



**ĐẶNG TUẤN THÀNH**  
(Sưu tầm và biên soạn)

# KỸ THUẬT LẬP TRÌNH

## Quyển 2



*Tháng 01 năm 2021*



## LỜI NÓI ĐẦU

Nhằm cung cấp tài liệu dạy học môn Tin học trong trường phổ thông và bồi dưỡng học sinh giỏi môn Tin học, tác giả sưu tầm và biên soạn 02 cuốn học liệu: **“21 giờ học lập trình”** và **“Kỹ thuật lập trình”**.

Tác giả luôn khắc ghi và biết ơn sự quan tâm của các đơn vị Phòng Giáo dục, trường THCS, trường THPT trong Tỉnh đối với việc áp dụng nguồn học liệu này và đã có những ý kiến góp ý quý báu, kịp thời để tác giả hoàn thiện học liệu.

Nguồn học liệu mới này được viết bởi ngôn ngữ lập trình Pascal và C++. Nội dung gồm **20** bài tập chuyên đề Duyệt đồ thị. Hệ thống kiến thức được trình bày từ dễ đến khó, tạo điều kiện thuận lợi cho việc tự học, tự nghiên cứu. Với mỗi đơn vị kiến thức, có các bài tập, hướng dẫn thuật toán và code chương trình tham khảo bằng Pascal và C++ giúp người học nhanh chóng cài đặt thành công thuật toán mới.

Trong thời gian tới, tác giả viết tiếp tài liệu mới **“Ngôn ngữ lập trình C++”** với các chuyên đề về ngôn ngữ lập trình C++ và bài tập ứng dụng để thay thế dần ngôn ngữ Pascal. Rất mong các bạn đón đọc.

Tài liệu này được biên soạn rất tỉ mỉ nhưng không tránh được những thiếu sót, tác giả rất mong được sự đóng góp nội dung, phản hồi của bạn đọc vào email: [dtthanh.c3ntt@yenbai.edu.vn](mailto:dtthanh.c3ntt@yenbai.edu.vn).

**Tác giả**



## Mục lục

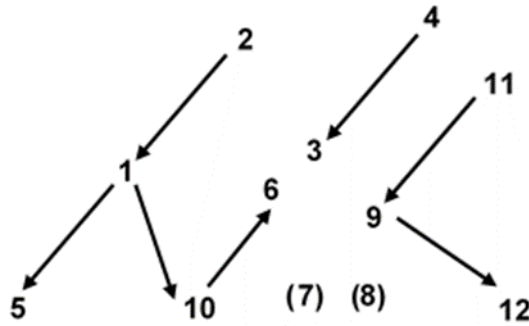
Một số khái niệm cơ bản về đồ thị .....	5
1. Đếm số vùng liên thông.. DFSVLT.* .....	8
2. Liệt kê các vùng liên thông... DFSNVLT.* .....	14
3. Tìm vùng liên thông có nhiều đỉnh nhất.. DFSMVLT.* .....	21
4. Liệt kê các đỉnh đến được từ đỉnh S.. DFSLVLT.* .....	29
5. Tìm đường đi từ đỉnh S đến đỉnh T.. DFSSTVLT.* .....	36
6. Xây cầu.. DFSBRIGE.* .....	40
7. Đường đi từ S.. DFSTRACE.* .....	44
8. Đường đi từ S qua nhiều đảo nhất.. DFSLMAX.* .....	47
9. Tham quan K đảo.. DFSLK.* .....	54
10. Hành trình từ S tới T khi vài đảo bị phong tỏa.. DFSSECUR.* .....	60
11. Các vùng liên thông có tổng lớn nhất.. DFSSMAX.* .....	65
12. Vùng nguyên tố lớn nhất.. DFSPRIME.* .....	70
13. Vùng có tổng chi phí nhỏ hơn hoặc bằng S.. DFSTS.* .....	76
14. Vé miễn phí.. DFSTK.* .....	80
15. Xây cầu 2.. DFSBRG2.* .....	85
16. Đường đi trên đồ thị có hướng.. DFSORIEN.* .....	90
17. Đường đi từ S tới T và ngược lại.. DFSOST.* .....	94
18. Tìm điểm hẹn.. DFSOST2.* .....	99
19. Đường truyền an toàn.. DFSSAFE.* .....	105
20. Điểm đến an toàn.. DFSSAFE2.* .....	110

## I. Một số khái niệm cơ bản về đồ thị

Một đồ thị kí hiệu là  $G = (V, E)$

Trong đó:

- $V$  là tập các đỉnh của đồ thị. Kí hiệu  $|V| = N$  là số đỉnh của đồ thị.
- $E$  là tập các cạnh của đồ thị. Kí hiệu  $|E| = M$  là số cạnh của đồ thị.



### Đỉnh:

Đỉnh biểu diễn các đối tượng trong đồ thị, thường được đánh dấu bằng các số 1, 2, ... hoặc kí hiệu bằng các chữ cái in thường  $u, v, \dots$

### Cạnh:

Cạnh nối đỉnh  $x$  với đỉnh  $y$  là một tập gồm hai phần tử  $(x, y)$ .  $(x, y)$  thường được vẽ dưới dạng một *đoạn thẳng* nối hai đỉnh.

### Cạnh có hướng (cung):

Là một cặp đỉnh có thứ tự. Trong mỗi cặp có thứ tự đó, đỉnh thứ nhất được gọi là đỉnh đầu, đỉnh thứ hai là đỉnh cuối.

### Cạnh vô hướng:

Không quan tâm đến hướng và coi hai đỉnh như nhau.

### Khuyên:

Là một cạnh nối một đỉnh với chính nó.

### Hai cạnh song song:

Là hai cạnh cùng nối hai đỉnh  $u, v$ .

### Đồ thị có hướng:

Là đồ thị mà tất cả các cạnh trong đồ thị đều có hướng.

### Đồ thị vô hướng:

Là đồ thị mà tất cả các cạnh trong đồ thị đều vô hướng.

**Đơn đồ thị:**

Là đồ thị không có khuyên và không có cạnh song song.

**Đa đồ thị:**

Là đồ thị không phải là đơn đồ thị.

**Bậc:**

Trong đồ thị vô hướng, bậc của đỉnh  $v$  trong đồ thị  $G$ , ký hiệu  $d_G(v)$ , là số cạnh liên thuộc với  $v$ , trong đó, khuyên được tính hai lần.

Ta có định lý:

Giả sử  $G=(V,E)$  là đồ thị vô hướng, khi đó tổng các bậc đỉnh trong  $V$  sẽ bằng 2 lần số cạnh.

$$\sum_{v \in V} d_G(v) = m * 2$$

**Hệ quả:** Trong đồ thị vô hướng, số đỉnh bậc lẻ là chẵn.

Trong đồ thị có hướng, ta định nghĩa **bán bậc ra** của  $u$  là số cung đi ra khỏi nó, kí hiệu  $d^+_G(u)$ , **bán bậc vào** của  $u$  là số cung đi vào đỉnh đó, kí hiệu  $d^-_G(u)$ .

Giả sử  $G=(V, E)$  là đồ thị có hướng, khi đó tổng các bán bậc vào bằng tổng các bán bậc ra và bằng số cung của đồ thị.

**Đường đi và chu trình:**

Một dãy các đỉnh  $P = (p_0, p_1, \dots, p_k)$  sao cho  $(p_{i-1}, p_i) \in E, \forall i: 1 \leq i \leq k$  được gọi là một đường đi.

Một đường đi là chu trình khi  $p_0 = p_k$ .

**Liên thông:**

Một đồ thị vô hướng là liên thông nếu tồn tại đường đi giữa hai cặp đỉnh bất kì thuộc đồ thị.

Một đồ thị có hướng là liên thông yếu nếu phiên bản vô hướng của đồ thị đó là liên thông.

## Biểu diễn đồ thị trên máy tính

Có nhiều cách để biểu diễn đồ thị trên máy tính, tùy thuộc vào tính chất của đồ thị hoặc thuật toán áp dụng với đồ thị... Ta cũng có thể lưu kèm theo các thông tin như trọng số, giá trị phù hợp với từng cạnh.

*\* Biểu diễn đồ thị theo ma trận kề:*

Tạo một ma trận  $A$  kích thước  $n \times n$  trong đó  $n$  là số đỉnh của đồ thị.

Nếu là đơn đồ thị, vô hướng không trọng số:

$a[u][v] = 0$  nếu không có cạnh nối hai đỉnh  $u, v$ .

$a[u][v] = a[v][u] = 1$  nếu có cạnh nối hai đỉnh  $u, v$ .

Nếu đồ thị là đa đồ thị vô hướng không trọng số, tương tự, chỉ khác ta có thể gán  $a[u][v] = \text{số cạnh nối } u \text{ và } v$ .

Nếu đồ thị vô hướng có trọng số thì chúng ta thay  $a[u][v] = 1$  và  $a[v][u] = 1$  đó thành  $a[u][v] = a[v][u] = c$  (với  $c$  chính là trọng số của cạnh đó).

Nếu đồ thị có hướng, dữ liệu đề bài nói rõ cạnh đi từ  $u$  đến  $v$  thì chỉ gán  $a[u][v] = 1$  và không gán chiều ngược lại là được.

Định nghĩa và gán tùy theo lập trình viên hiểu là vô hướng hay có hướng, đơn đồ thị hay đa đồ thị.

*\* Một trong các thuật toán tìm kiếm trên đồ thị là: DFS*

## II. Bài tập áp dụng thuật toán duyệt đồ thị DFS

### 1. Đếm số vùng liên thông.. DFSVLT.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

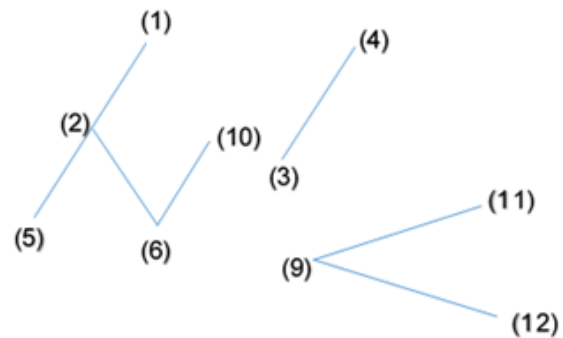
Cho đồ thị vô hướng có N đỉnh, M cạnh. Hãy đếm số lượng vùng liên thông trong đồ thị.

**Dữ liệu:** Vào từ file văn bản DFSVLT.INP gồm:

- Dòng 1: Ghi số nguyên N và M ( $M, N \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSVLT.OUT số lượng vùng liên thông.

DFSVLT.INP	DFSVLT.OUT
12 7 1 2 2 5 2 6 6 10 3 4 9 11 9 12	5 (gồm 3 vùng bên, đỉnh 7, đỉnh 8)



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ u tới v.

$a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp từ u tới v.

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ i, đánh dấu i đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh j đang free và có đường đi trực tiếp từ i đến j thì ta Duyệt ( j ).

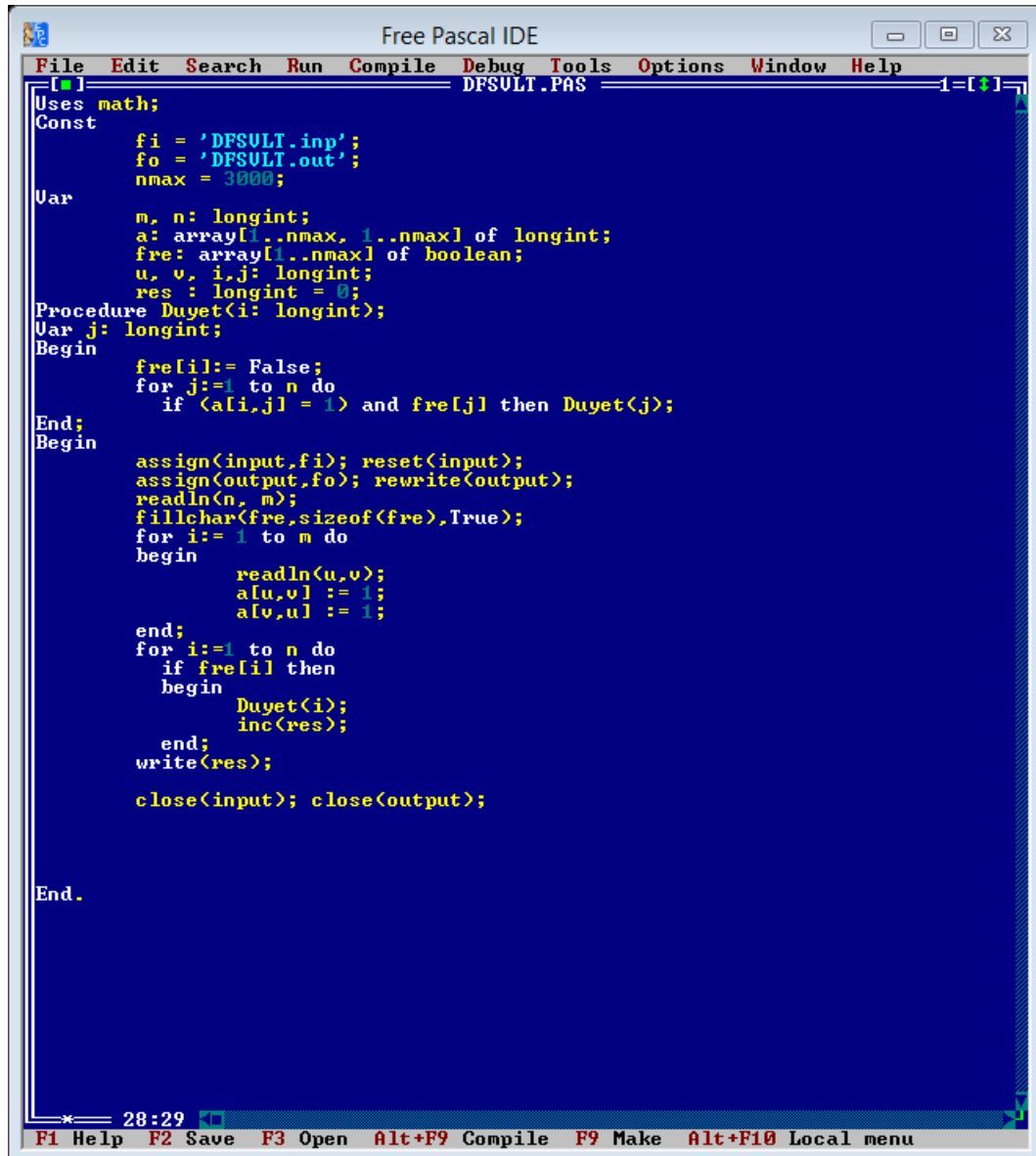
Tới đỉnh j, ta lặp lại việc duyệt như ở bước trên.



**Lưu ý:** Với mỗi đỉnh  $i$ , ta sẽ đi đến được tất cả các đỉnh có đường đi trực tiếp hoặc gián tiếp tới  $i$ . Như vậy, mỗi lần duyệt ( $i$ ) xong ta sẽ thu được một vùng liên thông. Ở bài này, ta tăng biến đếm số lượng vùng liên thông lên.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:



```
Free Pascal IDE
DFSULT.PAS

Uses math;
Const
  fi = 'DFSULT.inp';
  fo = 'DFSULT.out';
  nmax = 3000;

Var
  m, n: longint;
  a: array[1..nmax, 1..nmax] of longint;
  fre: array[1..nmax] of boolean;
  u, v, i, j: longint;
  res: longint = 0;

Procedure Duyệt(i: longint);
Var j: longint;
Begin
  fre[i] := False;
  for j := 1 to n do
    if (a[i, j] = 1) and fre[j] then Duyệt(j);
  end;
End;

Begin
  assign(input, fi); reset(input);
  assign(output, fo); rewrite(output);
  readln(n, m);
  fillchar.fre, sizeof(fre), True;
  for i := 1 to m do
    begin
      readln(u, v);
      a[u, v] := 1;
      a[v, u] := 1;
    end;
  for i := 1 to n do
    if fre[i] then
      begin
        Duyệt(i);
        inc(res);
      end;
  write(res);

  close(input); close(output);

End.
```

\* 28:29

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

## Code mẫu C++: Code1

```
D:\THANG82020\bailam\loc\DFSVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSVLT.cpp
1  #include <bits/stdc++.h>
2  #define nmax 100005
3  using namespace std;
4  int lab[nmax];
5  int findset(int u)
6  {
7      return lab[u]<0 ? u:lab[u]=findset(lab[u]);
8  }
9  void uniona(int r,int s)
10 {
11     if (lab[s]<lab[r]) swap (r,s);
12     lab[r]+=lab[s];
13     lab[s]=r;
14 }
15 void inp()
16 {
17     int n,m;
18     cin >> n >> m;
19     int cc=n;
20     memset(lab,-1,sizeof(lab));
21     for (int i=1;i<=m;i++)
22     {
23         int u,v;
24         cin >> u >> v;
25         int r=findset(u);
26         int s=findset(v);
27         if (r!=s)
28         {
29             uniona(r,s);
30             --cc;
31         }
32     }
33 }
C++ source file
```

```
36 int main()
37 {
38     freopen("DFSVLT.INP","r",stdin);
39     freopen("DFSVLT.OUT","w",stdout);
40     ios_base::sync_with_stdio(false);
41     cin.tie(NULL);
42     cout.tie(NULL);
43     inp();
44     return 0;
45 }
46
C++ source file
```

