hàng i
                Time := Time + 1;
                d[v] := Time; //Ghi n h n t h i i m duy t n n h v
            end;
until Queue = ∅;
end;
begin //Ch ng tr ì n h chính
    Input → th G;
    for v ∈ V do d[v] := 0; //M i n h u ch a c duy t n
    Time := 0;
    for v ∈ V do
        if d[v]=0 then BFSVisit(v);
    end.

```

Th i gian th c h i n g i i thu t c a BFS t ng t n h i v i DFS, b ng $(|V| + |E|)n$ u th c b i u đ i n b ng danh sách k ho c danh sách liên thu c, b ng $(|V|^2)n$ u th c b i u đ i n b ng ma tr n k , và b ng $(|V||E|)n$ u th c b i u đ i n b ng danh sách c nh.

Bài t p:

Bài 1:

Mê cung hình ch nh t kích th c $m \times n$ g m các ô vuông n v $(m, n \leq 1000)$. Trên m i ô ghi m t trong ba kí t :

- O: N u ô ó an toàn
- X: N u ô ó có c m b y
- E: N u là ô có m t nhà thám hi m ang ng.

Duy nh t ch có 1 ô ghi ch E. Nhà thám hi m có th t m t ô i sang m t trong s các ô chung c nh v i ô ang ng. M t cách i thoát kh i mê cung là



m t hành trình i qua các ô an toàn ra m t ô biên. Hãy ch giúp cho nhà thám hi m m t hành trình thoát ra kh i mê cung i qua ít ô nh t.

D li u vào t t p v n b n MECUNG.INP

- Dòng 1: Ghi m, n ($1 < m, n \leq 1000$).
- M dòng ti p theo th hi n b ng kích th c $m \times n$, mô t tr ng thái c a mê cung theo th t t trên xu ng d i, m i dòng n ký t theo th t t trái qua ph i.

Kết quả ghi ra file MECUNG.OUT

- Dòng 1: Ghi s b c i tìm c a hành trình tìm c.
- Dòng 2: Ghi m t xâu ký t S mô t hành trình tìm c (xâu ký t S ch g m các ch cái in hoa E, W, S, N mà m i ký t trong xâu S th hi n vị c i sang ô chung c nh theo h ng c mô t b i ký t ó. Ví d : E: i sang ô chung c nh theo h ng ông, W: i sang ô chung c nh theo h ng Tây, S: i sang ô chung c nh theo h ng Nam, N: i sang ô chung c nh theo h ng B c)

Ví d :

MECUNG.INP
4 5
XXXOX
XOOOX
XEXOO
XXXOO

MECUNG.OUT
4
NEEN

Chương trình

```
{ $MODE OBJFPC }
Const NMax = 1000;
    Fi = 'MECUNG.INP';
    Fo = 'MECUNG.OUT';
    dd: Array[1..4] of integer = ( 0, -1, 0, 1);
    dc: Array[1..4] of integer = (-1, 0, 1, 0);
    h: array[1..4] of char=('W', 'N', 'E', 'S');
Var a, tr: Array[1..NMax, 1..NMax] of integer;
    queue : Array[1..NMax*NMax] of Record
        d, c : integer;
    End;
    N, M, dau, cuoi, x0, y0, x1, y1: integer;
    ok: boolean;

Procedure DocF;
Var i, j : integer;
    s: string;
Begin
```



```
Assign(Input,Fi);
Reset(Input);
Readln(M,N);
For i:=1 to M do
  begin
    readln(s);
    For j:=1 to N do
      case s[j] of
        'O': a[i,j]:=0;
        'X': a[i,j]:=1;
        'E': begin
              a[i,j]:=1;
              x0:=i;
              y0:=j;
            end;
      end;
    end;
  end;
Close(Input);
End;

Procedure BFS(i,j : integer);
Var k,dong,cot,u,v : integer;
Begin
  ok:=false;
  if (x0=1) or (x0=m) or (y0=1) or (y0=n) then
    begin
      x1:=x0;
      y1:=y0;
      ok:=true;
      exit;
    end;
  Dau:=1;
  Cuoi:=1;
  queue[cuoi].d := i;
  queue[cuoi].c := j;
  tr[i,j] := 1;
  While dau<=cuoi do
    Begin
      dong := queue[dau].d;
      cot  := queue[dau].c;
      inc(dau);
      For k:=1 to 4 do
        Begin
```



```
u := dong + Dd[k];
v := cot + Dc[k];
If (u>0) and (u<=M) and (v>0) and (v<=N) then
  If (a[u, v]=0) and (tr[u,v]=0) then
    Begin
      Inc(cuoi);
      queue[cuoi].d := u;
      queue[cuoi].c := v;
      tr[u,v] := k;
      if (u=1) or (u=m) or (v=1) or (v=n) then
        begin
          x1:=u;
          y1:=v;
          ok:=true;
          exit;
        end;
    End;
  End;
End;

Procedure Inkq;
Var i, x, y: integer;
    s:string;
Begin
  Assign(OutPut,fo);
  Rewrite(OutPut);
  if not ok then writeln(-1)
  else
    begin
      s:='';
      while (x1<>x0) or (y1<>y0) do
        begin
          s:=h[tr[x1,y1]]+s;
          x:=x1;
          y:=y1;
          x1:=x-dd[tr[x,y]];
          y1:=y-dc[tr[x,y]];
        end;
      writeln(length(s));
      writeln(s);
    end;
  Close(Output);
```




End;

BEGIN

DocF;

BFS(x0,y0);

Inkq;

END.

Bài 2:

Trên bàn c $m \times n$ ($1 \leq m, n \leq 1000$) ô vuông có k quân mã ang ng nh ng ô nào ó ($1 \leq k \leq 1000$). Trong quá trình di chuy n, quân mã có th nh y n ô ã có nh ng quân mã khác ang ng. Hãy tìm cách di chuy n k quân mã n v trí ô [x0, y0] cho tr c sao cho t ng b c i c a các quân mã là nh nh t.

Dữ liệu vào t p v n b n HORSES.INP:

- Dòng 1 ch a 5 s nguyên d ng m, n, x0, y0, k.
- k dòng ti p theo m i dòng ghi 2 s nguyên là t a c a m t quân mã.

Kết quả ghi vào t p v n b n HORSES.OUT:

- Ghi m t s duy nh t là t ng s b c i c a các quân mã. Trong tr ng h p không di chuy n c m t quân mã nào ó v v trí [x0, y0] thì ghi -1.

Ví d :

HORSES.INP
8 8 8 8 3
1 1
2 2
3 3

HORSE.OUT
14

Phân tích:

Loang t i m (x0, y0) ra h t b ng.

Trong b ng len[1..n, 1..n], t i m i ô ghi s b c i c a quân mã di chuy n t ô [x0, y0] n ô ó.

N u t i ô có quân mã không có giá tr thì không có cách di chuy n quân mã ó n ô [x0, y0] ghi -1, ng c l i ta tính t ng s c a các s ghi trong các ô có quân mã ang ng, t ng s ó là áp s bài toán.

Chương trình

```
{ $MODE OBJFPC }
```

```
Const NMax = 1000;
```

```
Fi = 'HORSES.INP';
```

```
Fo = 'HORSES.OUT';
```

```
dd: Array[1..8] of integer = (-1, -2, -2, -1, 1, 2, 2, 1);
```



```
dc: Array[1..8] of integer = (-2,-1, 1, 2, 2, 1,-1,-2);
Var len: Array[1..NMax,1..NMax] of integer;
queue : Array[1..NMax*NMax] of Record
    d,c : integer;
    End;
N, M, x0, y0, q, dau, cuoi: integer;
x, y: array[1..NMax] of integer;
```

```
Procedure ReadFile;
Var i, j : integer;
Begin
    Assign(Input,Fi);
    Reset(Input);
    Readln(M,N, x0, y0, q);
    For i:=1 to q do readln(x[i],y[i]);
    Close(Input);
End;
```

```
Procedure BFS(i,j : integer);
Var k,dong,cot,u,v,t : integer;
Begin
    Dau:=1;
    Cuoi:=1;
    queue[cuoi].d:=i;
    queue[cuoi].c:=j;
    fillchar(len, sizeof(len),0)
    len[i,j] := 1;
    While dau<=cuoi do
        Begin
            dong := queue[dau].d;
            cot := queue[dau].c;
            inc(dau);
            For k:=1 to 8 do
                Begin
                    u := dong + Dd[k];
                    v := cot + Dc[k];
                    If (u>0) and (u<=M) and (v>0) and (v<=N) then
                        If len[u,v]=0 then
                            Begin
                                Inc(cuoi);
                                queue[cuoi].d := u;
                                queue[cuoi].c := v;
                                len[u,v] := len[dong,cot]+1;
```