## Code mẫu C++ code 2:

```
D:\THANG82020\bailam\bac\DFSFLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSFLT.cpp
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  bool a[1001][1001];
6  bool check[1001];
7  int n, m, lab=0;
8
9  void dfs(int i)
10 {
11     check[i] = true;
12     for (int j = 1; j <= n; j++)
13         if (a[i][j] && !check[j])
14             {
15                 check[j] = true;
16                 dfs(j);
17             }
18 }
19 int main()
20 {
21     ios_base::sync_with_stdio(false);
22     cin.tie(NULL);
23     cout.tie(NULL);
24
25     freopen("DFSFLT.INP", "r", stdin);
26     freopen("DFSFLT.OUT", "w", stdout);
27
28     cin >> n >> m;
29     for (int i=1; i<=m; i++)
30     {
31         int u, v;
32         cin >> u >> v;
33         a[u][v] = 1;
34         a[v][u] = 1;
35     }
36     for (int i=1; i<=n; i++)
37         if (!check[i])
38             {
39                 lab++;
40                 dfs(i);
41             }
42     cout << lab;
43     return 0;
44 }
45
C++ source file
```

Code 3 tham khảo:

```
*D:\THANG82020\bailam\hai\DFSVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSVLT.cpp x
1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSVLT"
18 #define nmax 5005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m;
23 Vi g[nmax];
24 bool check[nmax];
25
```

```
25
26 int read(){
27     int res = 0;
28     char c = getchar();
29     bool Neg = false;
30     while(c == ' ' || c == '\n') c = getchar();
31     if(c == '-'){
32         Neg = true;
33         c = getchar();
34     }
35     while('0' <= c && c <= '9'){
36         res = res * 10 + c - '0';
37         c = getchar();
38     }
39     if(Neg) return -res;
40     return res;
41 }
42
```

```
44 void dfs(int u, int par){
45     check[u] = 1;
46     for(auto xx : g[u]){
47         if(xx != par) dfs(xx,u);
48     }
49 }
50 void kurumi(){
51     n = read();
52     m = read();
53     for(int i = 1; i <= m; i++){
54         int u,v;
55         u = read();
56         v = read();
57         g[u].pb(v);
58         g[v].pb(u);
59     }
60     ll Shido = 0;
61     for(int i = 1; i <= n; i++){
62         if(check[i] == 0){
63             Shido++;
64             dfs(i,0);
65         }
66     }
67     cout << Shido;
68 }
69
70 int main()
71 {
72     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
73     freopen(Kurumi".inp","r",stdin);
74     freopen(Kurumi".out","w",stdout);
75     kurumi();
76     return 0;
77 }
78
```

C++ source filelength : 1,651 lines : 78

## 2. Liệt kê các vùng liên thông... DFSNVLT.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

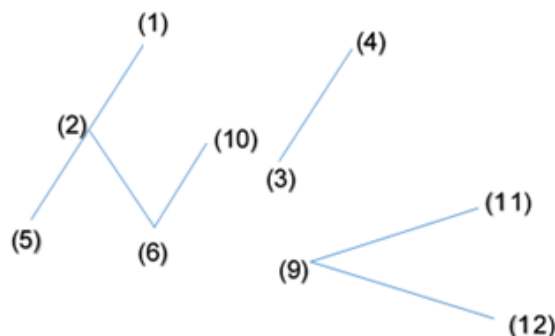
Cho đồ thị vô hướng có N đỉnh, M cạnh. Hãy liệt kê các vùng liên thông trong đồ thị. Trong mỗi vùng các đỉnh được sắp xếp thành dãy tăng. Mỗi vùng liên thông trên một hàng. Mỗi số cách nhau một dấu cách.

**Dữ liệu:** Vào từ file văn bản DFSNVLT.INP gồm:

- Dòng 1: Ghi số nguyên N và M ( $M, N \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSNVLT.OUT gồm nhiều dòng là các vùng liên thông trong đồ thị. Trong mỗi vùng các đỉnh được sắp xếp thành dãy tăng. Mỗi số cách nhau một dấu cách, mỗi vùng liên thông trên một hàng.

DFSNVLT.INP	DFSNVLT.OUT
12 7	1 2 5 6 10
1 2	3 4
2 5	7
2 6	8
6 10	9 11 12
3 4	
9 11	
9 12	



### Thuật toán:

Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

Mỗi lần Duyệt xong một vùng liên thông, ta đưa vùng đó ra và khởi tạo lại  $K=0$ ; Duyệt theo chiều sâu luôn tạo ra các vùng liên thông theo thứ tự từ điển. Do đó cần phải sắp xếp lại mảng C

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSMULT.PAS
Uses math;
Const
  fi = 'DFSMULT.inp';
  fo = 'DFSMULT.out';
  nmax = 3000;
Var
  m, n: longint; u, v, i, j: longint; k: longint = 0; // <-----
  a: array[1..nmax, 1..nmax] of longint;
  fre: array[1..nmax] of boolean;
  c: array[1..nmax] of longint;
Procedure Duyet(i: longint);
Var j: longint;
Begin
  fre[i] := False;
  inc(k); // <-----
  c[k] := i; // <-----
  for j := 1 to n do
    if (a[i, j] = 1) and fre[j] then Duyet(j);
End;
Procedure Sapxep();
Var i, j, temp: longint;
Begin
  For i := 1 to k-1 do
    for j := i+1 to k do
      if c[i] > c[j] then
        begin
          temp := c[i];
          c[i] := c[j];
          c[j] := temp;
        end;
End;
Begin
  assign(input, fi); reset(input);
  assign(output, fo); rewrite(output);
  readln(n, m);
  fillchar(fre, sizeof(fre), True);
  for i := 1 to m do
    begin
      readln(u, v);
      a[u, v] := 1;
      a[v, u] := 1;
    end;
  for i := 1 to n do
    if fre[i] then
      begin
        Duyet(i);
        sapxep();
        for j := 1 to k do write(c[j], #32);
        writeln();
        k := 0;
      end;
  close(input); close(output);
End.
*== 11:1 ==
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

```

Do các đỉnh cần sắp xếp lại, nên bạn cần code thêm đoạn sắp xếp mảng C trước khi ghi ra kết quả.

Code 1 tham khảo:

```
"D:\THANG82020\bailam\ha\DFSNVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSNVLT.cpp
1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSNVLT"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m;
23 Vi g[nmax];
24 Vi gg;
25 bool check[nmax];
26
27 int read(){
28     int res = 0;
29     char c = getchar();
30     bool Neg = false;
31     while(c == ' ' || c == '\n') c = getchar();
32     if(c == '-'){
33         Neg = true;
34         c = getchar();
35     }
36     while('0' <= c && c <= '9'){
37         res = res * 10 + c - '0';
38         c = getchar();
39     }
40     if(Neg) return -res;
41     return res;
42 }
43
44 void dfs(int u, int par){
45     gg.pb(u);
46     check[u] = 1;
47     for(auto xx : g[u]){
48         if(xx != par && check[xx] == 0) dfs(xx,u);
49     }
50 }
51
52
C++ source file
```



```

44
45 void dfs(int u, int par){
46
47     gg.pb(u);
48     check[u] = 1;
49     for(auto xx : g[u]){
50         if(xx != par && check[xx] == 0) dfs(xx,u);
51     }
52 }
53
54 void kurumi(){
55     cin >> n >> m;
56
57     for(int i = 1; i <= m; i++){
58         int u,v;
59         cin >> u >> v;
60         g[u].pb(v);
61         g[v].pb(u);
62     }
63     for(int i = 1; i <= n; i++){
64         if(check[i] == 0){
65             gg.clear();
66             dfs(i,0);
67             sort(gg.begin(), gg.end());
68             for(auto xx : gg){
69                 cout << xx << ' ';
70             }
71             cout << '\n';
72         }
73     }
74 }
75
76 int main()
77 {
78     ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
79     freopen("Kurumi".inp, "r", stdin);
80     freopen("Kurumi".out, "w", stdout);
81     kurumi();
82     return 0;
83 }
84

```

C++ source file length: 1,783 lines: 1

Code 2 tham khảo:

```

1 #include <bits/stdc++.h>
2 #define task "dfsnavlt"
3 using namespace std;
4 vector <int> ke[3009];
5 vector <int> vlt[3009];
6 bool check[3009];
7 int u,v,m,n,dem;
8 void dfs(int u, int chiso)
9 {
10     vlt[chiso].push_back(u);
11     check[u]=1;
12     for(int t: ke[u])
13     {
14         if (check[t]==0) dfs(t,chiso);
15     }
16 }
17

```

```

17
18 int main()
19 {
20     freopen(task".inp", "r", stdin);
21     freopen(task".out", "w", stdout);
22     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
23     cin >> n >> m;
24     for(int i=1; i<=m; i++)
25     {
26         cin >> u >> v;
27         ke[u].push_back(v);
28         ke[v].push_back(u);
29     }
30     for(int i=1; i<=n; i++)
31         if (check[i]==0)
32     {
33         dem++;
34         dfs(i, dem);
35     }
36     for(int i=1; i<=dem; i++) sort(vlt[i].begin(), vlt[i].end());
37     for(int i=1; i<=dem; i++)
38     {
39         for(int u: vlt[i]) cout << u << ' ';
40         cout << '\n';
41     }
42     return 0;
43 }
44

```

C++ source file length : 903 lines : 4

Code 3 tham khảo:

```

D:\THANG82020\bailam\aduc\DFSNVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run
DFS NVLT.cpp
1 #include <bits/stdc++.h>
2 #define nmax 100005
3
4 using namespace std;
5
6 vector <int> g[nmax];
7 vector <int> result;
8 vector <int> visit(nmax, 0);
9 int res = 0;
10
11 void dfs(int u)
12 {
13     visit[u] = 1;
14     result.push_back(u);
15     for(int j=0; j<g[u].size(); j++)
16     {
17         int v = g[u][j];
18         if(visit[v]==0) {
19             dfs(v);
20         }
21     }
22 }

```

```

23
24 void solve()
25 {
26     int n,m;
27     cin >> n >> m;
28     for(int i=1;i<=m;i++)
29     {
30         int u,v;
31         cin >> u >> v;
32         g[u].push_back(v);
33         g[v].push_back(u);
34     }
35     for(int i=1;i<=n;i++)
36     {
37         if(visit[i]==0){
38             dfs(i);
39             sort(result.begin(),result.end());
40             for(int i=0;i<result.size();i++) cout << result[i] << ' ';
41             cout << '\n';
42             result.clear();
43         }
44     }
45 }
46
47 int main()
48 {
49     freopen("DFS NVLT.inp","r",stdin);
50     freopen("DFS NVLT.out","w",stdout);
51     ios_base::sync_with_stdio(0);
52     cin.tie(NULL);cout.tie(NULL);
53     solve();
54     return 0;
55 }
56

```

C++ source file

#### Code 4 tham khảo:

```
D:\THANG82020\bailam\loc\DFSNVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins
DFSNVLT.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a[3001][3001];
5
6  int n, m, Free[3001], u, v;
7  vector<int> g;
8  void DFS(int u)
9  {
10     g.push_back(u);
11     Free[u] = false;
12     for (int v = 1; v <= n; v++)
13         if (a[u][v] == 1 && Free[v])
14             DFS(v);
15 }
16
17 int main()
18 {
19     freopen("DFSNVLT.INP", "r", stdin);
20     freopen("DFSNVLT.OUT", "w", stdout);
21     ios_base::sync_with_stdio(false);
22     cin.tie(NULL);
23     cout.tie(NULL);
24     cin >> n >> m;
25     for (int i = 1; i <= n; i++)
26         for (int j = 1; j <= n; j++)
27             a[i][j] = 0;
28
29     for (int i = 1; i <= m; i++)
30     {
31         cin >> u >> v;
32         a[u][v] = 1;
33         a[v][u] = 1;
34     }
35
36     for (int i = 1; i <= n; i++)
37         Free[i] = 1;
38     for (int i = 1; i <= n; i++)
39     {
40         g.clear();
41         if (Free[i] == 1)
42         {
43             DFS(i);
44             if (i != 1) cout << endl;
45         }
46         sort(g.begin(), g.end());
47         for (auto i : g)
48             cout << i << " ";
49     }
50     return 0;
51 }
52
```

C++ source file

### 3. Tìm vùng liên thông có nhiều đỉnh nhất.. DFSMVLT.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

Cho đồ thị vô hướng có N đỉnh, M cạnh. Hãy liệt kê vùng liên thông có nhiều đỉnh nhất trong đồ thị. Mỗi số cách nhau một dấu cách, nếu có nhiều vùng liên thông có cùng số lượng đỉnh nhiều nhất thì đưa ra vùng liên thông có thứ tự từ điển nhỏ nhất.

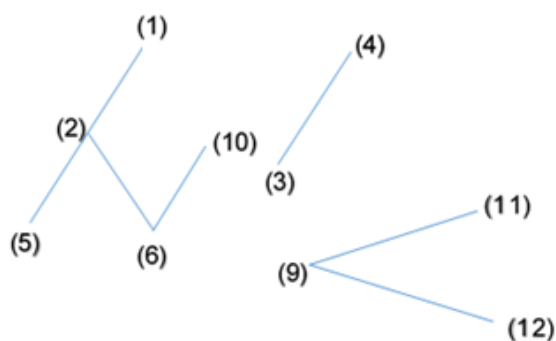
**Dữ liệu:** Vào từ file văn bản DFSMVLT.INP gồm:

- Dòng 1: Ghi số nguyên N và M ( $M, N \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSMVLT.OUT gồm:

- Dòng 1: Ghi số lượng đỉnh trong vùng liên thông tìm được.
- Dòng 2: Các đỉnh trong vùng liên thông tìm được, được sắp xếp thành dãy tăng. Mỗi số cách nhau một dấu cách.

DFSMVLT.INP	DFSMVLT.OUT
12 7	5
1 2	1 2 5 6 10
2 5	
2 6	
6 10	
3 4	
9 11	
9 12	



#### Thuật toán:

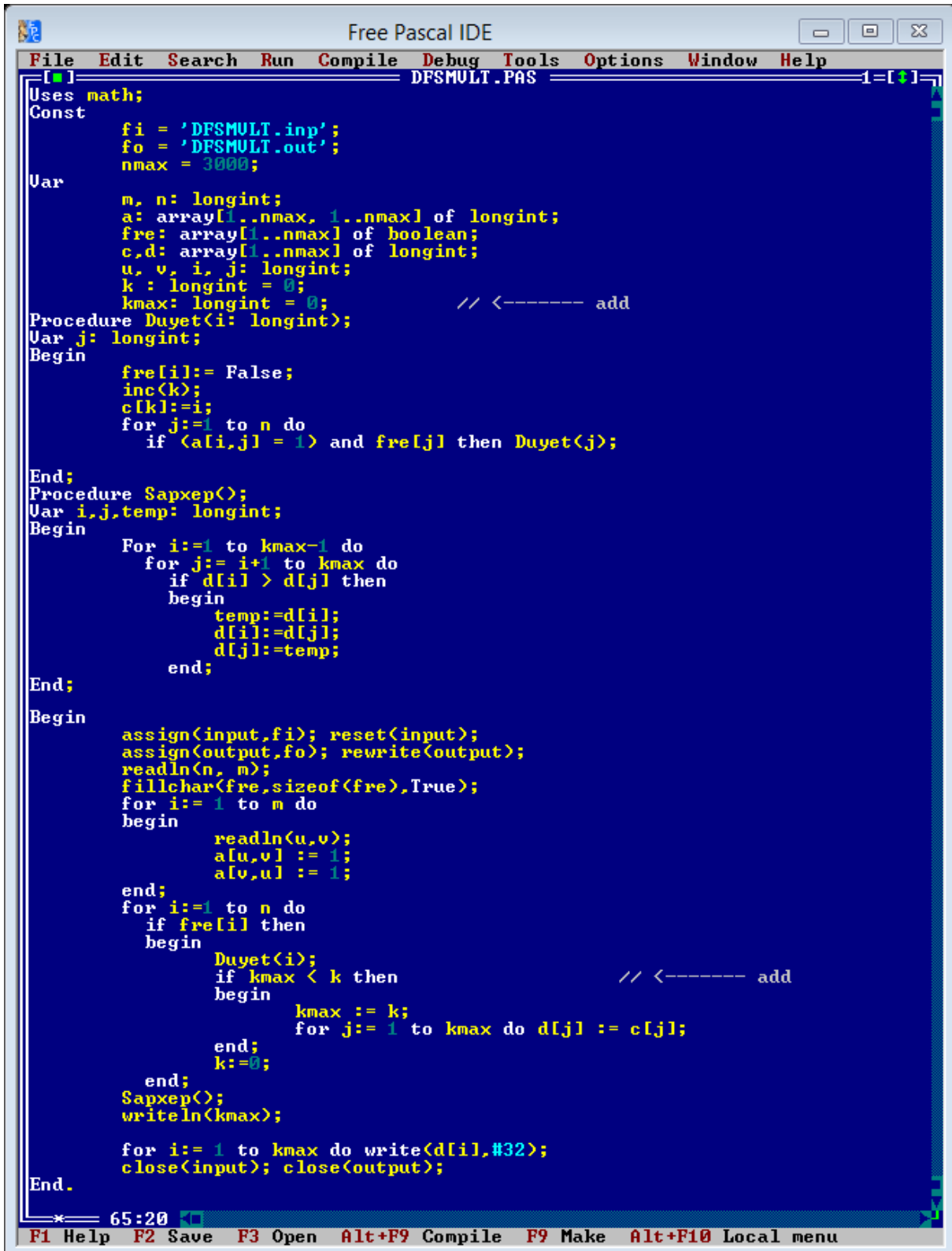
Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

Mỗi lần Duyệt xong một vùng liên thông, ta cập nhật lại kmax và lưu mảng C sang mảng D và khởi tạo lại K = 0.

Ta đưa kmax và mảng D sau khi sắp xếp ra.

Code chương trình tham khảo:



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSMULT.PAS
1=[ ]
Uses math;
Const
    fi = 'DFSMULT.inp';
    fo = 'DFSMULT.out';
    nmax = 3000;
Var
    m, n: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    c, d: array[1..nmax] of longint;
    u, v, i, j: longint;
    k: longint = 0;
    kmax: longint = 0;           // <----- add
Procedure Duyet(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    inc(k);
    c[k] := i;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then Duyet(j);
End;
Procedure Sapxep();
Var i, j, temp: longint;
Begin
    For i := 1 to kmax-1 do
        for j := i+1 to kmax do
            if d[i] > d[j] then
                begin
                    temp := d[i];
                    d[i] := d[j];
                    d[j] := temp;
                end;
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m);
    fillchar(fre, sizeof(fre), True);
    for i := 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
            a[v, u] := 1;
        end;
    for i := 1 to n do
        if fre[i] then
            begin
                Duyet(i);
                if kmax < k then           // <----- add
                    begin
                        kmax := k;
                        for j := 1 to kmax do d[j] := c[j];
                    end;
                k := 0;
            end;
    Sapxep();
    writeln(kmax);

    for i := 1 to kmax do write(d[i], #32);
    close(input); close(output);
End.
* 65:20
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```

Code 1 tham khảo:

```
D:\THANG82020\bailam\trung\DFSMVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSMVLT.cpp
1
2 #include <bits/stdc++.h>
3 using namespace std;
4 #define nmax 3001
5 int n,m,a[3001][3001],u,v,b[100],t=0,maxx=0,s;
6 bool visit[nmax];
7 bool check[nmax];
8 int res=0;
9
10 void dfs(int i)
11 {
12     visit[i]=true;
13     check[i]=true;
14     for(int v=1;v<=n;v++)
15     {
16         if(a[i][v]==true&& visit[v]==false)
17             dfs(v);
18     }
19 }
20
21 int main()
22 {
23     freopen("DFSMVLT.INP","r",stdin);
24     freopen("DFSMVLT.OUT","w",stdout);
25     cin>>n>>m;
26     for(int i=1;i<=m;i++)
27     {
28         cin >> u >> v;
29         a[u][v]=1;
30         a[v][u]=1;
31     }
32     vector <int> c;
33     for(int i=1;i<=n;i++)
34     {
35         if(visit[i]==false)
36         {
37             res = 0;
38             memset(check,0,sizeof(check));
39             dfs(i);
40             for(int j=1;j<=n;j++)
41                 if(check[j])
42                     res++;
43             if(maxx < res)
44                 {
```

```

45         c.clear();
46         maxx = res;
47         for(int f=1;f<=n;f++)
48         {
49             if(check[f])
50                 c.push_back(f);
51         }
52     }
53 }
54 }
55 cout << maxx << '\n';
56 for(int i = 0; i < c.size(); i++)
57     cout << c[i] << ' ';
58
59 return 0;
60 }
61

```

C++ source file length : 1,235

Code 2 tham khảo:

```

1
2 #include <bits/stdc++.h>
3 #define ll long long
4 // #define ford(x,a,b) for(ll x=a;x>=b;x--)
5 #define for(x,a,b) for(ll x=a;x<=b;x++)
6
7 #define task "DFSMVLT"
8 const ll oo=1e18;
9 const ll mod=1e9;
10 const ll nmax=1e4+7;
11 int n,t,u,v,res=0,temp,last,head;
12 ll mx=-oo;
13 bool fre[nmax],a[nmax][nmax],trace[nmax],trace2[nmax];
14
15 using namespace std;
16
17 void isReadFile()
18 {
19     freopen(task".INP","r",stdin);
20     freopen(task".OUT","w",stdout);
21 }
22
23

```



```

23
24 void dfs(int i) {
25     fre[i] = 0;
26     for(j,1,n) {
27         if ((fre[j]) && (a[i][j])) {
28             trace[j] = 1;
29             dfs(j);
30         }
31     }
32 }
33
34 void loli()
35 {
36     memset(fre,1,sizeof(fre));
37     cin >> n >> t;
38     for(i,1,t) {
39         cin >> u >> v;
40         a[u][v]=1;
41         a[v][u]=1;
42     }
43     for(k,1,n) {
44         if (fre[k]==0) continue;
45         memset(trace,0,sizeof(trace));
46         dfs(k);
47

```

C++ source file

```

52     temp=0;
53     for(j,2,n) {
54         if (trace[j]) temp++;
55     }
56     if (temp+1 > mx) {
57         head = k;
58         for(i,1,n) trace2[i] = trace[i];
59         mx=temp+1;
60     }
61 }
62 cout << mx << endl;
63 cout << head;
64 for(i,2,n) {
65     if (trace2[i]) cout << " " << i;
66 }
67 //cout << res;
68 }
69
70
71 int main()
72 {
73     ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
74     isReadFile();
75     loli();
76     return 0;
77 }

```

C++ source file

length: 1,490 lines: 79 Ln: 47 Col:

Code 2 tham khảo:

```
D:\THANG82020\bailam\loc\DFSMVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Wind
DFSMVLT.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a[3001][3001];
5
6  int n, m, Free[3001], u, v;
7  vector <int> g, l;
8  void DFS(int u)
9  {
10     g.push_back(u);
11     Free[u] = false;
12     for (int v = 1; v <= n; v++)
13         if (a[u][v] == 1 && Free[v])
14             DFS(v);
15 }
16
17 int main()
18 {
19     freopen("DFSMVLT.INP", "r", stdin);
20     freopen("DFSMVLT.OUT", "w", stdout);
21     ios_base::sync_with_stdio(false);
22     cin.tie(NULL);
23     cout.tie(NULL);
24     cin >> n >> m;
25
26     for (int i = 1; i <= m; i++)
27     {
28         cin >> u >> v;
29         a[u][v] = 1;
30         a[v][u] = 1;
31     }
32
33     for (int i = 1; i <= n; i++)
34         Free[i] = 1;
35     for (int i = 1; i <= n; i++)
36     {
37         if (Free[i] == 1)
38         {
39             g.clear();
40             DFS(i);
41             sort(g.begin(), g.end());
42             if (g.size() > l.size())
43             {
44                 l.clear();
45                 for (auto j : g)
46                     l.push_back(j);
47             }
48         }
49         cout << l.size() << '\n';
50         for (auto i : l)
51             cout << i << " ";
52         return 0;
53     }
54 }
```

C++ source file

### Code 3 tham khảo:

```
D:\THANG82020\bailam\lan\dfsmvlt.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Wi
dfsmvlt.cpp
1 // code by Minotour <3
2 #include <bits/stdc++.h>
3
4 // #define _MULTEST_
5 #define Mino "DFSMVLT"
6
7 #define ll long long
8 #define pii pair<int, int>
9 #define pb push_back
10 #define mp make_pair
11 #define mii map<int, int>
12 #define mib map<int, bool>
13 #define vi vector<int>
14 #define qi queue<int>
15 #define fi first
16 #define se second
17 #define gl(x) getline(cin, x)
18 #define resett(x, t) memset(x, t, sizeof(x))
19
20 using namespace std;
21 const int maxa = 1e3 + 7;
22 const int maxb = 1e6 + 7;
23 const int oo = 1e9 + 7;
24 const int maxtime = 1000; //with ms
25
26 int n, m, u, v;
27 vector<vi> a;
28 vi res, tmp;
29 mii p;
30
31 void dfs(int v)
32 {
33     p[v] = 1; tmp.pb(v + 1);
34     for (int i = 0; i < a[v].size(); i++)
35         if (!p[a[v][i]]) dfs(a[v][i]);
36 }
37
38 void solve();
39 void out();
40
41 void inp()
42 {
43     // #ifndef ONLINE_JUDGE
44     freopen(Mino".INP", "r", stdin);
45     freopen(Mino".OUT", "w", stdout);
46     // #endif // ONLINE_JUDGE
47     ios_base::sync_with_stdio(0);
48     cin.tie(NULL); cout.tie(NULL);
49     #ifdef _MULTEST_
50         int T;
51         cin >> T;
52         while (T--)
53             solve();
54     #endif // _MULTEST_
55 }
56
57
```

```

58 void solve()
59 {
60     cin >> n >> m; a.resize(n);
61     while (m--)
62     {
63         cin >> u >> v;
64         a[u - 1].pb(v - 1);
65         a[v - 1].pb(u - 1);
66     }
67     res.clear();
68     for (int i = 0; i < n; i++)
69     {
70         if (!p[i])
71         {
72             tmp.clear();
73             dfs(i);
74             if (res.size() < tmp.size()) res = tmp;
75         }
76     }
77     sort(res.begin(), res.end());
78     cout << res.size() << endl;
79     for (int i = 0; i < res.size(); i++) cout << res[i] << ' ';
80 }
81
83
84 int main()
85 {
86     inp();
87     #ifndef _MULTEST_
88     solve();
89     #endif // _MULTEST_
90
91     return 0;
92 }
93

```

C++ source file

length: 1,719 | li

#### 4. Liệt kê các đỉnh đến được từ đỉnh S.. DFSLVLT.\*

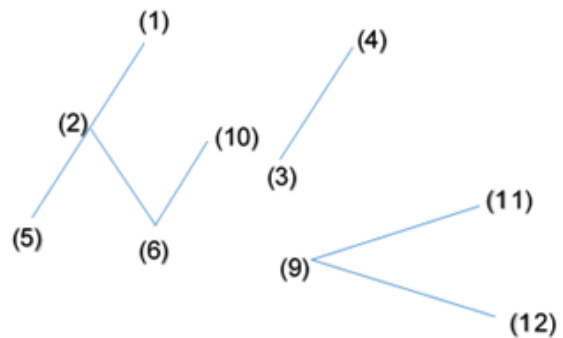
Cho đồ thị vô hướng có N đỉnh, M cạnh. Hãy liệt kê các đỉnh đi đến được từ đỉnh S.

**Dữ liệu:** Vào từ file văn bản DFSLVLT.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M và S ( $M, N \leq 3000, S \leq N$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSLVLT.OUT các đỉnh có thể đi đến được từ đỉnh S, các đỉnh được liệt kê theo thứ tự từ điển, mỗi số cách nhau một dấu cách. Nếu không có đỉnh nào tìm được thì ghi ra -1.

DFSLVLT.INP	DFSLVLT.OUT
12 7 2	1 5 6 10
1 2	
2 5	
2 6	
6 10	
3 4	
9 11	
9 12	



#### Thuật toán:

Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

Sắp xếp mảng C tăng, đưa mảng C ra. Nếu từ S không thể đi đến các đỉnh khác, ghi ra -1.

Code chương trình tham khảo:

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSLULT.PAS 1=1

Const
    fi = 'DFSLULT.inp';
    fo = 'DFSLULT.out';
    nmax = 3000;

Var
    m, n, s: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    c: array[1..nmax] of longint;
    u, v, i, j: longint;
    k: longint = 0;

Procedure Duet(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
            begin
                inc(k);
                c[k] := j;
                Duet(j);
            end;
    end;
End;

Procedure Sapxep();
var i, j, temp: longint;
Begin
    for i := 1 to k-1 do
        for j := i+1 to k do
            if c[i] > c[j] then
                begin
                    temp := c[i];
                    c[i] := c[j];
                    c[j] := temp;
                end;
        end;
    end;
End;

Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s);
    fillchar(fre, sizeof(fre), True);
    for i := 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
            a[v, u] := 1;
        end;
    duet(s);
    if k = 0 then write(-1)
    else
        begin
            Sapxep();
            for i := 1 to k do write(c[i], #32);
        end;
    close(input); close(output);
End.

3:17
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

```

Code 1 tham khảo:

```
D:\THANG82020\bailam\bac\DFSLVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSLVLT.cpp
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  bool a[1001][1001];
6  bool check[1001];
7  int n, m, s;
8
9  void dfs(int i)
10 {
11     check[i] = true;
12     for (int j = 1; j <= n; j++)
13         if (a[i][j] && !check[j])
14         {
15             check[j] = true;
16             dfs(j);
17         }
18 }
19
20 int main()
21 {
22     ios_base::sync_with_stdio(false);
23     cin.tie(NULL);
24     cout.tie(NULL);
25
26     freopen("DFSLVLT.INP", "r", stdin);
27     freopen("DFSLVLT.OUT", "w", stdout);
28
29     cin >> n >> m >> s;
30     for (int i = 1; i <= m; i++)
31     {
32         int u, v;
33
34         cin >> u >> v;
35         a[u][v] = 1;
36         a[v][u] = 1;
37     }
38     dfs(s);
39     int dem = 0;
40     for(int j = 1; j <= n; j++)
41     {
42         if(check[j] && j != s)
43         {
44             cout << j << ' ';
45             dem++;
46         }
47     }
48     if(dem == 0)
49         cout << -1;
50     return 0;
51 }
52
C++ source file length: 88
```

Code 2 tham khảo:

```
D:\THANG82020\bailam\aduc\DFSLVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSLVLT.cpp
1  #include <bits/stdc++.h>
2  #define nmax 100000
3
4  using namespace std;
5
6  vector <int> g[nmax];
7  vector <int> result;
8  vector <int> visit(nmax,0);
9  int res = 0;
10
11 void dfs(int u)
12 {
13     visit[u] = 1;
14     result.push_back(u);
15     for(int j=0;j<g[u].size();j++)
16     {
17         int v = g[u][j];
18         if(visit[v]==0){
19             dfs(v);
20         }
21     }
22 }
23
24 void solve()
25 {
26     int n,m,s;
27     cin >> n >> m >> s;
28     for(int i=1;i<=m;i++)
29     {
30         int u,v;
31         cin >> u >> v;
32         g[u].push_back(v);
33         g[v].push_back(u);
34     }
35     dfs(s);
36     sort(result.begin(),result.end());
37     for(int i=0;i<result.size();i++)
38         if(result[i]!=s) cout << result[i] << ' ';
39 }
40
41 int main()
42 {
43     ios_base::sync_with_stdio(0);
44     cin.tie(NULL);cout.tie(NULL);
45     freopen("DFSLVLT.inp","r",stdin);
46     freopen("DFSLVLT.out","w",stdout);
47     solve();
48     return 0;
49 }
50
C++ source file length: 90
```



### Code 3 tham khảo

```
*D:\THANG82020\bailam\hai\DFSLVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window

DFSLVLT.cpp
1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSLVLT"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m,s;
23 Vi g[nmax];
24 Vi gg;
25 Vi LL;
26 bool check[nmax];
27
28 int read(){
29     int res = 0;
30     char c = getchar();
31     bool Neg = false;
32     while(c == ' ' || c == '\n') c = getchar();
33     if(c == '-'){
34         Neg = true;
35         c = getchar();
36     }
37     while('0' <= c && c <= '9'){
38         res = res * 10 + c - '0';
39         c = getchar();
40     }
41     if(Neg) return -res;
42     return res;
43 }
44
45 void dfs(int u, int par){
46     if(u != par) gg.pb(u);
47     check[u] = 1;
48     for(auto xx : g[u]){
49         if(xx != par && check[xx] == 0) dfs(xx,u);
50     }
51 }
52
```

```

54 void kurumi() {
55     cin >> n >> m >> s;
56
57     for(int i = 1; i <= m; i++){
58         int u,v;
59         cin >> u >> v;
60         g[u].pb(v);
61         g[v].pb(u);
62     }
63     dfs(s,s);
64     if(gg.size() <= 1) cout << -1;
65     else{
66         sort(gg.begin(), gg.end());
67         for(auto xx : gg){
68             cout << xx << ' ';
69         }
70     }
71 }
72
73 int main()
74 {
75     ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
76     freopen(Kurumi".inp", "r", stdin);
77     freopen(Kurumi".out", "w", stdout);
78     kurumi();
79     return 0;
80 }
81

```

C++ source file length : 1,707 lines : 8

## Code 4 tham khảo

```

1 #include <bits/stdc++.h>
2 #define task "dfslvlt"
3 using namespace std;
4 vector <int> ke[3009];
5 bool check[3009];
6 int u,v,m,n,dem,s;
7 vector <int> ans;
8 void dfs(int u)
9 {
10     ans.push_back(u);
11     check[u]=1;
12     for(int t: ke[u])
13     {
14         if (check[t]==0) dfs(t);
15     }
16 }
17 int main()
18 {
19     freopen(task".inp", "r", stdin);
20     freopen(task".out", "w", stdout);
21     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
22     cin >> n >> m >> s;
23     for(int i=1; i<=m; i++)
24     {
25         cin >> u >> v;
26         ke[u].push_back(v);
27         ke[v].push_back(u);
28     }
29     if (ke[s].size()==0) cout << "-1";
30     else
31     {
32         dfs(s);

```

```
33         sort(ans.begin(),ans.end());
34         for(int i: ans) if (i!=s) cout <<i<<' ';
35     }
36
37     return 0;
38 }
39
```

C++ source file

length: 770 li

## 5. Tìm đường đi từ đỉnh S đến đỉnh T.. DFSSTVLT.\*

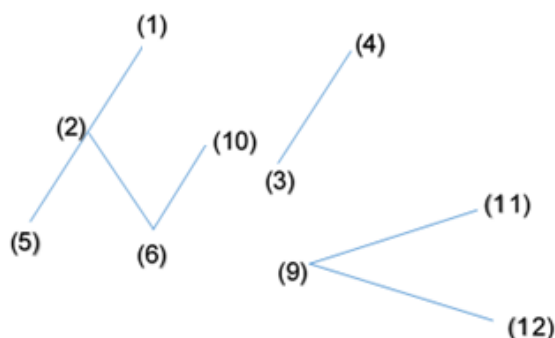
Cho đồ thị vô hướng có N đỉnh, M cạnh, không có chu trình. Hãy tìm đường đi từ đỉnh S đến đỉnh T.

**Dữ liệu:** Vào từ file văn bản DFSSTVLT.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M, S và T ( $M, N \leq 3000$ ,  $S, T \leq N$ ,  $S \neq T$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSSTVLT.OUT đường đi từ S tới T. Mỗi số cách nhau một dấu cách. Nếu không có đường đi thì ghi ra -1.