```

        End;
    End;
End;

Procedure PrintResult;
Var i, s: integer;
Begin
    Assign(OutPut,fo);
    Rewrite(OutPut);
    s:=0;
    for i:=1 to q do
        if len[x[i],y[i]]=0 then
            begin
                s:=-1;
                break;
            end
        else s:=s+len[x[i],y[i]]-1;
    writeln(s);
    close(Output);
End;

BEGIN
    ReadFile;
    BFS(x0,y0);
    PrintResult;
END.

```

Bài 3:

Cho m t th vô h ng có N nh c ánh s t 1 n N. Hãy tìm các vùng liên thông c a th .

Dữ liệu vào t file v n b n SVLT.INP

- Dòng 1: Ghi n, m l n l t là s nh và s c nh c a th ($1 < n \leq 100$)
- M dòng t i p theo: m i dòng ghi hai nh uc a m t c nh.

Kết quả ghi ra file SVLT.OUT

- Dòng 1: Ghi s K là s vùng liên thông.
- K dòng t i p theo: m i dòng ghi các nh thu c cùng 1 vùng liên thông.

Ví d :

SVLT.INP
11 10
1 2
3 4

SVLT.OUT
4
1 2
3 4 5 6 7 8



3 6
4 5
4 6
5 7
6 7
6 8
7 8
10 11

9
10 11

```
Const Max = 100;  
    Fi = 'SVLT.INP';  
    Fo = 'SVLT.OUT';  
Var A: Array[1..Max,1..Max] of boolean;  
    D: Array[1..Max] of integer;  
    queue : Array[1..Max*Max] of Integer;  
    N, dau, cuoi, sv: integer;
```

```
Procedure ReadFile;  
Var i, u, v, m : integer;  
Begin  
    Assign(Input,Fi);  
    Reset(Input);  
    Readln(N, m);  
    fillchar(a, sizeof(a), false);  
    For i:=1 to M do  
        begin  
            Read(u,v);  
            a[u, v]:=true;  
            a[v, u]:=true;  
        end;  
    Close(Input);  
End;
```

```
Procedure BFS(u : integer);  
Var v : integer;  
Begin  
    Dau:=1;  
    Cuoi:=1;  
    queue[cuoi] := u;  
    D[u] := sv;  
    While dau<=cuoi do  
        Begin  
            u := queue[dau];
```



```
inc(dau);
For v:=1 to n do
  If A[u,v] and (D[v]=0) then
    Begin
      Inc(cuoi);
      queue[cuoi] := v;
      D[v]      := sv;
    End;
  End;
End;

End;

Procedure Timsvlt;
var i: integer;
Begin
  Sv := 0;
  fillchar(D, sizeof(d), 0);
  Fillchar(D,sizeof(D),0);
  for i:=1 to n do
    if D[i]=0 then
      begin
        inc(sv);
        BFS(i);
      end;
End;

Procedure Inkq;
Var i, j: integer;
Begin
  Assign(OutPut,fo);
  Rewrite(OutPut);
  writeln(sv);
  For i:=1 to sv do
    Begin
      For j:=1 to N do
        If D[j]=i then Write(j,' ');
      Writeln;
    end;
  Close(Output);
End;

BEGIN
  ReadFile;
  Timsvlt;
```



Inkq;
END.

Bài 4:

Cho bảng hình chữ nhật chia thành $m \times n$ ô vuông. Trong m ô vuông có ghi số 0 hoặc 1. Một miền 0 của bảng là tập hợp các ô chung nhau của số 0. Hãy tính số miền 0 của bảng và diện tích của miền 0.

Dữ liệu vào từ file văn bản MIEN0.INP

- Dòng 1: Ghi m, n ($1 < m, n \leq 100$).
- Mỗi dòng tiếp theo ghi n số theo thứ tự từ trái qua phải.

Kết quả ghi ra file MIEN0.OUT

- Dòng 1: Ghi số miền 0.
- Dòng 2: ghi diện tích của các miền 0.

Ví dụ :

MIEN0.INP	MIEN0.OUT
8 10	4
0 1 0 0 0 0 0 0 1 0	1 25 14 9
1 1 0 0 0 0 0 0 1 0	
0 0 0 1 1 0 0 0 1 0	
1 1 1 0 1 1 0 0 1 0	
0 0 1 1 0 0 0 0 1 0	
0 0 0 1 1 1 1 1 1 0	
1 1 0 1 0 0 0 1 0 1	
0 0 0 1 0 0 1 0 1 0	

```

Const Max = 100;
Fi = 'MIEN0.INP';
Fo = 'MIEN0.OUT';
dc: Array[1..8] of integer = ( 0, 1, 1, 1, 0, -1, -1, -1);
dd: Array[1..8] of integer = (-1, -1, 0, 1, 1, 1, 0, -1);
Var A, D: Array[1..Max, 1..Max] of integer;
    QUEUE : Array[1..Max*Max] of Record
        d, c : integer;
    End;
    DT : Array[1..Max*Max] of Integer;
    N, M, dau, cuoi, sv : integer;

```

```

Procedure DocF;
Var i, j : integer;
Begin
    Assign(Input, Fi);

```



```
Reset(Input);
Readln(M,N);
For i:=1 to M do
    For j:=1 to N do Read(A[i,j]);
Close(Input);
End;

Procedure BFS(i,j : integer);
Var k,dong,cot,u,v : integer;
Begin
    Dau:=1;
    Cuoi:=1;
    QUEUE[cuoi].d := i;
    QUEUE[cuoi].c := j;
    D[i,j] := sv;
    DT[SV]:=1;
    While dau<=cuoi do
        Begin
            dong := QUEUE[dau].d;
            cot := QUEUE[dau].c;
            inc(dau);
            For k:=1 to 8 do
                Begin
                    u := dong + Dd[k];
                    v := cot + Dc[k];
                    If (u>0) and (u<=M) and (v>0) and (v<=N) then
                        If (A[u,v]=0) and (D[u,v]=0) then
                            Begin
                                Inc(cuoi);
                                QUEUE[cuoi].d := u;
                                QUEUE[cuoi].c := v;
                                D[u,v] := sv;
                                Inc(DT[sv]);
                            End;
                End;
            End;
        End;
    End;

Procedure Timsvlt;
var i, j: integer;
Begin
    Sv := 0;
    fillchar(D, sizeof(d), 0);
```



```
fillchar(DT, sizeof(DT), 0);
Fillchar(D, sizeof(D), 0);
for i:=1 to m do
  for j:=1 to n do
    if (a[i,j]=0) and (D[i,j]=0) then
      begin
        inc(sv);
        BFS(i,j);
      end;
End;
```

```
Procedure Inkq;
Var i: integer;
Begin
  Assign(OutPut, fo);
  Rewrite(OutPut);
  writeln(sv);
  For i:=1 to sv do Write(DT[i], ' ');
  Close(Output);
End;
```

```
BEGIN
  DocF;
  Timsvlt;
  Inkq;
END.
```

Bài 5:

M t lâu ài c chia thành $m \times n$ modul vuông ($1 < m, n \leq 50$). M i modul vuông có t 0 n 4 b c t ng. Hãy vi t ch ng trình tính :

- 1 - Lâu ài có bao nhiêu phòng?
- 2 - Di n tích phòng l n nh t là bao nhiêu?
- 3 - B c t ng nào c n lo i b phòng càng r ng càng t t?

Dữ liệu vào t t p v n b n LAUDAI.INP

Dòng 1: ghi s l ng các modul theo h ng B c-Nam và s l ng các modul theo h ng ông Tây.

Trong các dòng ti p theo, m i modul c mô t b i 1 s ($0 \leq p \leq 15$). S ó là t ng c a: 1 (= t ng phía Tây), 2 (= t ng phía B c), 4 (= t ng phía ông), 8 (= t ng phía Nam).

Các b c t ng bên trong c xác nh hai l n ; b c t ng phía Nam trong modul (1,1) ng th i là b c t ng phía B c trong modul (2,1)

K t qu ghi ra t p v n b n LAUDAI.OUT

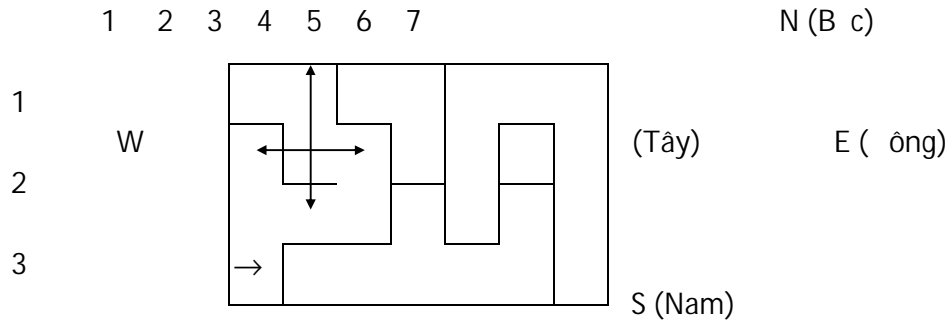


Dòng 1: ghi s l ng phòng.

Dòng 2: ghi di n tích c a phòng l n nh t (tính theo s modul)

Dòng 3: ghi b c t ng c n lo i b (tr c tiên là hàng sau ó là c t c a modul có t ng ó) và dòng cu i cùng là h ng c a b c t ng.

Ví d :



M i tên ch b c t ng c n lo i b theo k t qu ví d