DFSSTVLT.INP	DFSSTVLT.OUT
12 7 2 10	10 6 2
1 2	(truy vết ngược)
2 5	
2 6	
6 10	
3 4	
9 11	
9 12	



### Thuật toán:

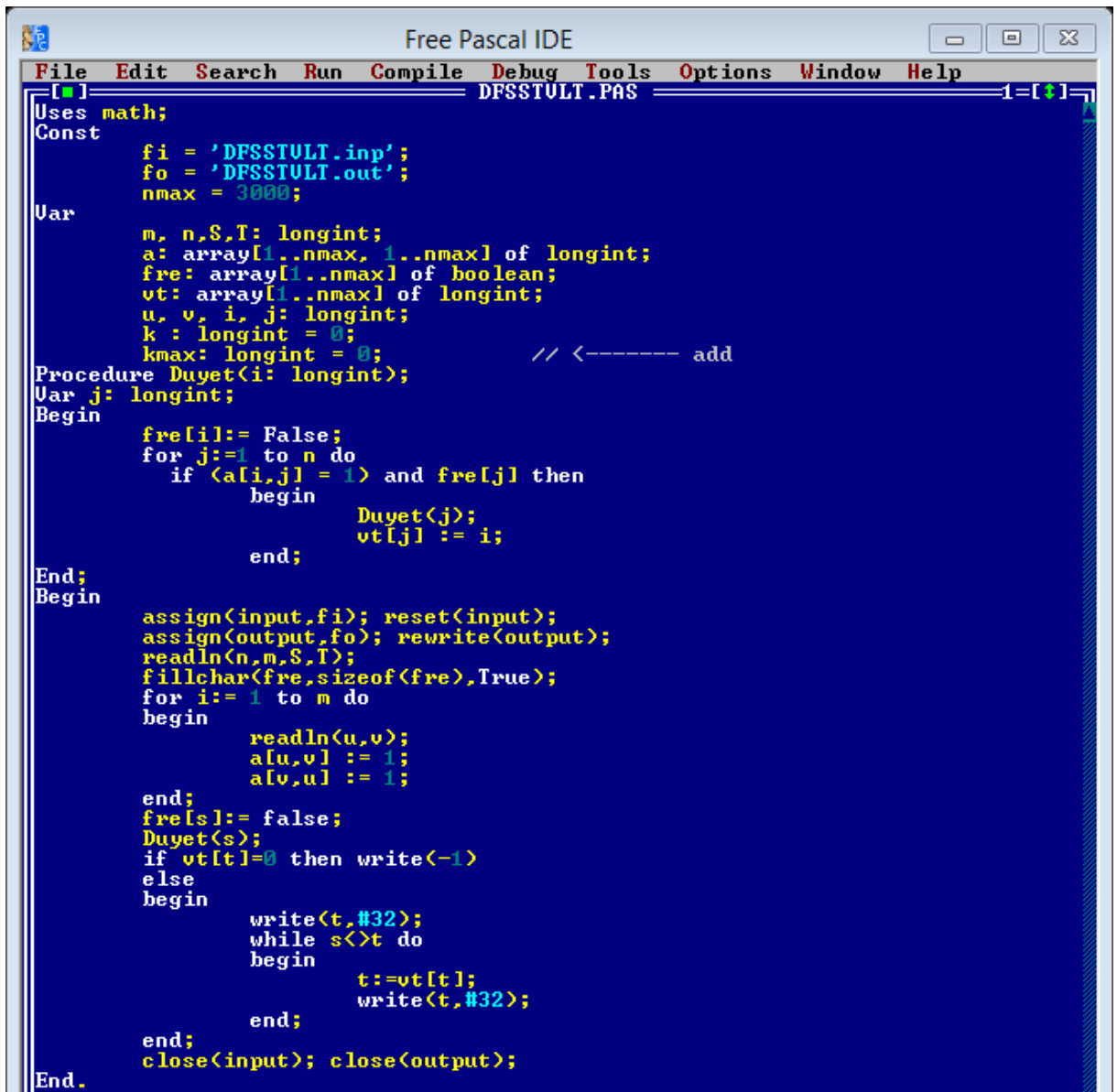
Ta đánh dấu đỉnh S đã thăm. Thực hiện Duyệt (S);

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta đánh dấu  $vt[j] = i$ ; thể hiện để đến j, ta đi qua đỉnh i.

Thực hiện, truy vết ngược để đưa ra đường đi từ S tới T (nếu có). Ngược lại, nếu không có đường đi tới T, chúng ta không tồn tại đường đi từ S tới T, ta ghi ra -1.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSSTULT.PAS
1=[ ]

Uses math;
Const
    fi = 'DFSSTULT.inp';
    fo = 'DFSSTULT.out';
    nmax = 3000;

Var
    m, n, S, I: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j: longint;
    k: longint = 0;
    kmax: longint = 0;           // <----- add

Procedure Duyet(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
            begin
                Duyet(j);
                vt[j] := i;
            end;
    end;
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, S, I);
    fillchar.fre, sizeof.fre, True);
    for i := 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
            a[v, u] := 1;
        end;
    fre[S] := false;
    Duyet(S);
    if vt[I] = 0 then write(-1)
    else
        begin
            write(t, #32);
            while s <> t do
                begin
                    t := vt[t];
                    write(t, #32);
                end;
        end;
    close(input); close(output);
End.
```

Code 1 tham khảo:

```
D:\THANG82020\baillam\duong\dfsstvt.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
dfsstvt.cpp
1  #include <bits/stdc++.h>
2  #define task "dfsstvt"
3  using namespace std;
4  vector <int> ke[3009];
5  bool check[3009];
6  int nb[3009];
7  bool kt;
8  int u,v,m,n,dem,s,t;
9  vector <int> ans;
10 void dfs(int u)
11 {
12     check[u]=1;
13     if (u==t) kt=1;
14     for(int t: ke[u])
15         if (check[t]==0)
16         {
17             nb[t]=u;
18             dfs(t);
19         }
20 }
21 int main()
22 {
23     freopen(task".inp","r",stdin);
24     freopen(task".out","w",stdout);
25     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
26     cin >>n>>m>>s>>t;
27
28     for(int i=1;i<=m;i++)
29     {
30         cin >>u>>v;
31         ke[u].push_back(v);
32         ke[v].push_back(u);
33     }
34     dfs(s);
35     if (kt==0) cout <<-1;
36     else
37     {
38         int j=t;
39         while (nb[j]!=s)
40         {
41             ans.push_back(j);
42             j=nb[j];
43         }
44         ans.push_back(s);
45         for(int i: ans) cout <<i<<' ';
46     }
47     return 0;
48 }
49
C++ source file length: 929 lines: 49 Ln: 20 Col: 2 Se
```

## Code 2 tham khảo:

```
D:\THANG82020\bailam\loc\DFSSTVLT.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSSTVLT.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a[3001][3001];
5
6  int n, m, Free[3001], f[3001], u, v, s, t, ma=INT_MIN, check=0;
7  void DFS(int u,int cnt)
8  {
9      f[cnt]=u;
10     if (u==t)
11     {
12         for (int i=cnt;i>=1;i--)
13             cout << f[i] << " ";
14         check=1;
15         return;
16     }
17     Free[u] = false;
18     for (int v = 1; v <= n; v++)
19         if (a[u][v] == 1 && Free[v])
20             DFS(v,cnt+1);
21 }
22
23 int main()
24 {
25     freopen("DFSSTVLT.INP","r",stdin);
26     freopen("DFSSTVLT.OUT","w",stdout);
27
28     ios_base::sync_with_stdio(false);
29     cin.tie(NULL);
30     cout.tie(NULL);
31     cin >> n >> m >> s >> t;
32     for (int i = 1; i <= n; i++)
33         for (int j = 1; j <= n; j++)
34             a[i][j] = 0;
35     for (int i = 1; i <= m; i++)
36     {
37         cin >> u >> v;
38         a[u][v] = 1;
39         a[v][u] = 1;
40     }
41     for (int i = 1; i <= n; i++)
42         Free[i] = 1;
43     DFS(s,1);
44     if (check==0)
45     {
46         cout << -1;
47         return 0;
48     }
49     return 0;
50 }
```

C++ source file      length : 1,022   lines : 51      Ln : 39   Col : 6

## 6. Xây cầu.. DFSBRIGE.\*

Tại quần đảo ZXY có N hòn đảo, một số hòn đảo đã có cầu nối với nhau. Từ đảo này ta có thể đi sang đảo khác bằng đường đi trực tiếp hoặc đi gián tiếp qua các đảo khác.

Để thuận tiện cho các phương tiện đi lại, ban quản lý sẽ xây thêm một số cầu để từ một đảo ta có thể đi đến các đảo còn lại trong quần đảo.

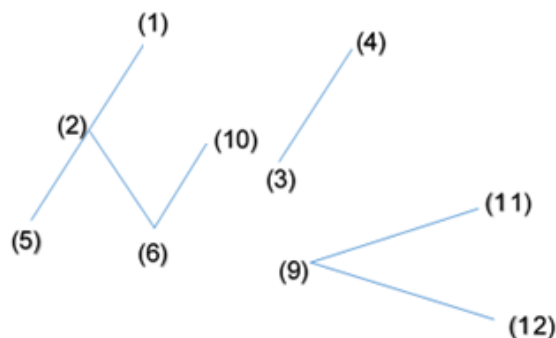
Hãy cho biết, ban quản lý cần xây ít nhất bao nhiêu cầu

**Dữ liệu:** Vào từ file văn bản DFSBRIGE.INP gồm:

- Dòng 1: Ghi số nguyên N và M ( $M, N \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSBRIGE.OUT số cầu ít nhất cần xây thêm.

DFSBRIGE.INP	DFSBRIGE.OUT
12 7 1 2 2 5 2 6 6 10 3 4 9 11 9 12	4



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ u tới v.

$a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp từ u tới v.

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ i, đánh dấu i đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh j đang free và có đường đi trực tiếp từ i đến j thì ta Duyệt ( j ).

Tới đỉnh j, ta lặp lại việc duyệt như ở bước trên.



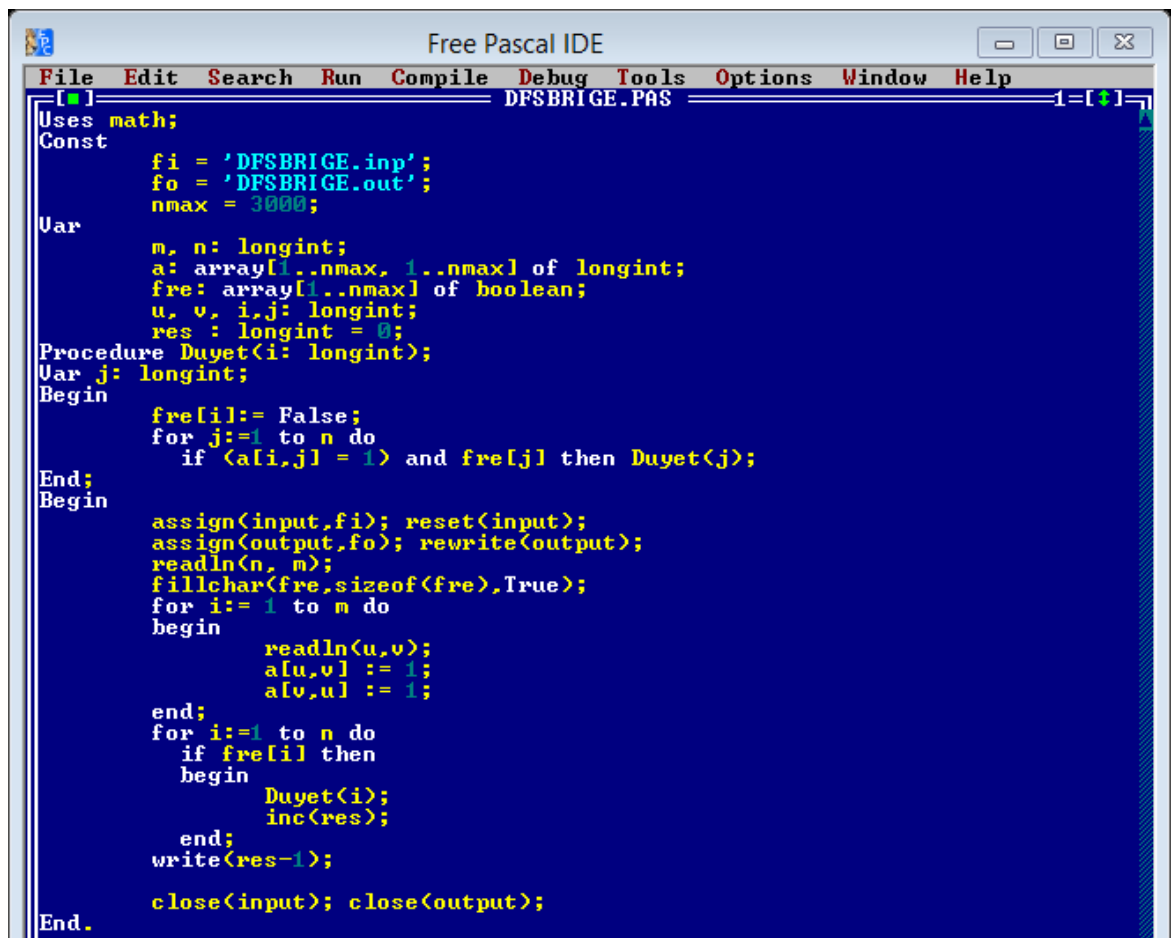
**Lưu ý:** Với mỗi đỉnh  $i$ , ta sẽ đi đến được tất cả các đỉnh có đường đi trực tiếp hoặc gián tiếp tới  $i$ . Như vậy, mỗi lần duyệt ( $i$ ) xong ta sẽ thu được một vùng liên thông. Ở bài này, ta tăng biến đếm số lượng vùng liên thông lên.

Số câu cần xây thêm là: Số vùng liên thông  $- 1$ .

Độ phức tạp thuật toán:  $O(M + N)$

Cải tiến: Dùng cấu trúc dữ liệu đặc biệt để làm bài này.

Code chương trình tham khảo:



```
Free Pascal IDE
DFSBRIGE.PAS
1=[ ]

Uses math;
Const
    fi = 'DFSBRIGE.inp';
    fo = 'DFSBRIGE.out';
    nmax = 3000;
Var
    m, n: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    u, v, i, j: longint;
    res: longint = 0;
Procedure Duyệt(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j:=1 to n do
        if (a[i,j] = 1) and fre[j] then Duyệt(j);
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m);
    fillchar.fre, sizeof(fre), True;
    for i:= 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
            a[v, u] := 1;
        end;
    for i:=1 to n do
        if fre[i] then
            begin
                Duyệt(i);
                inc(res);
            end;
    write(res-1);
    close(input); close(output);
End.
```

Code 1 tham khảo:

```
*D:\THANG82020\bailam\truong\DFSBRIGE.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSBRIGE.cpp
1  #include <bits/stdc++.h>
2  #define nmax 3001
3  using namespace std;
4
5  int a[3001][3001];
6  int n, m, u, v, s, dem=0, res=0;
7  bool fre[nmax];
8  void DFS(int u)
9  {
10     fre[u] = true;
11     for (int v = 1; v <= n; v++)
12         if (a[u][v] == 1 && !fre[v])
13             DFS(v);
14 }
15 int main()
16 {
17     freopen("DFSBRIGE.INP", "r", stdin);
18     freopen("DFSBRIGE.OUT", "w", stdout);
19     cin >> n >> m;
20     for (int i = 1; i <= m; i++)
21     {
22         cin >> u >> v;
23         a[u][v] = 1;
24         a[v][u] = 1;
25     }
26     for(int i=1;i<=n;i++)
27     {
28         if(!fre[i])
29         {DFS(i);
30          res++;}
31     }
32     cout<<res-1;
33     return 0;
34 }
35
C++ source file length : 657
```

## Code 2 tham khảo:

```
"D:\THANG82020\bailam\duong\DFSBRIGE.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSBRIGE.cpp
1 #include <bits/stdc++.h>
2 #define task "DFSBRIGE"
3 using namespace std;
4 vector <int> ke[3009];
5 bool check[3009];
6 int u,v,m,n,dem;
7 void dfs(int u)
8 {
9     check[u]=1;
10    for(int t: ke[u])
11    {
12        if (check[t]==0) dfs(t);
13    }
14 }
15
16 int main()
17 {
18     freopen(task".inp","r",stdin);
19     freopen(task".out","w",stdout);
20     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
21     cin >>n>>m;
22     for(int i=1;i<=m;i++)
23     {
24         cin >>u>>v;
25         ke[u].push_back(v);
26         ke[v].push_back(u);
27     }
28     for(int i=1;i<=n;i++)
29     {
30         if (check[i]==0)
31         {
32             dem++;
33             dfs(i);
34         }
35         cout <<dem-1;
36         return 0;
37 }
```

C++ source file length: 675 lines: 37 Ln: 15 Col: 1 Sel: 0 |

## 7. Đường đi từ S.. DFSTRACE.\*

Tại quần đảo ZXY có N hòn đảo, một số hòn đảo đã có cầu nối với nhau, không có chu trình. Từ đảo này ta có thể đi sang đảo khác bằng đường đi trực tiếp hoặc đi gián tiếp qua các đảo khác.

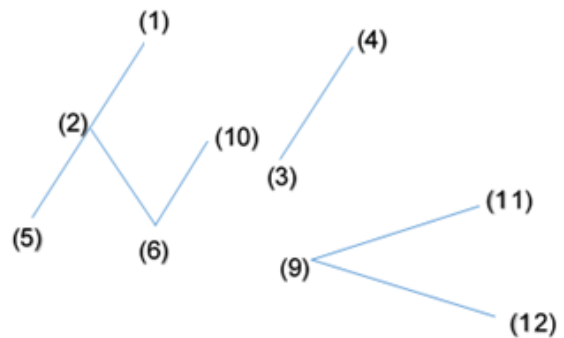
Jame vừa đến hòn đảo S, anh muốn đi bằng đường bộ đến các đảo khác. Hãy cho biết các lộ trình mà Jame có thể đi.

**Dữ liệu:** Vào từ file văn bản DFSTRACE.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M và S ( $M, N \leq 3000, S \leq N$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSTRACE.OUT các lộ trình mà Jame có thể đi bằng đường bộ từ đảo S. Mỗi số ghi cách nhau một dấu cách. Nếu Jame không có đường đi sang đảo khác thì ghi ra -1.

DFSTRACE.INP	DFSTRACE.OUT
12 7 2	1 2
1 2	5 2
2 5	6 2
2 6	10 6 2
6 10	(Truy vết ngược)
3 4	
9 11	
9 12	



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ u tới v.

$a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp từ u tới v.

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ i, đánh dấu i đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh j đang free và có đường đi trực tiếp từ i đến j thì ta:

```

{
    Lưu đường đi tới j là i vào mảng vt[j] := i;
    Truy vết, đưa hành trình từ đỉnh j về đỉnh i.
    Duyệt ( j ).
}

```

Tới đỉnh j, ta lặp lại việc duyệt như ở bước trên.

Khi không có đường đi từ S tới đảo khác ta ghi ra -1.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSTRACE.pas
Const
    fi = 'DFSTRACE.inp';
    fo = 'DFSTRACE.out';
    nmax = 3000;
Var
    m, n, s: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j: longint;
Procedure trace(t: longint);
Begin
    while t <> s do
    begin
        write(t, #32);
        t := vt[t];
    end;
    writeln(s);
    vt[s] := 3000;
End;
Procedure Duyet(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
        begin
            vt[j] := i;
            trace(j);
            Duyet(j);
        end;
    end;
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s);
    fillchar.fre, sizeof(fre), True;
    for i := 1 to m do
    begin
        readln(u, v);
        a[u, v] := 1;
        a[v, u] := 1;
    end;
    duyet(s);
    if vt[s] = 0 then write(-1);
    close(input); close(output);
End.

```

## Code 1 tham khảo:

```
"D:\THANG82020\bailam\duong\DFSTRACE.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSTRACE.cpp
1  #include <bits/stdc++.h>
2  #define task "DFSTRACE"
3  using namespace std;
4  vector <int> ke[3009];
5  bool check[3009];
6  int nb[3009];
7  int u,v,m,n,s;
8  vector <int> ans;
9  void dfs(int u)
10 {
11     check[u]=1;
12     for(int t: ke[u])
13     if (check[t]==0)
14     {
15         ans.push_back(t);
16         nb[t]=u;
17         dfs(t);
18     }
19 }
20
21 int main()
22 {
23     freopen(task".inp","r",stdin);
24     freopen(task".out","w",stdout);
25     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
26     cin >>n>>m>>s;
27     for(int i=1;i<=m;i++)
28     {
29         cin >>u>>v;
30         ke[u].push_back(v);
31         ke[v].push_back(u);
32     }
33     dfs(s);
34
35     // sort (ans.begin(),ans.end());
36     if (ke[s].size()==0) cout <<-1;
37     else
38     for(int t: ans)
39     {
40         vector<int> in;
41
42         int j=t;
43         while (nb[j]!=s)
44         {
45             in.push_back(j);
46             j=nb[j];
47         }
48         in.push_back(j);
49         in.push_back(s);
50         for(int i: in) cout <<i<<' ';
51         cout <<'\\n';
52     }
53     return 0;
54 }
```

C++ source file length: 1,030 lines: 53 Ln: 34 Col: 5 Sel:

```
39     int j=t;
40     while (nb[j]!=s)
41     {
42         in.push_back(j);
43         j=nb[j];
44     }
45     in.push_back(j);
46     in.push_back(s);
47     for(int i: in) cout <<i<<' ';
48     cout <<'\\n';
49 }
50 return 0;
51 }
```

C++ source file length: 1,025 lines: 51

Code 2 tham khảo:

```
*D:\THANG82020\baillam\hduc\DFSTRACE.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSTRACE.cpp
1  #include<bits/stdc++.h>
2  using namespace std;
3  int n,m,dem=0,u,v,temp=1,k,s,t,ck=0;
4  bool a[3005][3005];
5  int cx[3005];
6  int b[10000];
7
8  void DFS(int i,int s)
9  {
10     cx[s]=1;
11     for (int j=1; j<=n; j++)
12         if (cx[j]==0 && a[s][j] && s!=j)
13         {
14             b[i]=j;
15             ck=1;
16             for (int p=i;p>=0;p--) cout<<b[p]<<" ";cout<<endl;
17             DFS(i+1,j);
18         }
19
20
21
22
21 int main()
22 {
23     freopen("DFSTRACE.inp","r",stdin);
24     freopen("DFSTRACE.out","w",stdout);
25     cin >> n >> m >> s;
26     for (int i=1; i<=m; i++)
27     {
28         cin >> u >> v;
29         a[u][v]=1;
30         a[v][u]=1;
31     }
32     b[0]=s;
33     DFS(1,s);
34     if (ck==0) cout<<-1;
35     return 0;
36 }
37
38
39
C++ source file length : 687 lines : 39
```

## 8. Đường đi từ S qua nhiều đảo nhất.. DFSLMAX.\*

Tại quần đảo ZXY có N hòn đảo, một số hòn đảo đã có cầu nối với nhau, không có chu trình. Từ đảo này ta có thể đi sang đảo khác bằng đường đi trực tiếp hoặc đi gián tiếp qua các đảo khác.

Jame vừa đến hòn đảo S, anh muốn đi bằng đường bộ đến các đảo khác. Khi đến một đảo, anh ấy có thể nghỉ ngơi ở đảo ấy hoặc đi tiếp sang các đảo khác. Tất nhiên, anh ấy chỉ muốn tham quan *mỗi đảo một lần*.

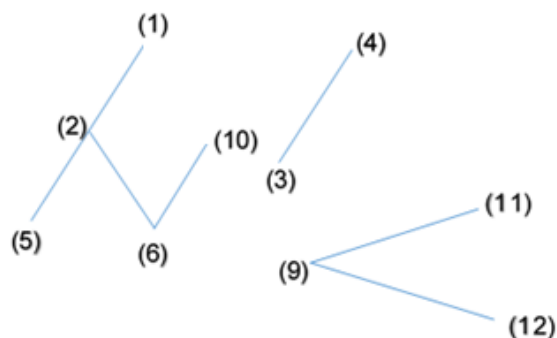
Hãy tìm số đảo nhiều nhất mà Jame có thể tham quan bằng đường bộ.

**Dữ liệu:** Vào từ file văn bản DFSLMAX.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M và S ( $M, N \leq 3000, S \leq N$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSLMAX.OUT số đảo nhiều nhất mà Jame có thể tham quan bằng đường bộ.

DFSLMAX.INP	DFSLMAX.OUT
12 7 2	3
1 2	(gồm các đảo 10
2 5	6 2)
2 6	
6 10	
3 4	
9 11	
9 12	



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ u tới v.

$a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp từ u tới v.

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ i, đánh dấu i đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh j đang free và có đường đi trực tiếp từ i đến j thì ta:

{

Lưu đường đi tới j là i vào mảng  $vt[j] := i$ ;



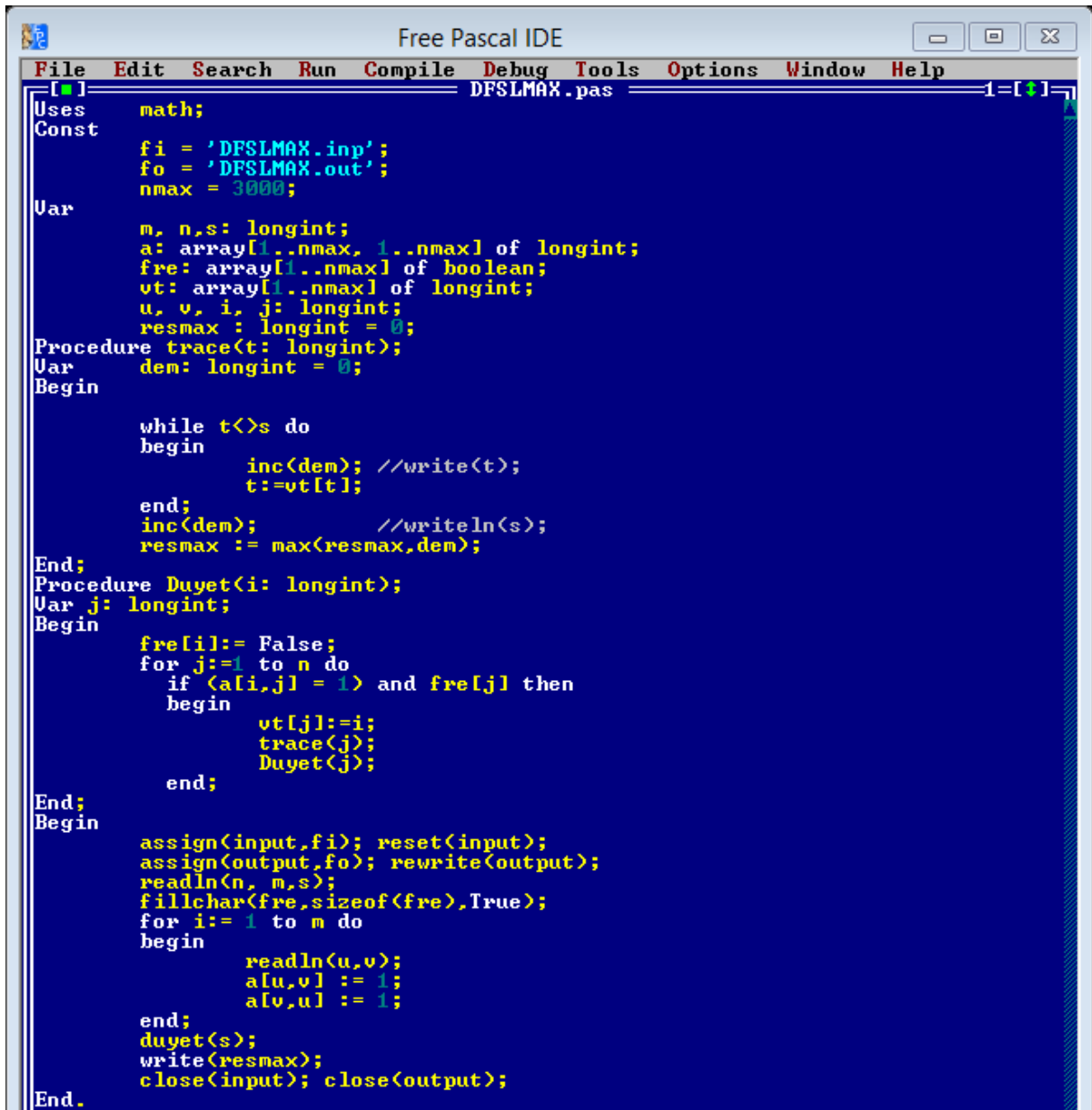
Truy vết, đưa hành trình từ đỉnh  $j$  về đỉnh  $i$  và cập nhật số lượng đỉnh nhiều nhất đi qua vào  $resmax$ .

Duyệt ( $j$ ).

}

Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Độ phức tạp thuật toán:  $O(M + N)$



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
[ ] DFSLMAX.pas 1=1

Uses
Const
    math;
    fi = 'DFSLMAX.inp';
    fo = 'DFSLMAX.out';
    nmax = 30000;

Var
    m, n, s: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j: longint;
    resmax: longint = 0;

Procedure trace(t: longint);
Var
    dem: longint = 0;
Begin
    while t <> s do
    begin
        inc(dem); //write(t);
        t := vt[t];
    end;
    inc(dem); //writeln(s);
    resmax := max(resmax, dem);
End;

Procedure Duyệt(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
        begin
            vt[j] := i;
            trace(j);
            Duyệt(j);
        end;
    end;
End;

Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s);
    fillchar.fre, sizeof.fre, True;
    for i := 1 to m do
    begin
        readln(u, v);
        a[u, v] := 1;
        a[v, u] := 1;
    end;
    duyet(s);
    write(resmax);
    close(input); close(output);
End.
```

Code 1 tham khảo:

```
*D:\THANG82020\baillam\thao\DFSLMAX.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSLMAX.cpp
1  #include <bits/stdc++.h>
2  #define ft first
3  #define sc second
4  using namespace std;
5  int n,m,k;
6  int v[3009];
7  set<int>b[3009];
8  set<int>::iterator p;
9  int dem=1;
10 int res=0;
11 vector<int>a[3009];
12 void dfs(int i)
13 {
14     res=max(res,dem);
15     v[i]=1;
16     for(int j=0;j<a[i].size();j++)
17     {
18         if(v[a[i][j]]==0)
19         {
20             dem++;
21             dfs(a[i][j]);
22             dem--;
23         }
24     }
25
26     for(int k=1;k<=n;k++)
27         for(int j=0;j<a[k].size();j++)
28         {
29             if(a[k][j]==i&&v[k]==0)
30             {
31                 dem++;
32                 dfs(k);
33                 dem--;
34             }
35         }
36 }
37
38 int main()
39 {
40     freopen("DFSLMAX.inp","r",stdin);
41     freopen("DFSLMAX.out","w",stdout);
42     cin>>n>>m>>k;
43     while(m--)
44     {
45         int u,v;
46         cin>>u>>v;
47         a[u].push_back(v);
48     }
49     dfs(k);
50     cout<<res;
51     return 0;
52 }
53
C++ source file
```

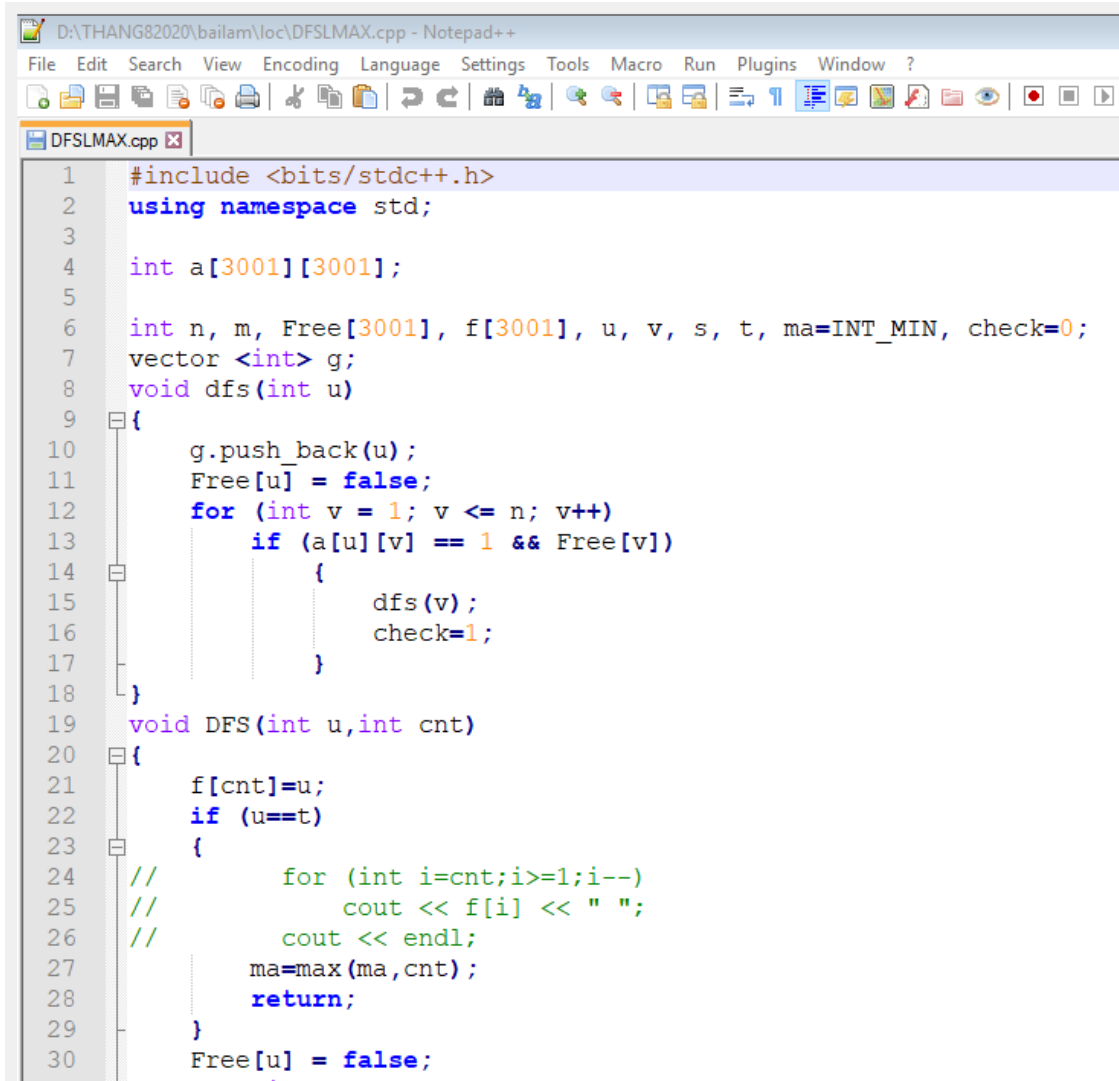
Code 2 tham khảo:

```
D:\THANG82020\bailam\quang\DFSLMAX.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSLMAX.cpp x
1
2 #include <bits/stdc++.h>
3 #define ll long long
4 #define fd(x,a,b) for(int x=a;x>=b;x--)
5 #define fi(x,a,b) for(int x=a;x<=b;x++)
6
7 #define task "DFSLMAX"
8
9 const ll oo=1e18;
10 const ll mod=1e9;
11 const ll nmax=3007;
12
13 int n,t,u,v,res=0,s,mx=0;
14 using namespace std;
15 vector<int> g[nmax],trace;
16 bool fre[nmax],kt;
17
18
19 void isReadFile()
20 {
21     freopen(task".INP","r",stdin);
22     freopen(task".OUT","w",stdout);
23 }
24
25 void dfs(int u)
26 {
27     fre[u] = 0;
28     trace.insert(trace.begin(), u);
29     if (trace.size() > 1)
30     {
31         //         for (auto i: trace) cout << i << " ";
32         //         cout << endl;
33         mx=max(mx, (int)trace.size());
34     }
35     if (g[u].size() != 0) {
36         fi(v,0,g[u].size()-1) {
37             if (fre[g[u][v]]) {
38                 kt=1;
39                 dfs(g[u][v]);
40             }
41         }
42     }
43     trace.erase(trace.begin());
44 }
45
46 void loli()
47 {
48     memset(fre,1,sizeof(fre));
49     cin >> n >> t >> s;
50     fi(i,1,t) {
51         cin >> u >> v;
52         g[u].push_back(v);
53         g[v].push_back(u);
54     }
55 }
```

```
62
63 int main()
64 {
65     ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
66     isReadFile();
67     loli();
68     return 0;
69 }
70
71
```

C++ source file length: 1,291 lines: 71

Code 3 tham khảo:



```
D:\THANG82020\bailam\loc\DFSLMAX.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSLMAX.cpp
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 int a[3001][3001];
5
6 int n, m, Free[3001], f[3001], u, v, s, t, ma=INT_MIN, check=0;
7 vector<int> g;
8 void dfs(int u)
9 {
10     g.push_back(u);
11     Free[u] = false;
12     for (int v = 1; v <= n; v++)
13         if (a[u][v] == 1 && Free[v])
14             {
15                 dfs(v);
16                 check=1;
17             }
18 }
19 void DFS(int u,int cnt)
20 {
21     f[cnt]=u;
22     if (u==t)
23     {
24         //         for (int i=cnt;i>=1;i--)
25         //             cout << f[i] << " ";
26         //         cout << endl;
27         ma=max(ma,cnt);
28         return;
29     }
30     Free[u] = false;
```

```

31     for (int v = 1; v <= n; v++)
32         if (a[u][v] == 1 && Free[v])
33             DFS(v, cnt+1);
34     }
35
36     int main()
37     {
38         freopen("DFSLMAX.INP", "r", stdin);
39         freopen("DFSLMAX.OUT", "w", stdout);
40         ios_base::sync_with_stdio(false);
41         cin.tie(NULL);
42         cout.tie(NULL);
43         cin >> n >> m >> s;
44         for (int i = 1; i <= n; i++)
45             for (int j = 1; j <= n; j++)
46                 a[i][j] = 0;
47         for (int i = 1; i <= m; i++)
48         {
49             cin >> u >> v;
50             a[u][v] = 1;
51             a[v][u] = 1;
52         }
53         for (int i = 1; i <= n; i++)
54             Free[i] = 1;
55         dfs(s);
56
57         for (int i=1; i<g.size(); i++)
58         {
59             for (int i=1; i<=n; i++)
60                 Free[i]=1;
61
62         }
63         if (check==0)
64         {
65             cout << 1;
66             return 0;
67         }
68         else
69             cout << ma;
70         return 0;
71     }
72 }
73
74

```

C++ source file

length : 1,457 line

## 9. Tham quan K đảo.. DFSLK.\*

Tại quần đảo ZXY có N hòn đảo, một số hòn đảo đã có cầu nối với nhau, không có chu trình. Từ đảo này ta có thể đi sang đảo khác bằng đường đi trực tiếp hoặc đi gián tiếp qua các đảo khác.

Jame vừa đến hòn đảo S, anh muốn đi bằng đường bộ đến các đảo khác. Khi đến một đảo, anh ấy có thể nghỉ ngơi ở đảo ấy hoặc đi tiếp sang các đảo khác. Tất nhiên, anh ấy chỉ muốn tham quan mỗi đảo một lần. Do thời gian đi du lịch không nhiều, nên anh ấy muốn tham qua K đảo trong quần đảo ZXY.

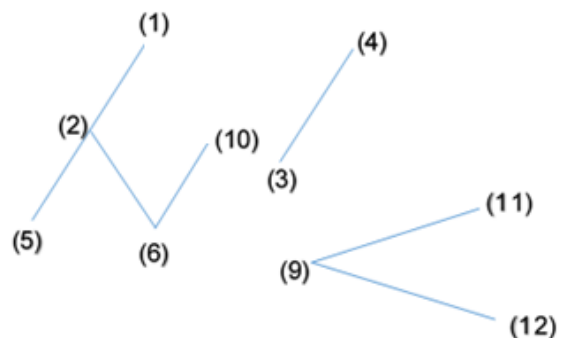
Hãy tìm các hành trình mà Jame có thể tham quan bằng đường bộ theo yêu cầu trên.

**Dữ liệu:** Vào từ file văn bản DFSLK.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M, S và K ( $M, N \leq 3000, K, S \leq N$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSLK.OUT các hành trình tham quan đi qua K hòn đảo. Mỗi số cách nhau một dấu cách. Nếu không có phương án tham quan, ghi ra -1.

DFSLK.INP	DFSLK.OUT
12 7 1 3	5 2 1
1 2	6 2 1
2 5	(truy vết ngược)
2 6	
6 10	
3 4	
9 11	
9 12	



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ u tới v.

$a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp từ u tới v.

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

{

Lưu đường đi tới  $j$  là  $i$  vào mảng  $vt[j] := i$ ;

Truy vết, đưa hành trình từ đỉnh  $j$  về đỉnh  $i$  và đếm số lượng đỉnh đi qua vào đếm.

Nếu đếm  $> k$  thì ta bỏ qua đường đi này.

Nếu đếm  $= k$  thì ta truy vết đưa hành trình ra.

Duyệt ( $j$ ).

}

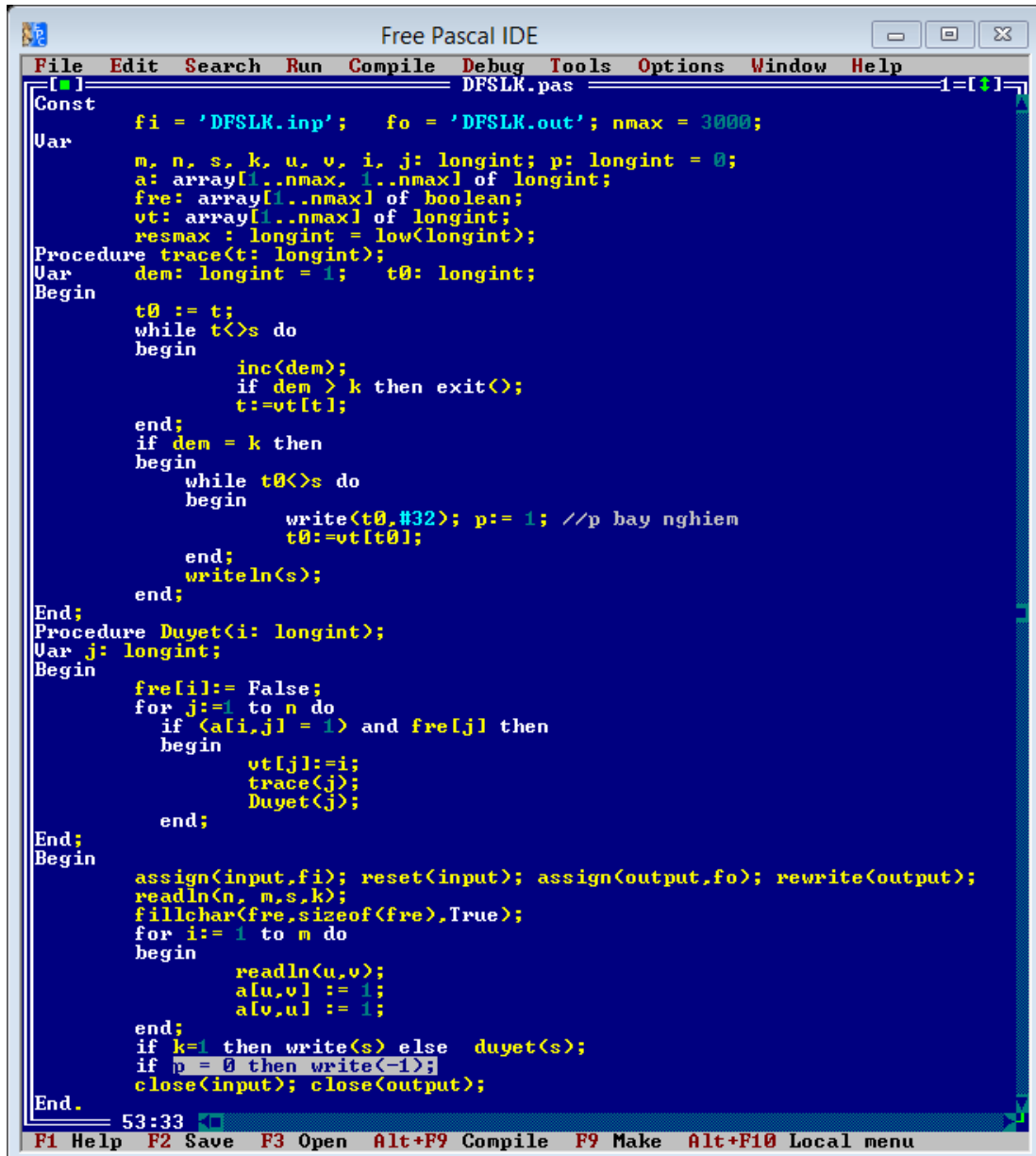
Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Lưu ý: Khi  $K = 1$  tức là Jame chỉ tham quan đảo  $S$  đang đứng, ta đưa  $S$  ra.

Khi Jame đứng ở một đảo mà không thể đi đến  $K$  đảo, ta đưa ra -1.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSLK.pas 1=1

Const
  fi = 'DFSLK.inp';  fo = 'DFSLK.out'; nmax = 3000;
Var
  m, n, s, k, u, v, i, j: longint; p: longint = 0;
  a: array[1..nmax, 1..nmax] of longint;
  fre: array[1..nmax] of boolean;
  vt: array[1..nmax] of longint;
  resmax: longint = low(longint);
Procedure trace(t: longint);
Var
  dem: longint = 1;  t0: longint;
Begin
  t0 := t;
  while t <> s do
  begin
    inc(dem);
    if dem > k then exit();
    t := vt[t];
  end;
  if dem = k then
  begin
    while t0 <> s do
    begin
      write(t0, #32); p := 1; //p hay nghiem
      t0 := vt[t0];
    end;
    writeln(s);
  end;
End;
Procedure Duyệt(i: longint);
Var j: longint;
Begin
  fre[i] := False;
  for j := 1 to n do
  if (a[i, j] = 1) and fre[j] then
  begin
    vt[j] := i;
    trace(j);
    Duyệt(j);
  end;
End;
Begin
  assign(input, fi); reset(input); assign(output, fo); rewrite(output);
  readln(n, m, s, k);
  fillchar(fre, sizeof(fre), True);
  for i := 1 to m do
  begin
    readln(u, v);
    a[u, v] := 1;
    a[v, u] := 1;
  end;
  if k = 1 then write(s) else  duyet(s);
  if p = 0 then write(-1);
  close(input); close(output);
End.
53:33
F1 Help  F2 Save  F3 Open  Alt+F9 Compile  F9 Make  Alt+F10 Local menu
```



Code 1 tham khảo:

```
D:\THANG82020\baillam\loc\DFS_LK.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFS_LK.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a[3001][3001];
5
6  int n, m, Free[3001], f[3001], u, v, s, t, ma=INT_MIN, check=0, b[3001];
7  void DFS(int u,int cnt)
8  {
9      f[cnt]=u;
10     if (cnt==t)
11     {
12         for (int i=cnt;i>=1;i--)
13             cout << f[i] << " ";
14         cout << endl;
15         check=1;
16         return;
17     }
18     Free[u] = false;
19     for (int v = 1; v <= n; v++)
20         if (a[u][v] == 1 && Free[v])
21             DFS(v,cnt+1);
22 }
23
24 int main()
25 {
26     freopen("DFS_LK.INP","r",stdin);
27     freopen("DFS_LK.OUT","w",stdout);
28     ios_base::sync_with_stdio(false);
29     cin.tie(NULL);
30     cout.tie(NULL);
31     cin >> n >> m >> s >> t;
32     for (int i = 1; i <= n; i++)
33         for (int j = 1; j <= n; j++)
34             a[i][j] = 0;
35     for (int i = 1; i <= m; i++)
36     {
37         cin >> u >> v;
38         a[u][v] = 1;
39         a[v][u] = 1;
40     }
41
42     for (int i = 1; i <= n; i++)
43         Free[i] = 1;
44     DFS(s,1);
45     if (check==0)
46     {
47         cout << -1;
48         return 0;
49     }
50     return 0;
51 }
52
53
C++ source file | length : 1,052 | lines : 53
```

Code 2 tham khảo:

```
*D:\THANG82020\bailam\hai\DFSLK.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window

DFSLK.cpp
1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSLK"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m,s,t,kt = 0, k;
23 Vi g[nmax];
24 Vi gg;
25 int a[nmax];
26 bool check[nmax];
27
28
29 void dfs(int u,int par,int cnt, int k){
30     a[cnt] = u;
31     if(check[u] == 0 && cnt == k){
32         for(int i = cnt; i >= 1; i--){
33             cout << a[i] << ' ';
34         }
35         cout << '\n';
36         kt = 1;
37     }
38     gg.pb(u);
39     check[u] = 1;
40     for(auto xx : g[u]){
41         if(xx != par && check[xx] == 0) dfs(xx,u,cnt + 1,k);
42     }
43 }
44
45 void kurumi(){
46     cin >> n >> m >> s >> k;
47
48     for(int i = 1; i <= m; i++){
49         int u,v;
50         cin >> u >> v;
51         g[u].pb(v);
52         g[v].pb(u);
53     }
54     for(int i = 1; i <= n; i++){
55         sort(g[i].begin(), g[i].end());
56     }
57     dfs(s,0,1,k);
58     if(kt == 0) cout << -1;
59 }
```

```
60
61 int main()
62 {
63     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
64     freopen(Kurumi".inp","r",stdin);
65     freopen(Kurumi".out","w",stdout);
66     kurumi();
67     return 0;
68 }
69
```

C++ source file length : 1,530 lines : 69

## 10. Hành trình từ S tới T khi vài đảo bị phong tỏa.. DFSSECUR.\*

Tại quần đảo ZXY có N hòn đảo, một số hòn đảo đã có cầu nối với nhau, không có chu trình. Từ đảo này ta có thể đi sang đảo khác bằng đường đi trực tiếp hoặc đi gián tiếp qua các đảo khác.

Jame vừa đến hòn đảo S, anh muốn đi bằng đường bộ đến các đảo khác. Khi đến một đảo, anh ấy có thể nghỉ ngơi ở đảo ấy hoặc đi tiếp sang các đảo khác. Tất nhiên, anh ấy chỉ muốn tham quan mỗi đảo một lần. Vì một số lí do đặc biệt, Ban quản lý đảo đã cấm du khách đến một số đảo.

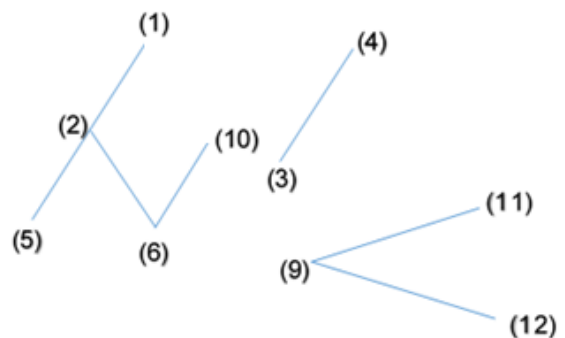
Hãy tìm các hành trình mà Jame có thể tham quan bằng đường bộ mà không đi vào các đảo bị cấm.

**Dữ liệu:** Vào từ file văn bản DFSSECUR.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M, S và K ( $M, N \leq 3000$ ,  $K, S \leq N$ ).
- Dòng 2: Ghi K số nguyên dương  $b_i$  là chỉ số các đảo bị cấm ( $b_i \leq N$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đảo u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSSECUR.OUT các hành trình mà Jame có thể tham quan bằng đường bộ mà không đi vào các đảo bị cấm. Mỗi số cách nhau một dấu cách. Nếu đảo S cũng bị cấm thì ghi ra -1.

DFSSECUR.INP	DFSSECUR.OUT
12 7 11 3	9 11
1 2 5	12 9 11
1 2	(truy vết ngược)
2 5	
2 6	
6 10	
3 4	
9 11	
9 12	



### Thuật toán:

- \* Dùng mảng  $ck[i] = \text{true}$  nếu đảo i không bị cấm, ngược lại  $ck[i] = \text{false}$ ;
- \* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ  $u$  tới  $v$ .

$ck[u] = ck[v] = \text{true}$  và  $a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp và được phép đi từ  $u$  tới  $v$  và chiều ngược lại.

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

{

Lưu đường đi tới  $j$  là  $i$  vào mảng  $vt[j] := i$ ;

Truy vết, đưa hành trình từ đỉnh  $j$  về đỉnh  $i$ .

Duyệt ( $j$ ).