```
Const NMax = 50;
Fi = 'LAUDAI.INP';
Fo = 'LAUDAI.OUT';
dd: Array[0..3] of integer = ( 0,-1, 0, 1);
dc: Array[0..3] of integer = (-1, 0, 1, 0);
h: array[0..3] of char=('W','N', 'E', 'S');
Var A, D: Array[1..NMax,1..NMax] of integer;
queue : Array[1..NMax*NMax] of Record
    d,c : integer;
End;
DT : Array[1..NMax*NMax] of Integer;
N, M, dau, cuoi, sp, MaxDT, i0, j0, k0: integer;
```

```
Procedure DocF;
Var i,j : integer;
Begin
    Assign(Input,Fi);
    Reset(Input);
    Readln(M,N);
    For i:=1 to M do
        For j:=1 to N do Read(A[i,j]);
    Close(Input);
End;
```

```
Procedure BFS(i,j : integer);
Var k,dong,cot,u,v : integer;
Begin
```



```
Dau:=1;
Cuoi:=1;
queue[cuoi].d := i;
queue[cuoi].c := j;
D[i,j] := sp;
DT[sp]:=1;
While dau<=cuoi do
  Begin
    dong := queue[dau].d;
    cot  := queue[dau].c;
    inc(dau);
    For k:=0 to 3 do
      Begin
        u := dong + Dd[k];
        v := cot  + Dc[k];
        If (u>0) and (u<=M) and (v>0) and (v<=N) then
          If ((A[dong,cot] shr k) and 1 =0) and (D[u,v]=0)
            then
              Begin
                Inc(cuoi);
                queue[cuoi].d := u;
                queue[cuoi].c := v;
                D[u,v]      := sp;
                Inc(DT[sp]);
              End;
            End;
          End;
        End;
      End;
    End;
  End;
End;

procedure TimDtMax;
var i: integer;
begin
  MaxDT:=0;
  for i:=1 to sp do
    if MaxDT<DT[i] then MaxDT:=DT[i];
  end;

procedure TimTuong;
var i, j, k, max, u, v: integer;
begin
  max:=0;
  for i:=1 to m-1 do
    for j:=1 to n-1 do
```



```
for k:=2 to 3 do
begin
    u:=i+dd[k];
    v:=j+dc[k];
    if ((a[i,j] shr k) and k =1) and (D[i,j]<>D[u,v])
    then
        if max < DT[D[i,j]]+DT[D[u,v]] then
            begin
                max:=DT[D[i,j]]+DT[D[u,v]];
                i0:=i;
                j0:=j;
                k0:=k;
            end;
        end;
end;

end;

Procedure Timsvlt;
var i, j: integer;
Begin
    sp := 0;
    fillchar(DT, sizeof(DT), 0);
    Fillchar(D,sizeof(D),0);
    for i:=1 to m do
        for j:=1 to n do
            if D[i,j]=0 then
                begin
                    inc(sp);
                    BFS(i,j);
                end;
    End;

Procedure Inkq;
Var i: integer;
Begin
    Assign(OutPut,fo);
    Rewrite(OutPut);
    writeln(sp);
    Writeln(MaxDT);
    writeln(i0,' ', j0, ' ', h[k0]);
    Close(Output);
End;

BEGIN
    DocF;
    Timsvlt;
```



TimDtMax;

TimTuong;

Inkq;

END.

Bài 6:

Cho m t l i hình ch nh t kích th c $m \times n$ g m các ô vuông n v , m i ô c tô 1 trong 6 màu ký hi u màu 1 , màu 2... màu 6. Gi thi t màu c a 2 ô trái trên và ph i d i là khác nhau. Hai ô chung c nh cùng thu c m t mi n n u cùng màu . Ng i A ng mi n có ch a ô góc trái trên, ng i B ng mi n có ch a ô ph i d i . Hai ng i ch i l n l t, n l t mình ng i ch i có th tô l i màu c a mi n mà mình ang ng. Trò ch i k t thúc khi hai ng i ng hai mi n c nh nhau (chung nhau ít nh t m t c nh c a m t ô vuông). Tính s l t i ít nh t trò ch i ó k t thúc.

Gi i h n: $1 \leq m, n \leq 100$. S l ng mi n ≤ 100 .

Dữ liệu vào t t p v n b n DOIMAU.INP:

- Dòng u: ghi hai s m, n .
- M dòng ti p theo, s th j c a dòng j ghi s hi u màu c a ô $[i, j]$.

Kết quả ghi ra t t p v n b n DOIMAU.OUT: ghi 1 s duy nh t là s l t i ít nh t trò ch i k t thúc.

Ví d :

DOIMAU.INP	DOIMAU.OUT
4 3	3
1 2 2	
2 2 1	
1 4 3	
1 3 2	

Phân tích:

+ Loang t ô $[1, 1]$ tìm s mi n (sm).

1	2	2
2	2	3
4	5	6
4	7	8

+ Xây d ng véc t V màu c a t ng mi n

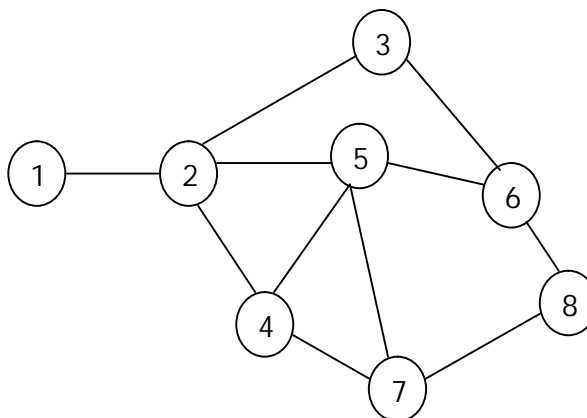
V=

1	2	3	4	5	6	7	8
1	2	1	1	4	3	3	2



+ Xây dựng đồ thị đồ thị msm nh, xem m t m i n là m t nh c a th. Giả a hai nh có c nh n i n u hai m i n ó có chung nhau ít nh t m t c nh c a m t ô vuông.

+ Tìm ng i ng n nh t t nh 1 n nh sm



Trong th t c BFS, t i m i b c, ta th m m t nh u danh sách (g i s nh ó là nh u), lo i nó ra kh i danh sách và cho nh ng nh v, ch a “x p hàng” k v i u x p hàng thêm vào cu i danh sách, tô màu nh v g i ng màu nh u, ng th i cho các nh k v i nh v có màu g i ng v i nh u, ch a x p “x p hàng” thêm vào cu i danh sách.

```
{ $MODE OBJFPC }
Const Max = 100;
  Fi = 'DOIMAU.INP';
  Fo = 'DOIMAU.OUT';
  dd: Array[1..4] of integer = ( 0, -1, 0, 1);
  dc: Array[1..4] of integer = (-1, 0, 1, 0);
Var
  A, B, D: Array[1..Max, 1..Max] of integer;
  Queue : Array[1..Max*Max] of record
    d, c: integer;
  end;

  len: array[1..max] of integer;
  mau: array[1..max] of integer;
  N, M, sv : integer;
```

```
Procedure DocF;
Var i, j : integer;
Begin
  Assign(Input, Fi);
  Reset(Input);
  Readln(M, N);
  For i:=1 to M do
    For j:=1 to N do Read(A[i, j]);
  fillchar(b, sizeof(b), 0);
  fillchar(mau, sizeof(mau), 0);
  Close(Input);
End;
```



```
Procedure BFS(i,j : integer);
Var k,dong,cot,u,v, dau, cuoi: integer;
    Queue : Array[1..Max*Max] of record
                                d, c: integer;
                                end;
Begin
    Dau:=1;
    Cuoi:=1;
    Queue[cuoi].d := i;
    Queue[cuoi].c := j;
    D[i,j] := sv;
    mau[sv]:=a[i,j];
    While dau<=cuoi do
        Begin
            dong := Queue[dau].d;
            cot := Queue[dau].c;
            inc(dau);
            For k:=1 to 4 do
                Begin
                    u := dong + Dd[k];
                    v := cot + Dc[k];
                    If (u>0) and (u<=M) and (v>0) and (v<=N) then
                        begin
                            If (a[u,v]=a[i,j])and(D[u,v]=0) then
                                Begin
                                    Inc(cuoi);
                                    Queue[cuoi].d := u;
                                    Queue[cuoi].c := v;
                                    D[u,v] := sv;
                                End;
                            if (a[u,v]<>a[i,j]) and (D[u,v]<>0) then
                                begin
                                    b[d[u,v],sv]:=1;
                                    b[sv,d[u,v]]:=1;
                                end;
                            end;
                        end;
                End;
            End;
        End;
    End;
```

```
Procedure Timsvlt;
var i, j: integer;
Begin
    Sv := 0;
    fillchar(D, sizeof(d), 0);
    for i:=1 to m do
        for j:=1 to n do
            if D[i,j]=0 then
                begin
                    inc(sv);
                    BFS(i,j);
                end;
```



```
End;

procedure BFS1;
Var k, u, v, dau, cuoi : integer;
    queue: array[1..max] of integer;
Begin
    Dau:=1;
    Cuoi:=1;
    Queue[cuoi]:=1;
    len[1]:=1;
    While dau<=cuoi do
        Begin
            u:=queue[dau];
            inc(dau);
            For v:=1 to sv do
                if (b[u,v]=1) and (len[v]=0) then
                    Begin
                        Inc(cuoi);
                        Queue[cuoi]:=v;
                        len[v]:=len[u]+1;
                        mau[v]:=mau[u];
                        for k:=1 to sv do
                            if (b[v,k]=1)and(mau[k]=mau[u])and(len[k]=0) then
                                begin
                                    inc(cuoi);
                                    queue[cuoi]:=k;
                                    len[k]:=len[v];
                                end;
                        end;
                    End;
            End;
        End;
End;

Procedure Inkq;
Var i, j: integer;
Begin
    Assign(OutPut,fo);
    Rewrite(OutPut);
    {writeln(sv);
    For i:=1 to m do
        begin
            for j:=1 to n do Write(D[i,j], ' ');
            writeln;
        end;}
    write(len[sv]-1);
    Close(Output);
End;