}

Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Nếu  $S$  bị cấm thì đưa ra -1.

Nếu  $S$  không bị cấm,  $S$  không đi được đến các đảo khác thì đưa  $S$ .

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSSECUR.pas 1=[+]-
Const
  fi = 'DFSSECUR.inp'; fo = 'DFSSECUR.out'; nmax = 3000;
Var
  m, n, s, k, u, v, i, j: longint; p: longint = 0; q: longint = 0;
  a: array[1..nmax, 1..nmax] of longint;
  fre, ck: array[1..nmax] of boolean;
  vt: array[1..nmax] of longint;
Procedure trace(t: longint);
Begin
  while t <> s do
  begin
    write(t, #32); p := 1;
    t := vt[t];
  end;
  writeln(s);
End;
Procedure Duet(i: longint);
Var j: longint;
Begin
  fre[i] := False; q := 1;
  for j := 1 to n do
    if (a[i, j] = 1) and fre[j] then
    begin
      vt[j] := i;
      trace(j);
      Duet(j);
    end;
  end;
End;
Begin
  assign(input, fi); reset(input);
  assign(output, fo); rewrite(output);
  readln(n, m, s, k);
  fillchar(fre, sizeof(fre), True);
  fillchar(ck, sizeof(ck), True);
  for i := 1 to k do
  begin
    read(u);
    ck[u] := false;
  end;
  for i := 1 to m do
  begin
    readln(u, v);
    if ck[u] and ck[v] then
    begin
      a[u, v] := 1;
      a[v, u] := 1;
    end;
  end;
  duet(s);
  if not (ck[s]) then write(-1)
  else
    if (p = 0) and (q = 1) then write(s);
  close(input); close(output);
End.
31:9 =
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

```

## Code 1 tham khảo:

```
D:\THANG82020\bailam\bac\DFSSECUR.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSSECUR.cpp
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  vector <int> p[3005];
6  map <int, bool> checked;
7  int n, maxx, k, m, s;
8  vector <int> res;
9  int dem = 0;
10
11 void input()
12 {
13     cin >> n >> m >> s >> k;
14     for(int i = 1; i <= k; i++)
15     {
16         int f;
17         cin >> f;
18         checked[f] = true;
19     }
20     for(int i=0; i<m; i++)
21     {
22         int a, b;
23         cin >> a >> b;
24         p[a].push_back(b);
25         p[b].push_back(a);
26     }
27 }
28 int cc = 0;
29
30 void dfs(int i)
31 {
32     checked[i] = true;
33     res.push_back(i);
34     cc++;
35     if(res.size() > 1)
36     {
37         dem++;
38         for(int y = res.size() - 1; y >= 0; y--)
39             cout << res[y] << ' ';
40         cout << endl;
41     }
42     for(int j = 0; j < p[i].size(); j++)
43         if(checked[p[i][j]] == false)
44             dfs(p[i][j]);
45     res.pop_back();
46     checked[i] = false;
47     cc--;
48 }
49 void solve()
50 {
51     dfs(s);
52     if(dem == 0) cout << s;
53 }
54 int main()
55 {
56     freopen("DFSSECUR.inp", "r", stdin);
57     freopen("DFSSECUR.out", "w", stdout);
58     input();
59     solve();
60     return 0;
61 }
```

C++ source file

## Code 2 tham khảo:

```
D:\THANG82020\bailam\thao\DFSSECUR.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSSECUR.cpp
1  #include <bits/stdc++.h>
2  #define ft first
3  #define sc second
4  using namespace std;
5  int n,m,l,s;
6  int v[3009];
7  int t[3009];
8  deque<int> b;
9  int dem=0;
10 vector<int>a[3009];
11 void dfs(int i)
12 {
13     v[i]=1;
14     b.push_back(i);
15     if(b.size()>1)
16     {
17         dem=1;
18         for(int i=b.size()-1;i>=0;i--) cout<<b[i]<<" ";
19         cout<<"\n";
20     }
21     for(int j=0; j<a[i].size(); j++)
22     {
23         if(v[a[i][j]]==0)
24         {
25             dfs(a[i][j]);
26         }
27     }
28     b.pop_back();
29     v[i]=0;
30 }
31 int main()
32 {
C++ source file lengt
```

```
33     freopen("DFSSECUR.inp","r",stdin);
34     freopen("DFSSECUR.out","w",stdout);
35     cin>>n>>m>>s>>l;
36     while( l-->0)
37     {
38         int u;
39         cin>>u;
40         if(u==s)
41         {
42             cout<<-1;
43             return 0;
44         }
45         t[u]=1;
46     }
47     while(m-->0)
48     {
49         int u,v;
50         cin>>u>>v;
51         if(t[u]==0&&t[v]==0)
52         {a[u].push_back(v);
53          a[v].push_back(u);}
54     }
55     sort(a[s].begin(),a[s].end());
56     dfs(s);
57     if(dem==0) cout<<s;
58     return 0;
59 }
60
C++ source file lengt
```



### 11. Các vùng liên thông có tổng lớn nhất.. DFSSMAX.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

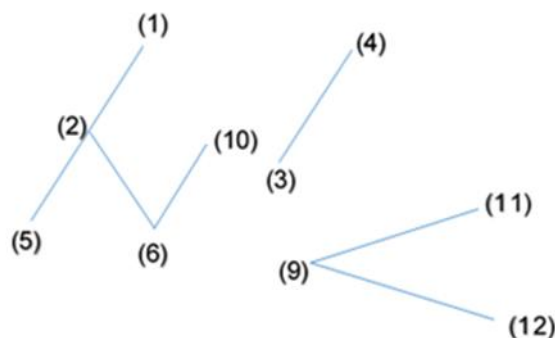
Cho đồ thị vô hướng có N đỉnh, M cạnh. Các đỉnh được đánh số từ 1 tới N. Hãy tìm vùng liên thông có tổng chỉ số lớn nhất. Đưa tổng lớn nhất ra.

**Dữ liệu:** Vào từ file văn bản DFSSMAX.INP gồm:

- Dòng 1: Ghi số nguyên N và M ( $M, N \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSSMAX.OUT tổng chỉ số lớn nhất tìm được.

DFSSMAX.INP	DFSSMAX.OUT
12 7	32
1 2	(Vùng liên thông
2 5	9, 11, 12)
2 6	
6 10	
3 4	
9 11	
9 12	



#### Thuật toán:

Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

Mỗi lần Duyệt xong một vùng liên thông, ta tính tổng chỉ số các đỉnh, tìm resmax và khởi tạo lại  $K = 0$ ;

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSSMAX.PAS 1=1

Uses      math;
Const
    fi = 'DFSSMAX.inp';
    fo = 'DFSSMAX.out';
    nmax = 30000;
Var
    m, n: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    c: array[1..nmax] of longint;
    u, v, i, j, temp: longint;
    k: longint = 0;
    resmax: longint = low(longint);
Procedure Duet(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    inc(k);
    c[k] := i;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
            Duet(j);
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m);
    fillchar(fre, sizeof(fre), True);
    for i := 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
            a[v, u] := 1;
        end;
    for i := 1 to n do
        if fre[i] then
            begin
                Duet(i);
                temp := 0;
                for u := 1 to k do inc(temp, c[u]);
                resmax := max(resmax, temp);
                k := 0;
            end;
    write(resmax);
    close(input); close(output);
End.
```

## Code 1 tham khảo:

```
D:\THANG82020\baillam\quang\DFSSMAX.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSSMAX.cpp
1
2 #include <bits/stdc++.h>
3 #define ll long long
4 #define for(x,a,b) for(ll x=a;x<=b;x++)
5 #define task "DFSSMAX"
6
7 const ll oo=1e18;
8 const ll mod=1e9;
9 const ll nmax=3007;
10 int n,t,u,v,res=0,last,head,s,trace[nmax],f;
11 ll tmp,mx=-oo;
12 bool fre[nmax],a[nmax][nmax],kt;
13
14 using namespace std;
15
16 void isReadFile()
17 {
18     freopen(task".INP","r",stdin);
19     freopen(task".OUT","w",stdout);
20 }
21
22 void dfs(int i) {
23     fre[i] = 0;
24     for(j,1,n) {
25         if ((fre[j]) && (a[i][j])) {
26             trace[j] = i;
27             dfs(j);
28         }
29     }
30 }
31
32 void loli()
33 {
34     memset(fre,1,sizeof(fre));
35     cin >> n >> t;
36     for(i,1,t) {
37         cin >> u >> v;
38         a[u][v]=1;
39         a[v][u]=1;
40     }
41     for(k,1,n) {
42         if (fre[k]==0) continue;
43         memset(trace,0,sizeof(trace));
44         dfs(k);
45         tmp=k;
46         for(j,2,n) {
47             if (trace[j]) tmp+=j;
48         }
49         // cout << tmp << endl;
50         mx=max(mx,tmp);
51     }
52     cout << mx;
53 }
54
55
56 int main()
57 {
58     ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
59     isReadFile();
60     loli();
61     return 0;
62 }
C++ source file length: 1,136 lines: 64
```

Code 2 tham khảo:

```
*D:\THANG82020\bailam\bac\DFSSMAX.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window

DFSSMAX.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3  bool a[1001][1001];
4  bool check[1001];
5  map <int, bool> check2;
6  int n, m, lab=0, maxx = 0;
7  void dfs(int i)
8  {
9      check[i] = true;
10     check2[i] = true;
11     for (int j = 1; j <= n; j++)
12         if (a[i][j] && !check[j])
13             dfs(j);
14 }
15 int main()
16 {
17     ios_base::sync_with_stdio(false);
18     cin.tie(NULL);
19     cout.tie(NULL);
20
21     freopen("DFSSMAX.INP", "r", stdin);
22     freopen("DFSSMAX.OUT", "w", stdout);
23
24     cin >> n >> m;
25     for (int i=1; i<=m; i++)
26     {
27         int u, v;
28         cin >> u >> v;
29         a[u][v] = 1;
30         a[v][u] = 1;
31     }
32
33     for (int i=1; i<=n; i++)
34     {
35         if (!check[i])
36         {
37             check2.clear();
38             lab++;
39             dfs(i);
40             int s = 0;
41             for(int j = 1; j <= n; j++)
42             {
43                 if(check2[j] == true)
44                 {
45                     s += j;
46                 }
47             }
48             if(prime(s) == true)
49             {
50                 maxx = max(maxx, s);
51             }
52         }
53     }
54     cout << maxx;
55     return 0;
56 }
```

C++ source file

### Code 3 tham khảo:

```

1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSSMAX"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m;
23 Vi g[nmax];
24 Vi gg;
25 bool check[nmax];
26 void dfs(int u, int par){
27     gg.pb(u);
28     check[u] = 1;
29     for(auto xx : g[u]){
30         if(xx != par && check[xx] == 0) dfs(xx,u);
31     }
32 }
33
34 void kurumi(){
35     cin >> n >> m;
36
37     for(int i = 1; i <= m; i++){
38         int u,v;
39         cin >> u >> v;
40         g[u].pb(v);
41         g[v].pb(u);
42     }
43     ll Lelouch = -OO;
44     for(int i = 1; i <= n; i++){
45         if(check[i] == 0){
46             gg.clear();
47             dfs(i,0);
48             sort(gg.begin(), gg.end());
49             ll Shido = 0;
50             for(auto xx : gg){
51                 Shido += xx;
52             }
53             Lelouch = max(Lelouch, Shido);
54             cout << '\n';
55         }
56     }
57     cout << Lelouch;
58 }
59 int main()
60 {
61     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
62     freopen(Kurumi".inp","r",stdin);
63     freopen(Kurumi".out","w",stdout);    kurumi();
64     return 0;
65 }

```

C++ source file      length : 1,538    lines : 65

## 12. Vùng nguyên tố lớn nhất.. DFSPRIME.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

Cho đồ thị vô hướng có N đỉnh, M cạnh. Các đỉnh được đánh số từ 1 tới N.

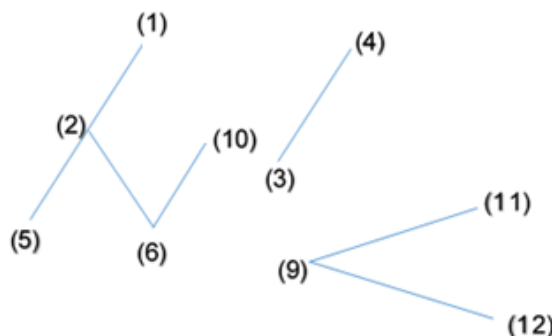
Hãy tìm vùng liên thông có tổng chỉ số là số nguyên tố lớn nhất. Đưa tổng lớn nhất ra.

**Dữ liệu:** Vào từ file văn bản DFSPRIME.INP gồm:

- Dòng 1: Ghi số nguyên N và M ( $M, N \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSPRIME.OUT tổng chỉ số nguyên tố lớn nhất tìm được. Nếu không có ghi ra 0.

DFSPRIME.INP	DFSPRIME.OUT
12 7	7
1 2	(Vùng liên thông
2 5	3, 4 hoặc vùng 7)
2 6	
6 10	
3 4	
9 11	
9 12	



### Thuật toán:

Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

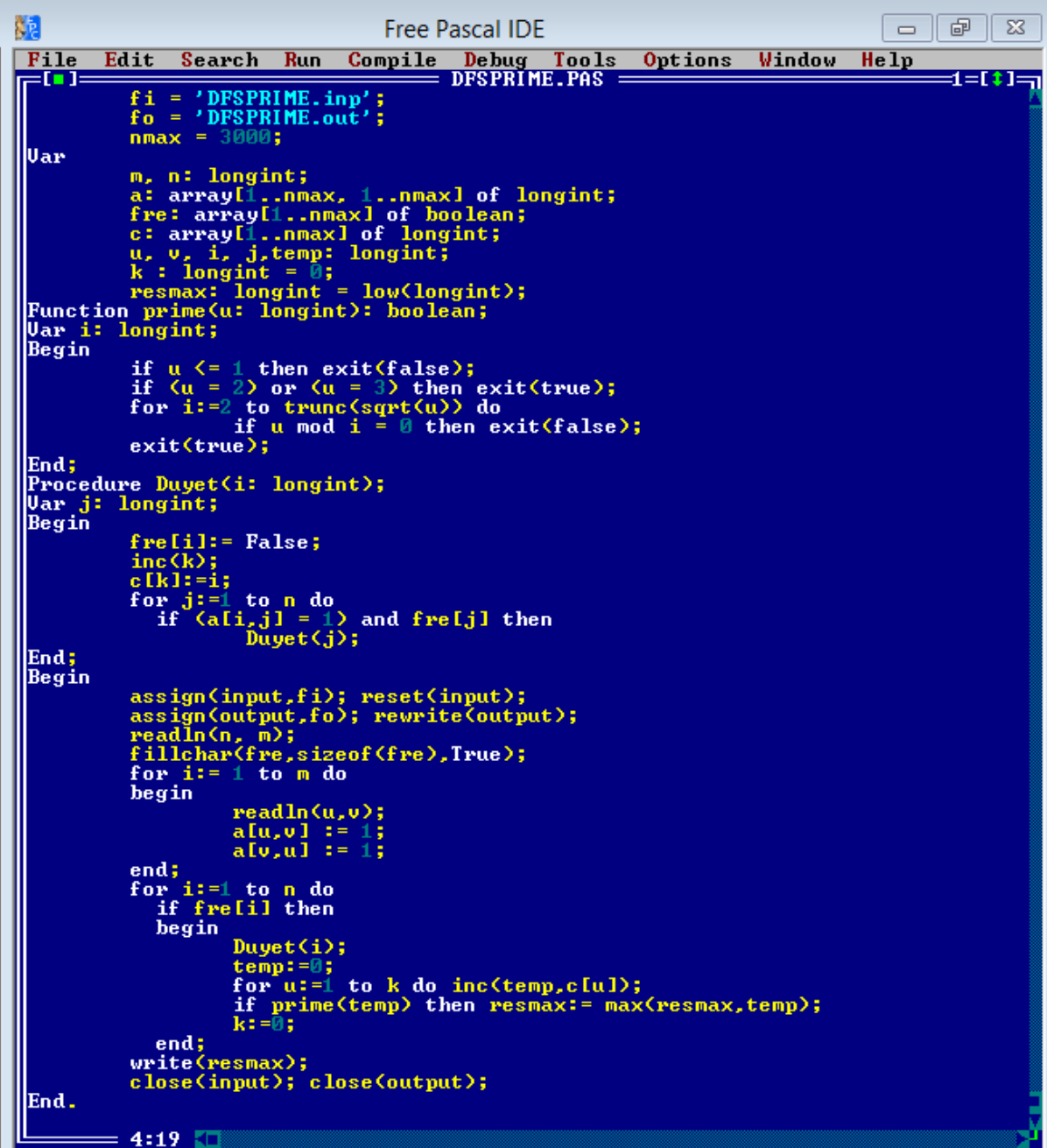
Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

Mỗi lần Duyệt xong một vùng liên thông, ta tính tổng chỉ số các đỉnh, ta kiểm tra tính nguyên tố của tổng, cập nhật resmax và khởi tạo lại  $K = 0$ ;

Độ phức tạp thuật toán:  $O(M + N)$

Để cải tiến: Ta nên dùng Sàng Eratosthenes tới  $10^7$ . Để thu được mảng F[i] lưu tính nguyên tố của số i. Khi đó, độ phức tạp thuật toán chỉ còn là  $O(M + N + N) \sim O(M + N)$

Code chương trình tham khảo:



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSPRIME.PAS
fi = 'DFSPRIME.inp';
fo = 'DFSPRIME.out';
nmax = 3000;

Var
  n, m: longint;
  a: array[1..nmax, 1..nmax] of longint;
  fre: array[1..nmax] of boolean;
  c: array[1..nmax] of longint;
  u, v, i, j, temp: longint;
  k: longint = 0;
  resmax: longint = low(longint);