BEGIN
    DocF;
    Timsvlt;
    BFS1;
    Inkq;
END.
```



Trên đây là m t s bài t p tôi thu th p c d y cho h c sinh trong ph n các ph ng pháp tìm ki m trên th . Vì th i gian chu n b quá ng n nên không tránh kh i nh ng sai sót, r t mong nh n c nh ng óng góp chân tình c a các Th y Cô, tôi xin chân thành c m n.

TÀI LI U THAM KH O:

1. Tài li u giáo khoa chuyên Tin quy n 1.
2. Chuyên b i d ng h c sinh gi i Tin H c Trung h c ph thông ng d ng lý thuy t th (tác gi H S àm – Tr n Hùng)

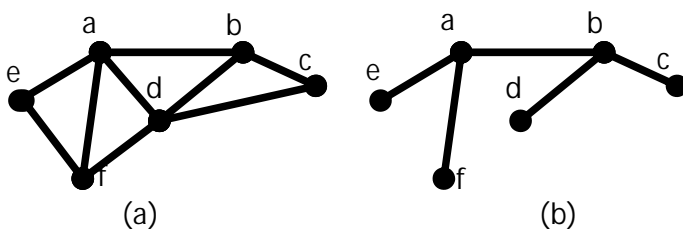


CÂY KHUNG VÀ CÂY KHUNG NH TH T

Tống Thanh Kiều
THPT Chuyên Vĩnh Phúc

1. t v n

Hệ thống giao thông của một thành phố được biểu diễn như hình vẽ cho bởi hình 1a. Các đường phố như hình vẽ có thể bị ngập lụt vào mùa đông là phải cào tuyết để thông xe. Chính quyền địa phương muốn cào tuyết ít nhất các con đường sao cho luôn có thông xe từ hai thành phố bất kỳ. Có thể làm được điều này bằng cách nào?



Hình 1. a) Hệ thống đường và b) tập các con đường cần phải cào tuyết

Cần phải cào tuyết ít nhất trên những con đường nào để mọi cặp thành phố bất kỳ. Hình 1b biểu diễn một tập hợp các con đường như vậy. Ta nhận thấy tập con biểu diễn các con đường này là một cây vì nó liên thông và chỉ có một chu trình, nên nó là một cây.

Bài toán trên có thể giải bằng thuật toán tìm kiếm có hướng để tìm ra các cạnh và chu trình của các cạnh để thu được kết quả. Thuật toán này là một cây.

2. Cây khung

2.1. định nghĩa

Cho G là một đồ thị. Một cây khung của G là một đồ thị con của G và chứa tất cả các đỉnh của G .

Một đồ thị có cây khung sẽ là một đồ thị liên thông vì có một cây khung. Một đồ thị có hai thành phần bất kỳ. Nếu đồ thị liên thông, thì nó là một đồ thị liên thông và có cây khung.



2.2. nh lí

M t n th là liên thông n u và ch n u nó có cây khung

Chứng minh: Tr c tiên, gi s th G có cây khung T. T ch a t t c các nh c a G. H n n a, có ng i trong T gi a hai nh b t k . Vì T là th con c a G nên có ng i trong G gi a hai nh c a nó. Do ó G là liên thông.

Bây gi , gi s G là liên thông. N u G không ph i là m t cây thì nó ph i có chu trình n. Xóa i m t c nh c a m t trong các chu trình n này. th nh n c m t s ít c nh h n nh ng v n còn ch a t t c các nh c a G và v n liên thông. N u th con này không là cây thì nó còn ch a chu trình n. C ng gi ng nh trên, ta l i xóa i m t c nh c a chu trình n. L p l i quá trình này cho n khi không còn chu trình n. i u này là có th vì ch có m t s h u h n các c nh trong th . Quá trình k t thúc khi không còn chu trình n trong th nh n c. Cây c t o ra vì th v n còn liên thông khi xóa i các c nh. Cây này là cây khung vì nó ch a t t c các nh c a G.

2.3. Tìm kì m u tiên theo chỉ u sâu

Cách ch ng minh nh lí 1 a ra m t thu t toán tìm cây khung b ng cách xóa i các c nh kh i các chu trình n. Thu t toán này là không hi u qu vì nó òi h i ph i nh n bi t c các chu trình n. Thay cho vi c xây d ng cây khung b ng cách lo i b các c nh, cây khung có th c xây d ng b ng cách l n l t ghép thêm các c nh.

Ta s xây d ng cây khung c a m t th liên thông b ng ph ng pháp tìm kì m u tiên theo chỉ u sâu. Ngh a là s t o m t cây có g c và cây khung s là th vô h ng n n c a cây có g c này. Ch n tùy ý m t nh c a th làm g c. Xây d ng ng i t nh này b ng cách l n l t ghép thêm các c nh vào sao cho m i m nh m i ghép s n i nh cu i cùng trên ng i v i m t nh còn ch a thu c ng i. Ti p t c ghép thêm c nh vào ng i ch ng nào không th thêm c n a thì thôi. N u ng i qua t t c các nh c a th thì cây do ng i này t o nên s là cây khung. Nh ng n u ng i không i qua t t c các nh thì c n thêm các c nh khác vào ng i. Lùi l i nh tr c nh cu i cùng c a ng i và n u có th , xây d ng ng i m i xu t phát t nh này qua các nh còn ch a thu c ng i. N u i u ó không th làm c thì lùi thêm m t nh n a trên ng i, t c là lùi l i hai nh trên ng i và th xây d ng ng i m i.

L p l i th t c này, b t u t nh cu i cùng c ghé th m lùi theo ng i m i l n m t nh, xây d ng ng i m i càng dài càng t t cho t i khi nào không th thêm c m t c nh nào n a. Vì th có h u h n c nh và là liên thông nên quá



trình ó s k t thúc và t o c cây khung. M i nh mà t i ó ng i k t thúc m i giai o n c a thu t toán s là lá trong cây có g c. M i nh t i ó ng i b t u t ó s là m t nh trong.

Tìm ki m u tiên chi u sâu c ng c g i là th t c quay lui vì nó quay l i nh ã ghé th m tr c trên ng i.

Các c nh c a th tìm c nh tím ki m u tiên theo chi u sâu g i là các c nh c a cây. Các c nh khác c a th có th n i v i nh tr c ho c sau nó trong cây. Các c nh này g i là các c nh quay lui.

2.4. Thu t toán tìm ki m u tiên theo chi u sâu

Trong thu t toán này, chúng ta xây d ng cây khung c a th G v i các nh v_1, v_2, \dots, v_n b ng cách l y nh v_1 làm g c c a cây. Kh i t o t p T là cây ch có m t nh này. Trong m i b c, thêm m t nh m i vào cây T cùng v i c nh i ra t nh c a T không ch a chu trình vì không có c nh c thêm vào mà nó n i v i nh ã có trong cây. Tuy nhiên, T v n là liên thông nh nó c xây d ng. Vì G là liên thông, m i nh trong G u c th m và c ghép vào cây. T ó suy ra T là cây khung c a G .

procedure $DFS(G$: th liên thông v i các nh v_1, v_2, \dots, v_n)

$T :=$ cây ch ch a m t nh v_1

visit(v_1)

procedure $visit(v$: nh c a G)

for m i nh w li n k v i v và ch a có trong T

Begin

thêm nh w và c nh (v, w) vào T

visit(w)

end

Phân tích ph c t p: V i m i nh v th t c visit(v) c g i khi nh v l n u tiên c g p trong tìm ki m và không c g i l i. Gi s ta có danh sách k c a G , tìm các nh k c a v không c n ph i tính toán gì c . Theo t ng b c c a thu t toán chúng ta xem xét m t c nh nhi u nh t hai l n quy t nh xem có nên thêm c nh này vào nh cu i c a nó hay không. Do v y th t c DFS xây d ng cây khung dùng $O(e)$ hay $O(n^2)$ các b c trong ó e và n t ng ng là s c nh, s nh c a G .



2.5. Tìm kiếm ưu tiên chi nhánh

Có thể xây dựng cây khung của một đồ thị bằng thuật toán tìm kiếm ưu tiên theo chi nhánh. Một lần nữa, cây có gốc sẽ được xây dựng và đồ thị vô hướng không có cây có gốc sẽ tạo nên cây khung.

Chọn một nhánh bất kỳ của đồ thị làm gốc. Sau đó ghép vào tiếp các cạnh liên thuộc với nhánh này. Các cạnh mới ghép vào trong giai đoạn này trở thành các cạnh mới của cây khung. Số cạnh phải ghép theo một đồ thị tùy ý.

Tiếp theo với mỗi nhánh mới ghép thêm theo đồ thị các số cạnh phải ghép tiếp các cạnh liên thuộc với nó vào cây mà không tạo ra chu trình. Số cạnh phải ghép các cạnh còn lại của nhánh mới theo một đồ thị nào đó. Quá trình này tạo ra nhánh mới của cây. Tiếp tục làm lại đồ thị này cho tới khi tất cả các cạnh của đồ thị được ghép vào cây. Đồ thị này kết thúc vì chỉ có một số hữu hạn các cạnh của đồ thị. Cây khung được tạo ra vì xây dựng các cạnh của đồ thị.

2.6. Thuật toán tìm kiếm ưu tiên theo chi nhánh

Trong thuật toán này chúng ta gán các cạnh v_1, v_2, \dots, v_n của đồ thị liên thông G là số cạnh phải ghép theo đồ thị nào đó. Chúng ta cần dùng một số để mô tả đồ thị thêm cạnh mới và các cạnh mới vào cây kết quả để hình thành đồ thị mà số cạnh không tạo ra vòng lặp.

procedure $BFS(G: \text{đồ thị liên thông với các cạnh } v_1, v_2, \dots, v_n)$

$T :=$ cây con của đồ thị v_1

$L :=$ danh sách rỗng

Đưa v_1 vào danh sách L gồm các cạnh không xấp xỉ

While L khác rỗng

Begin

Xóa cạnh đầu tiên, v_1 khỏi L

For mỗi cạnh w của v

If w chưa nằm trong L và không thuộc T **then**

Begin

Thêm cạnh w vào cuối danh sách L

Thêm cạnh w và các cạnh $\{v, w\}$ vào T



end

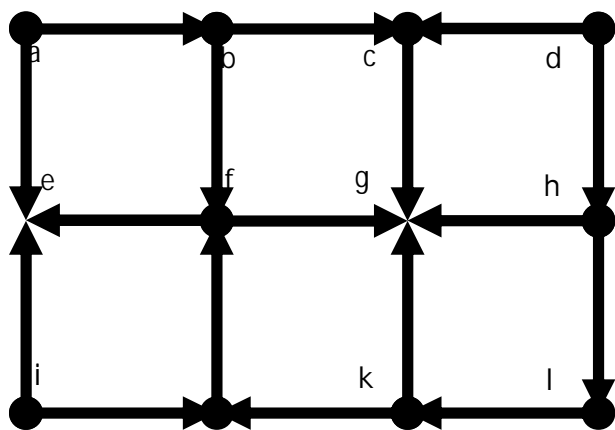
end

Phân tích ph c t p: V i m i nh v c a th xem xét t t c các nh li n k v i v và thêm vào cây T m i nh còn ch a c th m. Gi s có danh sách các nh k c a th. Khi ó d dàng xác nh xem nh nào li n k v i nh ã cho, xét m i c nh nh i u nh t hai l n xem có thêm c nh này hay không và nh cu i ã n m trong cây hay ch a. T ó suy ra thu t toán tìm ki m u tiên chi u r ng dùng $O(e)$ ho c $O(n^2)$ b c.

2.7. Tìm ki m u tiên chi u r ng trong th có h ng

Chúng ta c ng có th d dàng thay i c tìm ki m u tiên chi u sâu và tìm ki m u tiên chi u r ng chúng có th ch y khi u vào là các th có h ng. Tuy nhiên, thông tin ra không nh t thi t là cây khung mà có l là r ng khung. Trong c hai thu t toán có th thêm m t c nh ch khi mà nó có h ng i ra t nh ang c th m i t i nh ch a c thêm vào. N u giai o n c a thu t toán mà không có c nh b t u t nh ã c thêm vào t i nh ch a c thêm vào thì c nh t p theo c a vào thu t toán s tr thành g c c a m t cây m i trong r ng khung.

Ví dụ: Cho đồ thị có hướng G. Hãy xác định rừng khung của nó bằng thuật toán tìm kiếm ưu tiên chiều sâu.



Giải: Chúng ta b t u tìm ki m u tiên chi u sâu t i nh a và thêm các nh b, c và g và các c nh t ng ng, n ây thì không i ti p c n a. Khi ó c n quay l i c nh ng ó v n b t c và do v y quay lui t i b. T i ây ch n nh f và e và các c nh t ng ng. V i c quay lui t i p a ta v a. Khi ó xây c cây m i t i d và thêm các nh h, l, k, j và các c nh t ng ng. Khi ó không i

t i p c n a, quay lui v k, sau ó l, r i h, và v d. Cu i cùng chúng ta l i xây cây m i t i i và k t thúc tìm ki m.

3. Cây khung nh nh t



M t l p r t r ng các bài toán có th gi i b ng cách tìm cây khung nh nh t trong m t th có tr ng s sao cho t ng tr ng s c a các c nh c a cây là nh nh t.

nh nh a: Cây khung nh nh t trong m t th liên thông có tr ng s là m t cây khung có t ng tr ng s trên các c nh c a nó là nh nh t.

2.9. Thu t toán tìm cây khung nh nh t

Sau ây trình bày hai thu t toán tìm cây khung nh nh t. C hai u c ti n hành b ng cách ghép các c nh có tr ng s nh nh t trong s các c nh có m t tính ch t nào ó mà ch a c dùng. Nh ng thu t toán này là nh ng ví d v thu t toán tham lam. Thu t toán tham lam là m t th t c th c hi n m t l a ch n t i u m i giai o n. T i u hóa m i giai o n c a thu t toán không m b o t o r a l i g i t i u toàn c c, nh ng hai thu t toán sau ây xây d ng cây khung nh nh t là các thu t toán tham lam t o r a l i g i t i u.

Thu t toán u tiên do Robert Prim a ra n m 1957. th c hi n thu t toán ta b t u b ng vi c ch n m t c nh b t k có tr ng s nh nh t, t nó vào cây khung. L n l t ghép vào cây các c nh có tr ng s t i thi u liên thu c v i m t nh c a cây và không t o ra chu trình trong cây. Thu t toán d ng khi $(n-1)$ c nh ã c ghép vào cây.

procedure PRIM(G : đồ thị liên thông có trọng số với n đỉnh)

$T :=$ cạnh có trọng số nhỏ nhất

For $i := 1$ to $n-2$

Begin

$E :=$ cạnh có trọng số tối thiểu liên thuộc với một đỉnh trong T và không tạo ra chu trình trong T nếu ghép nó vào T

$T := T$ với e được ghép vào

End { T là cây khung nhỏ nhất}

L u ý: Vi c ch n m t c nh ghép vào cây trong m i giai o n c a thu t toán là không xác nh khi có nhi u h n m t c nh cùng tr ng s và th a mãn nh ng tiêu chu n nào ó. C n s p x p các c nh theo m t th t nào ó v i c ch n m t c nh c xác nh. C ng c n chú ý là có nhi u h n m t cây khung nh nh t ng v i m t th liên thông và có tr ng s .

Thu t toán th hai do Joseph Kruskal phát minh vào n m 1956. th c hi n thu t toán này ch n c nh có tr ng s nh nh t c a th . L n l t ghép thêm vào c nh có



tr ng s t i thi u và không t o thành chu trình v i các c nh ã c ch n. Thu t toán d ng sau khi (n-1) c nh ã c ch n.

procedure KRUSKAL(G : đồ thị n đỉnh, liên thông, có trọng số)

$T :=$ đồ thị rỗng

For $i := 1$ to $n-1$

Begin

$E :=$ một cạnh bất kỳ của G với trọng số nhỏ nhất và không tạo ra chu trình trong T , khi ghép nó vào T .

$T := T$ với cạnh e đã được ghép thêm vào.

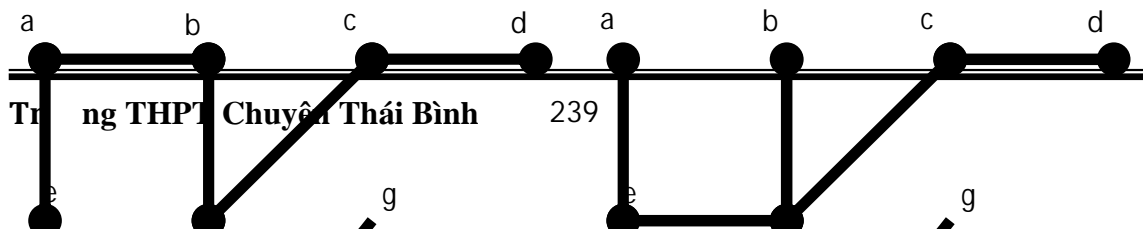
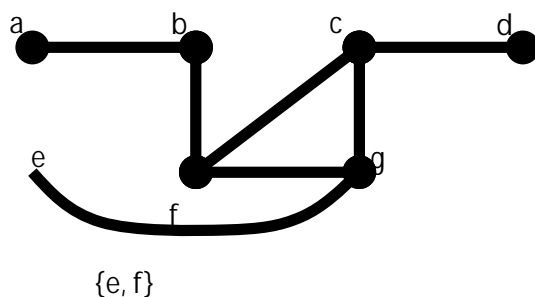
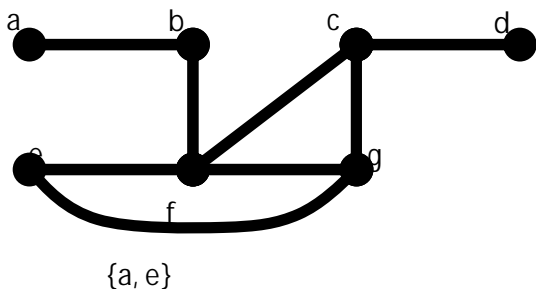
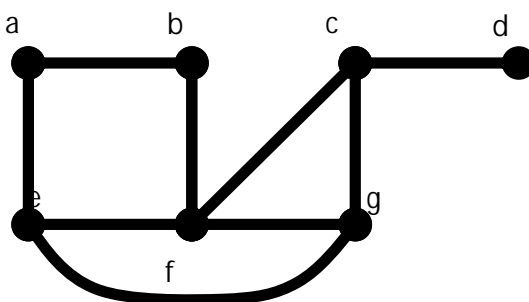
End { T là cây khung nhỏ nhất}

S khác nhau gi a hai thu t toán: Trong PRIM ch n các c nh có tr ng s t i thi u liên thu c v i các nh ã thu c cây và không t o ra chu trình. KRUKAL ch n các c nh có tr ng s t i thi u mà không nh t thi t ph i liên thu c v i các nh c a cây và không t o ra chu trình.

2.10. Bài t p ng d ng:

Bài 1. Cho th nh hình d i bên ph i, tìm cây khung.

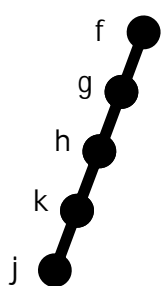
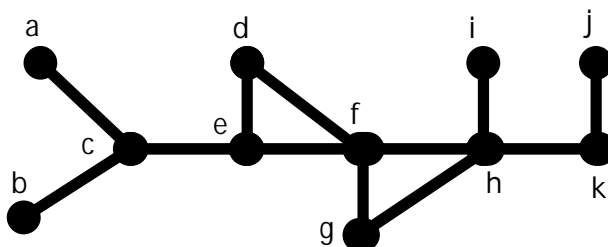
Gi i: th G liên thông, nh ng không ph i là m t cây vì nó ch a chu trình n. Xóa c nh $\{a, e\}$ s lo i c m t chu trình, th con nh n c v n còn liên thông và ch a t t c các nh c a G . Ti p theo xóa c nh $\{e, f\}$ s lo i c m t chu trình n a, cu i cùng xóa c nh $\{c, g\}$ s sinh ra m t th không có chu trình. th này là cây khung vì nó là cây và ch a t t c các nh c a G . áp án c cho b i hình d i đây: **Các cây khung của G cho bởi các hình dưới đây:**



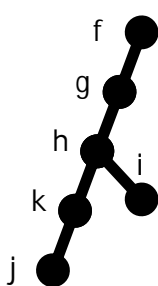


Bài 2. Dùng thuật toán tìm kiếm ưu tiên chi u sâu, tìm cây khung c a th G cho b i hình bên.

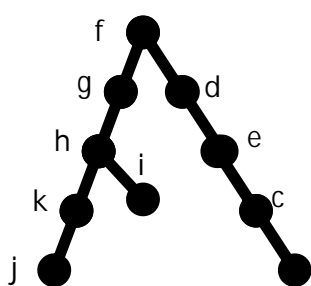
Gi i: Xu t phát t m t nh tùy ý, ví d nh f. ng i c xây d ng b ng cách l n l t ghép càng nhi u càng t t, các c nh liên thu c v i các nh còn ch a thu c ng i, i u ó t o ra c ng i f, g, h, k j. Ti p theo, lùi l i k. Không còn ng i b t u t k ch a các nh ch a c ghé th m. Vì th lùi l i t i h, t h có ng i h, i. Sau ó lùi v h và ti p t c lùi v f. T f có ng i f, d, e, c, a. Ta l i lùi v c và xây d ng ng i c, b. Th t c này ã xây d ng xong cây khung.



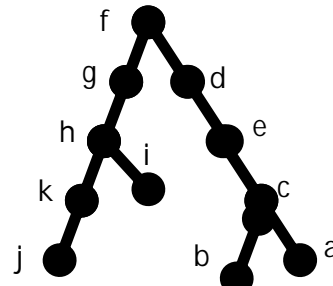
(a)



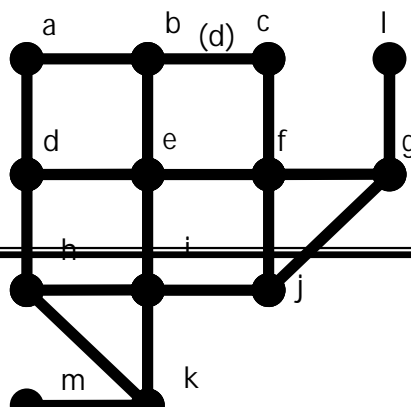
(b)



(c)



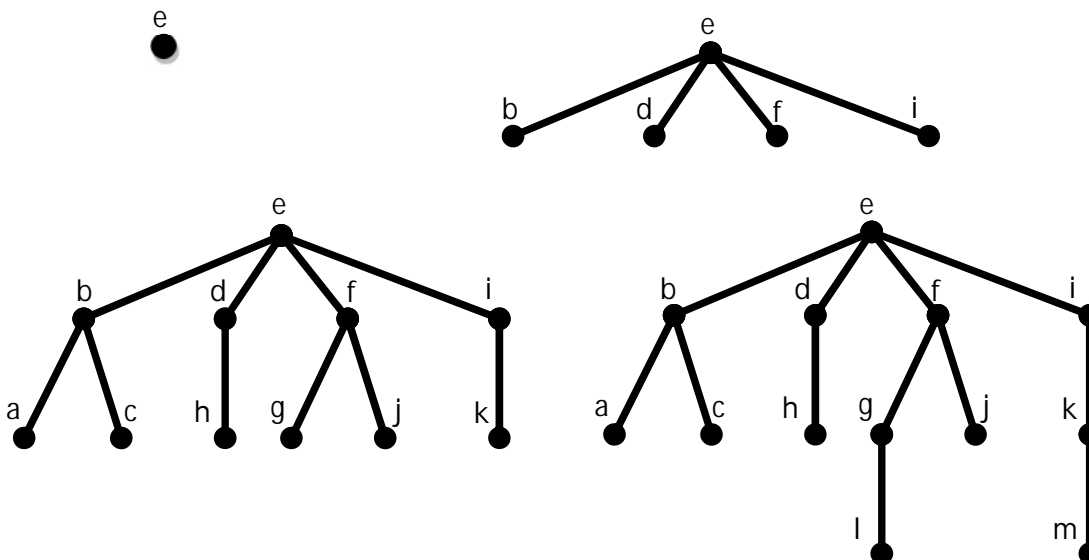
Bài 3. Dùng thuật toán tìm kiếm ưu tiên chi u r ng, tìm





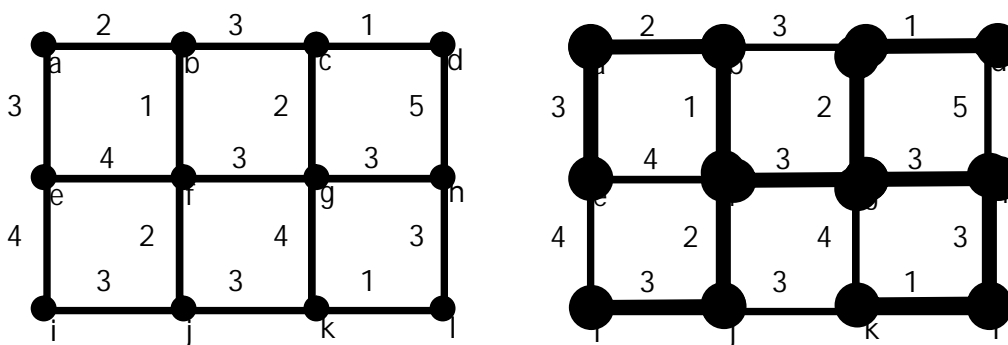
cây khung c a th G cho b i hình bên.

Gi i: Ch n nh e làm g c c a cây, sau ó thêm các c nh liên thu c vào t t c các nh link v i e, t c là các c nh t e t i b, d, f và i c ghép vào. V y m c 1 c a cây có 4 nh. T p theo, ghép các c nh t các nh m c 1 n i v i các nh còn ch a có trong cây. Vì th các c nh t b t i a và c c ghép vào, c ng nh th , các c nh t d t i h, t f t i j và g và t i t i k. Các nh m i a, c, h, g, j, k m c 2 c a cây. T p theo, ghép các c nh t các nh này n i v i các nh còn ch a thu c vào cây, t c là ghép thêm các c nh t g t i l và t k t i m.



Bài 4. Dùng thu t toán PRIM, tìm cây khung nh nh t c a th ã cho nh hình d i ây.

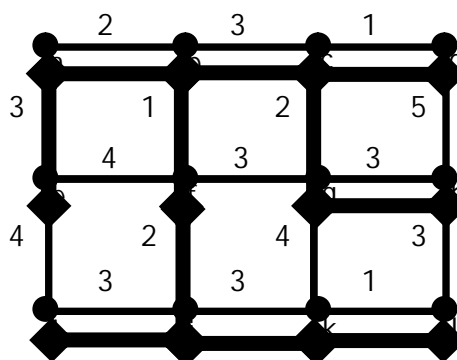
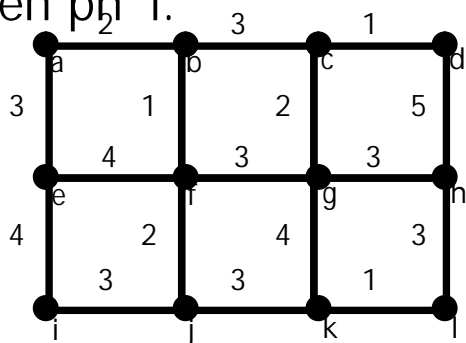
Gi i: Cây khung nh nh t c xây d ng b ng thu t toán PRIM th hi n nh hình d i, bên ph i.



Bài 5. Dùng thu t toán KRUSKAL, tìm cây khung nh nh t c a th ã cho nh hình d i ây.



Gi i: Cây khung nh nh t c xây d ng
b ng thu t toán KRUSKAL th hi n nh hình d i,
bên ph i.





PH L C: M T S CH NG TRÌNH M U THAM KH O.

1. KRUSKAL:

```
#include<iostream>
#include<algorithm>
#define SIZE 100
using namespace std;

struct Edge {
    int beginVertex, endVertex, weight;
};

Edge edges[SIZE*SIZE] =
    {{0,1,6},{0,2,5},{0,3,4},{1,3,8},{2,3,3},{2,4,2},{3,4,1}};
int vertexNumber = 5;
int edgeNumber = 7;

Edge mst[SIZE*SIZE];
int mstNumber;
int parent[SIZE];

bool compareEdge(const Edge& e1, const Edge& e2) {
    return (e1.weight < e2.weight);
}

int findSet(int u) {
    int v = u;
    while(parent[v] >= 0)
        v = parent[v];
    return v;
}

void unionSet(int u,int v) {
    int x = findSet(u);
    int y = findSet(v);
    if (parent[x] > parent[y]){
        parent[y] += parent[x];
        parent[x] = y;
    } else {
        parent[x] += parent[y];
        parent[y] = x;
    }
}

void kruskal() {
    int i, beginRoot, endRoot;

    for(i=0; i<vertexNumber; i++)
        parent[i] = -1;

    sort(edges, edges+edgeNumber, compareEdge);
```



```
for(i = 0; i<edgeNumber; ++i) {
    beginRoot = findSet(edges[i].beginVertex);
    endRoot = findSet(edges[i].endVertex);
    if(beginRoot != endRoot) {

        mst[mstNumber++] = edges[i];
        unionSet(beginRoot, endRoot);
        if(mstNumber == vertexNumber - 1) break;
    }
}

int main() {
    kruskal();

    int i;
    for(i=0; i<mstNumber; ++i)
        cout<<mst[i].beginVertex<<" "<<mst[i].endVertex<<"
"<<mst[i].weight<<endl;
    return 0;
}
```

2. PRIM

```
#include <iostream.h>
#include "Graph.h"
#include "Queue.h"
#include "Tree.h"

#define MAX 200

#define NIL -1
#define VoCung 32765

int Pi[MAX],key[MAX];

void Prim(const GRAPH G,const int r);
void Print(const GRAPH &G,int A[]);
double TiSoTrongSoMST(const GRAPH &G,int A[]);

void main()
{
    int ch,r;
    GRAPH G;
    do{
        cout<<"\n1.Nhap Do Thi.";
        cout<<"\n2.Xuat Do Thi.";
        cout<<"\n3.Prim.";
        cout<<"\n4.Ti So Trong So MST voi DoThi.";
        cout<<"\n0.Thoat.";
        cout<<"\n Ban Chon :";cin>>ch;
        switch(ch)
```



```
{
    case 1:
        cin>>G;
        break;
    case 2:
        cout<<G;
        break;
    case 3:
        cout<<"Nhap Dinh Bat dau :";cin>>r;
        Prim(G,r);
        Print(G,Pi);
        break;
    case 4:
        Prim(G,1);//mac dinh lay dinh dau tien
        cout<<"\n"<<TiSoTrongSoMST(G,Pi)<<"\n";
        break;
}
}while(ch!=0);
}

void Print(const GRAPH &G,int A[])
{
    for(int k=1;k<=G.TongSoDinh;k++)
        if(A[k]!=NIL)
            cout<<"("<<k<<" "<<A[k]<<") ";
}
////////////////////////////////////
double TiSoTrongSoMST(const GRAPH &G,int A[])
{
    int sTrongSoMST = 0;
    for(int k=1;k<=G.TongSoDinh;k++)
        if(A[k]!=NIL)
            sTrongSoMST = sTrongSoMST + G.A[k][A[k]];

    int sTrongSoG = 0;
    for(int i=1;i<=G.TongSoDinh;i++)
        for(int j = i;j<=G.TongSoDinh;j++)
            sTrongSoG = sTrongSoG + G.A[i][j];

    return (1.0*sTrongSoMST/sTrongSoG)*100;
}

void Prim(const GRAPH G,const int r)
{
    int u;
    Queue Q;
    Tree T;

    for(u=1;u<=G.TongSoDinh;u++)
    {
        key[u] =VoCung;
```



```
Pi[u] = NIL;
Q.Them(u);
}

key[r] = 0;
Pi[r] = NIL;

while(Q.isEmpty()!=1)
{
    u = Q.Extract_min(G,T,r); // tiêu chuẩn ưu tiên là cạnh nối đỉnh sẽ
    được bổ sung vào Tree là nhỏ nhất
    for(int v = 1; v<=G.TongSoDinh; v++)
        if(G.A[u][v]!=0)
            if((Q.isContain(v)==1) && (G.A[u][v]<key[v]))
            {
                key[v] = G.A[u][v];
                Pi[v] = u;
            }
    T.Them(u);
}
}
```

3. Breadth First Search

```
#include <iostream>
#include <queue>

using std::cout;
using std::endl;
using std::endl;
using std::cin;

const int maxx = 20;

void Read_input_from_user(bool grid[][maxx], int vertices)
{
    int u, v;
    for(int x = 0; x < vertices; ++x)
    {
        cout << "Enter u : \t";
        cin >> u;
        u--;
        cout << "Enter v : \t";
        cin >> v;
        v--;
        grid[u][v] = true;
        grid[v][u] = true;
        cout << "-----\n";
    }
}

void Breadth_first_search(std::queue<int> &Q, std::vector<int> &trace,
                          bool grid[][maxx], int start, int nodes)
```



```
{
    int u;
    Q.push(start);
    trace[start] = -1;
    do{
        u = Q.front();
        Q.pop();
        for(int v = 0; v < nodes; ++v)
        {
            if((grid[u][v] == true) && trace[v] == 0)
            {
                Q.push(v);
                trace[v] = u;
            }
        }
    }while(!Q.empty());
}

void Trace_result(std::vector<int> &trace, int start, int end, int nodes)
{
    cout << "From _nodes" << start + 1 << " you can visit :\n";
    for(int v = 0; v < nodes; ++v)
    {
        if(trace[v] != 0)
        {
            cout << " _nodes : " << v + 1 << " , ";
        }
    }

    cout << "\n-----\n";
    cout << "The path from " << start + 1 << " to " << end + 1 << '\n';

    if(trace[end] == 0){
        cout << "Unavailable.! to go to from " << end + 1
            << " to -> " << start + 1 << '\n';
    }
    else{
        while(end != start)
        {
            cout << end + 1 << "<-";
            end = trace[end];
        }
        cout << start + 1 << endl;
    }
}

int main()
{
    //Initialization
    std::vector<int> trace(maxx, 0);
}
```



```
std::queue<int> Q;
bool grid[maxx][maxx] = {false};

int nodes, vertices;
cout << "Please input the number of Node : \n";
cin >> nodes;
cout << "Please input the number of Vertices : \n";
cin >> vertices;

//Set value for all vertices.
Read_input_from_user(grid, vertices);

//Read the necessary path
int starting_position, finishing_position;
cout << "Please Input the Starting Node : \n";
cin >> starting_position;
cout << "Please Input the Finishing Node : \n";
cin >> finishing_position;
//Decrease to fit with index of C++ start from 0->size-1
starting_position--;
finishing_position--;
//Algorithm starts
Breadth_first_search(Q, trace, grid, starting_position, nodes);
Trace_result(trace, starting_position, finishing_position, nodes);

return 0;
}
```

4. Depth First Search

/*Bài toán duyệt dinh của đồ thị :

Input : file văn bản path.inp trong đó :

- Dòng 1 chứa số đỉnh n ($n \leq 100$), số cạnh m của đồ thị và dinh xuất phát s , dinh kết thúc f cách nhau 1 dấu cách.
- m dòng tiếp theo, mỗi dòng có dạng 2 số nguyên dương u và v cách nhau 1 dấu cách, thể hiện cạnh nối đỉnh u và v trong đồ thị.

Output :

- Danh sách các dinh có thể đến dc s .
- Đường đi từ $s \rightarrow f$.

Ví dụ:

Path.inp:

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

Path.out

```
#####
#                               #
# From 1 you can visit         #
# 1, 2, 3, 4, 5                #
# The path from 1 -> 5:        #
# 5 <- 3 <- 2 <- 1            #
#                               #
#####
```




```
# 4 6      #
# 7 8      #
#####

*/
#include <iostream>
#include <vector>

using std::cout;
using std::cin;
using std::endl;

const int maxx = 20;

void Read_input_from_user(bool grid[][maxx], int vertices)
{
    int u, v;
    for(int x = 0; x < vertices; ++x)
    {
        cout << "Enter u : \t";
        cin >> u;
        u--;
        cout << "Enter v : \t";
        cin >> v;
        v--;
        grid[u][v] = true;
        grid[v][u] = true;
        cout << "-----\n";
    }
}

void Depth_first_search(bool grid[][maxx], std::vector<int> &trace,
                        int nodes, int u)
{
    for(int v = 0; v < nodes; ++v)
    {
        if((grid[u][v] == true) && (trace[v] == 0))
        {
            trace[v] = u;
            //recursive step
            Depth_first_search(grid, trace, nodes, v);
        }
    }
}

void Trace_result(std::vector<int> &trace, int start, int end, int nodes)
{
    cout << "From _nodes" << start + 1 << " you can visit :\n";
    for(int v = 0; v < nodes; ++v)
    {
        if(trace[v] != 0)
        {
```



```
        cout << " _nodes : " << v + 1 << " , ";
    }
}

cout << "\n-----\n";
cout << "The path from " << start + 1 << " to " << end + 1 << '\n';

if(trace[end] == 0){
    cout << "Unavailable.! to go to from " << end + 1
        << " to -> " << start + 1 << '\n';
}
else{
    while(end != start)
    {
        cout << end + 1 << "<-";
        end = trace[end];
    }
    cout << start + 1 << endl;
}
}

int main()
{
    bool grid[maxx][maxx] = { false };
    std::vector<int> trace(maxx, 0);
    int nodes, vertices;
    cout << "Please input the number of Node : \n";
    cin >> nodes;
    cout << "Please input the number of Vertices : \n";
    cin >> vertices;

    //Set value for all vertices.
    Read_input_from_user(grid, vertices);

    //Read the necessary path
    int starting_position, finishing_position;
    cout << "Please Input the Starting Node : \n";
    cin >> starting_position;
    cout << "Please Input the Finishing Node : \n";
    cin >> finishing_position;
    //Decrease to fit with index of C++ start from 0->size-1
    starting_position--;
    finishing_position--;

    Depth_first_search(grid, trace, nodes, starting_position);
    Trace_result(trace, starting_position, finishing_position, nodes);
    return 0;
}
```



TÀI LIỆU THAM KHẢO

1. Kenneth H. Rosen *Toán học rời rạc ứng dụng trong Tin học* NXB Khoa học và kỹ thuật, Hà Nội 2007.
2. Nguyễn Văn Nghĩa, Nguyễn Tô Thành *Toán học rời rạc* NXB Giáo dục Quốc gia Hà Nội 2003.
3. Bùi Minh Trí *Giáo trình Toán ứng dụng trong Tin học* NXB Giáo dục 2004.
4. Xuân Lôi *Giáo trình Cấu trúc dữ liệu và Giải thuật* NXB Giáo dục 2005.
5. Kenneth H. Rosen *Discrete Mathematics and Its Applications, Fourth Edition* McGraw-Hill Education 1999.
6. Hồ Sĩ Đàm, Hoàng Công, Lê Minh Hoàng, Nguyễn Thanh Hùng *Tài liệu giáo khoa chuyên tin* NXB Giáo dục Việt Nam 2009.