Function prime(u: longint): boolean;
Var i: longint;
Begin
  if u <= 1 then exit(false);
  if (u = 2) or (u = 3) then exit(true);
  for i:=2 to trunc(sqrt(u)) do
    if u mod i = 0 then exit(false);
  exit(true);
End;

Procedure Duyet(i: longint);
Var j: longint;
Begin
  fre[i] := False;
  inc(k);
  c[k] := i;
  for j:=1 to n do
    if (a[i, j] = 1) and fre[j] then
      Duyet(j);
End;

Begin
  assign(input, fi); reset(input);
  assign(output, fo); rewrite(output);
  readln(n, m);
  fillchar(fre, sizeof(fre), True);
  for i:= 1 to m do
    begin
      readln(u, v);
      a[u, v] := 1;
      a[v, u] := 1;
    end;
  for i:=1 to n do
    if fre[i] then
      begin
        Duyet(i);
        temp:=0;
        for u:=1 to k do inc(temp, c[u]);
        if prime(temp) then resmax:= max(resmax, temp);
        k:=0;
      end;
  write(resmax);
  close(input); close(output);
End.
```

4:19

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

Code 1 tham khảo:

```
D:\THANG82020\bailam\ha\DFSPRIME.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSPRIME.cpp x
1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSPRIME"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m;
23 Vi g[nmax];
24 Vi gg;
25 bool check[nmax];
26
27 void dfs(int u, int par){
28     gg.pb(u);
29     check[u] = 1;
30     for(auto xx : g[u]){
31         if(xx != par && check[xx] == 0) dfs(xx,u);
32     }
```



```

34
35 void kurumi(){
36     cin >> n >> m;
37
38     for(int i = 1; i <= m; i++){
39         int u,v;
40         cin >> u >> v;
41         g[u].pb(v);
42         g[v].pb(u);
43     }
44     ll Lelouch = 0;
45     for(int i = 1; i <= n; i++){
46         if(check[i] == 0){
47             gg.clear();
48             dfs(i,0);
49             sort(gg.begin(), gg.end());
50             ll Shido = 0;
51             int kt = 0;
52             for(auto xx : gg){
53                 Shido += xx;
54             }
55             for(int j = 2; j <= sqrt(Shido); j++){
56                 if(Shido % j == 0){
57                     kt = 1;
58                     break;
59                 }
60             }
61             if(kt == 0) Lelouch = max(Lelouch, Shido);
62             cout << '\n';
63         }
64     }
65     cout << Lelouch;

```

```

68 int main()
69 {
70     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
71     freopen(Kurumi".inp","r",stdin);
72     freopen(Kurumi".out","w",stdout);
73     kurumi();
74     return 0;
75 }
76

```

C++ source file

length : 1,760 lines : 76

## Code 2 tham khảo:

```
DFSPRIME.cpp x DFSPRIME.cpp x
1
2 #include <bits/stdc++.h>
3
4 using namespace std;
5
6 bool a[1001][1001];
7 bool check[1001];
8 map<int, bool> check2;
9 int n, m, lab=0, maxx = 0;
10
11 void dfs(int i)
12 {
13     check[i] = true;
14     check2[i] = true;
15     for (int j = 1; j <= n; j++)
16         if (a[i][j] && !check[j])
17             dfs(j);
18 }
19
20 bool prime(int s)
21 {
22     if(s < 2)
23         return false;
24     if(s == 2)
25         return true;
26     for(int i = 2; i * i <= s; i++)
27         if(s % i == 0)
28             return false;
29     return true;
30 }
31
32 int main()
33 {
34     ios_base::sync_with_stdio(false);
35     cin.tie(NULL);
36     cout.tie(NULL);
37
38     freopen("DFSPRIME.INP", "r", stdin);
39     freopen("DFSPRIME.OUT", "w", stdout);
40
41     cin >> n >> m;
42     for (int i=1; i<=m; i++)
43     {
44         int u, v;
45         cin >> u >> v;
46         a[u][v] = 1;
47         a[v][u] = 1;
48     }
49     for (int i=1; i<=n; i++)
50         if (!check[i])
51         {
52             check2.clear();
53             lab++;
54             dfs(i);
55             int s = 0;
56             for(int j = 1; j <= n; j++)
57             {
58                 if(check2[j] == true)
59                 {
60                     s += j;
61                 }
62             }
63         }
64 }
```

C++ source file length: 1,34

```
63         if(prime(s) == true)
64         {
65             maxx = max(maxx, s);
66         }
67     }
68 }
69 cout << maxx;
70 return 0;
71 }
72
```

C++ source file | length: 1,341

### 13. Vùng có tổng chi phí nhỏ hơn hoặc bằng S.. DFSTS.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

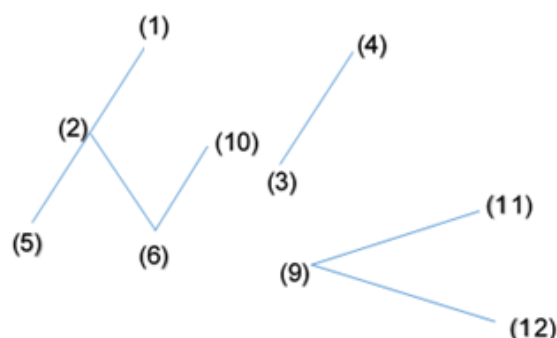
Cho đồ thị vô hướng có N đỉnh, M cạnh. Các đỉnh được đánh số từ 1 tới N. Chi phí thăm đỉnh thứ i là  $t[i]$ . Hãy tìm các vùng liên thông có tổng chi phí nhỏ hơn hoặc bằng S.

**Dữ liệu:** Vào từ file văn bản DFSTS.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M và S ( $M, N \leq 3000, S \leq 10^9$ ).
- Dòng 2: Ghi N số nguyên dương  $t_i$  là chi phí thăm đỉnh i ( $t_i \leq 10^9$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSTS.OUT các vùng liên thông tìm được. Mỗi vùng liên thông ghi trên một hàng. Mỗi số cách nhau một dấu cách. Nếu không có ghi ra -1.

DFSTS.INP	DFSTS.OUT
12 7 5	3 4
1 2 3 1 2 4 4 4 3 3 2 1	7
1 2	8
2 5	
2 6	
6 10	
3 4	
9 11	
9 12	



#### Thuật toán:

Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

Mỗi lần Duyệt xong một vùng liên thông, ta tính tổng chi phí các đỉnh, ta kiểm tra tổng  $\leq S$  thì đưa cấu hình đó ra và khởi tạo lại  $K = 0$ ;

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo

```
Free Pascal IDE

Const
    fi = 'DFSTS.inp';
    fo = 'DFSTS.out';
    nmax = 3000;
Var
    m, n, S: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    c, t: array[1..nmax] of longint;
    u, v, i, j, temp: longint;
    k: longint = 0;
    p: longint = 0;
Procedure Duet(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    inc(k);
    clk := i;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
            Duet(j);
    End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s);
    for i := 1 to n do read(t[i]);
    fillchar(fre, sizeof(fre), True);
    for i := 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
            a[v, u] := 1;
        end;
    for i := 1 to n do
        if fre[i] then
            begin
                Duet(i);
                temp := 0;
                for u := 1 to k do inc(temp, t[clk]);
                if temp <= S then
                    begin
                        for u := 1 to k do write(c[u], #32);
                        writeln();
                        p := 1;
                    end;
                k := 0;
            end;
        if p = 0 then write(-1);
    close(input); close(output);
End.

13:1 =
```

## Code 1 tham khảo:

```
*D:\THANG82020\baclam\bacl\DFSTS.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSTS.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3  bool a[1001][1001];
4  int t[1001];
5  bool check[1001];
6  map <int, bool> check2;
7  int n, m, lab=0, maxx = 0, g;
8  void dfs(int i)
9  {
10     check[i] = true;
11     check2[i] = true;
12     for (int j = 1; j <= n; j++)
13         if (a[i][j] && !check[j])
14             dfs(j);
15 }
16 int main()
17 {
18     ios_base::sync_with_stdio(false);
19     cin.tie(NULL);
20     cout.tie(NULL);
21
22     freopen("DFSTS.INP", "r", stdin);
23     freopen("DFSTS.OUT", "w", stdout);
24
25     cin >> n >> m >> g;
26     for(int i = 1; i <= n; i++)
27         cin >> t[i];
28
29     for (int i=1; i<=m; i++)
30     {
31         int u, v;
32         cin >> u >> v;
33         a[u][v] = 1;
34         a[v][u] = 1;
35     }
36     int dem = 0;
37     for (int i=1; i<=n; i++)
38         if (!check[i])
39         {
40             check2.clear();
41             lab++;
42             dfs(i);
43             int s = 0;
44             for(int j = 1; j <= n; j++)
45             {
46                 if(check2[j] == true)
47                     s += t[j];
48             }
49             if(s <= g)
50             {
51                 dem++;
52                 for(int j = 1; j <= n; j++)
53                 {
54                     if(check2[j] == true)
55                     {
56                         cout << j << ' ';
57                     }
58                 }
59                 cout << '\n';
60             }
61         }
62     if(dem == 0)
63         cout << -1;
64     return 0;
65 }
```

C++ source file

## Code 2 tham khảo:

```
D:\THANG82020\bailam\hai\DFSTS.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSTS.cpp
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSTS"
18 #define nmax 50005
19 const ll OO = 1e15+7;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m,s;
23 Vi g[nmax];
24 int t[nmax];
25 Vi gg;
26 bool check[nmax];
27
28 void dfs(int u, int par){
29
30     gg.pb(u);
31     check[u] = 1;
32     for(auto xx : g[u]){
33         if(xx != par && check[xx] == 0) dfs(xx,u);
34     }
35 }
36
37 void kurumi(){
38     cin >> n >> m >> s;
39     for(int i = 1; i <= n; i++){
40         cin >> t[i];
41     }
42     for(int i = 1; i <= m; i++){
43         int u,v;
44         cin >> u >> v;
45         g[u].pb(v);
46         g[v].pb(u);
47     }
48     int kt = 0;
49     for(int i = 1; i <= n; i++){
50         if(check[i] == 0){
51             gg.clear();
52             dfs(i,0);
53             sort(gg.begin(), gg.end());
54             ll Shido = 0;
55             for(auto xx : gg){
56                 Shido += t[xx];
57             }
58             if(Shido <= s){
59                 kt = 1;
60                 for(auto xx : gg){
61                     cout << xx << ' ';
62                 }
63                 cout << '\n';
64             }
65         }
66     }
67
68     if(kt == 0) cout << -1;
69 }
70
C++ source file
```

#### 14. Vé miễn phí.. DFSTK.\*

Vùng liên thông trong đồ thị là tập hợp các đỉnh mà từ một đỉnh bất kỳ có đường đi trực tiếp hoặc gián tiếp đến các đỉnh khác trong tập hợp đó.

Cho đồ thị vô hướng có N đỉnh, M cạnh. Các đỉnh được đánh số từ 1 tới N. Chi phí thăm đỉnh thứ i là  $t[i]$ . Bạn có một vé miễn phí để dùng một lần khi thăm 1 đỉnh.

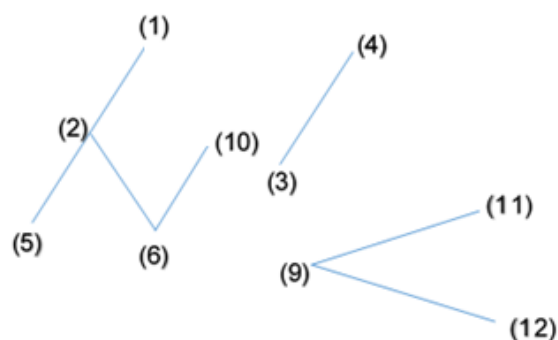
Bạn ghé thăm các đỉnh trong vùng liên thông có nhiều đỉnh nhất và sử dụng vé miễn phí để tổng chi phí nhỏ nhất.

**Dữ liệu:** Vào từ file văn bản DFSTK.INP gồm:

- Dòng 1: Ghi số nguyên dương N, M ( $M, N \leq 3000$ ).
- Dòng 2: Ghi N số nguyên dương  $t_i$  là chi phí thăm đỉnh i ( $t_i \leq 10^9$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện có đường đi giữa hai đỉnh u và v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSTK.OUT số đỉnh được thăm nhiều nhất và tổng chi phí nhỏ nhất tìm được.

DFSTK.INP	DFSTK.OUT
12 7	5 8
1 2 3 1 2 4 4 4 3 3 2 1	(Chọn vùng nhiều đỉnh nhất:
1 2	1, 2, 5, 6, 10;
2 5	Chi phí $1 + 2 +$
2 6	$2 + 4 + 3 = 12$ .
6 10	Dùng vé miễn
3 4	phí cho đỉnh 6,
9 11	thu được tổng
9 12	chi phí: $12 - 4 = 8$ )



#### Thuật toán:

Ta duyệt qua tất cả các đỉnh, nếu i đang free thì ta Duyệt (i).

Thủ tục Duyệt (i) ngoài việc đánh dấu i đã đi đến, ta lưu i vào mảng C có K phần tử.

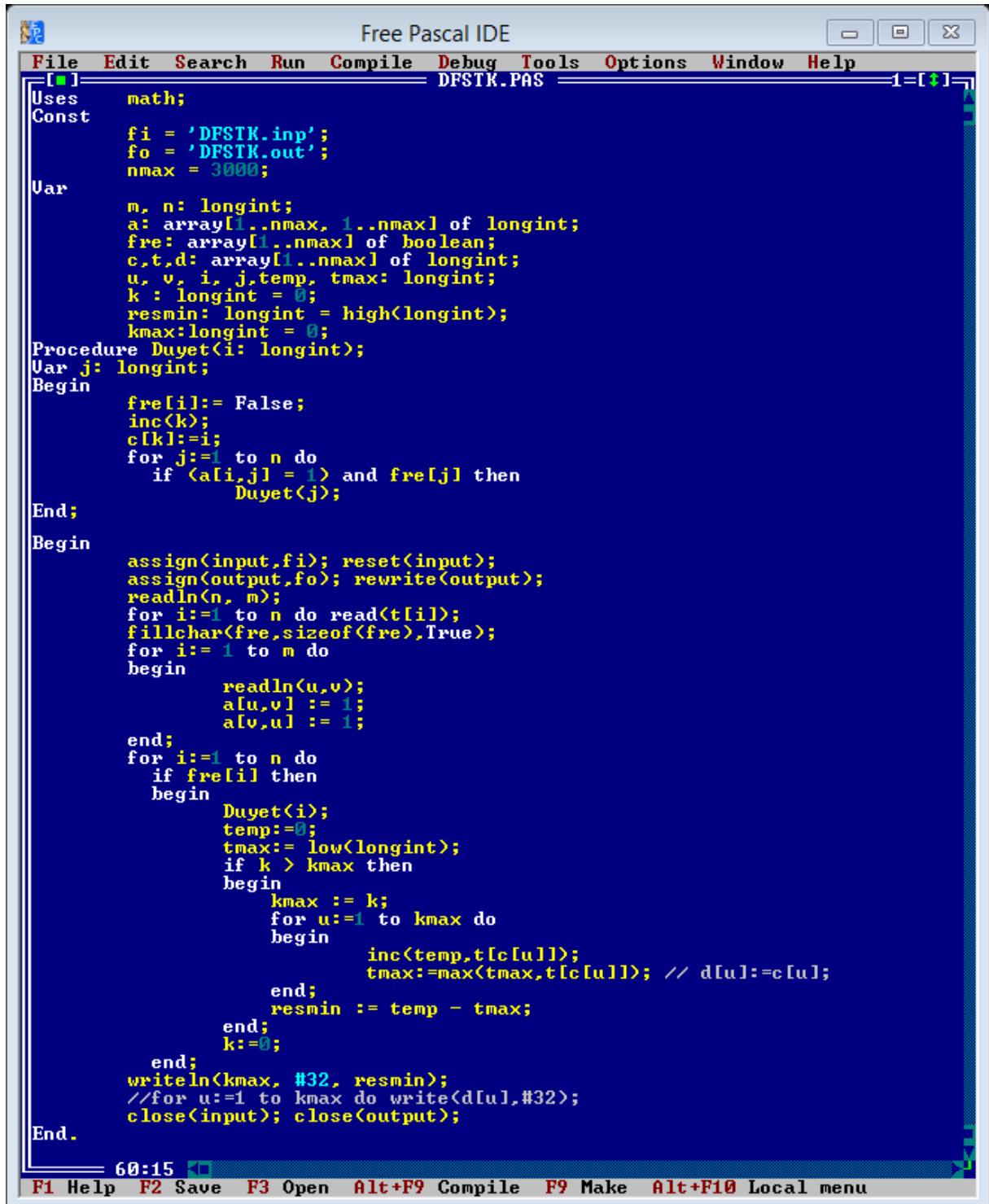
Mỗi lần Duyệt xong một vùng liên thông, ta cập nhật kmax,  $t[i]_{\max}$  là giá vé cao nhất trong vùng liên thông đó; Tổng chi phí: tổng các đỉnh trong vùng liên



thông đó –  $t[i] \max$ ; vì ta sử dụng vé miễn phí thăm đỉnh có chi phí lớn nhất trong mỗi vùng liên thông có nhiều đỉnh nhất.

Khởi tạo lại  $K = 0$ ; làm tương tự với các cấu hình khác.

Độ phức tạp thuật toán:  $O(M + N)$



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSTK.PAS 1=1

Uses
Const
  math;
  fi = 'DFSTK.inp';
  fo = 'DFSTK.out';
  nmax = 3000;

Var
  m, n: longint;
  a: array[1..nmax, 1..nmax] of longint;
  fre: array[1..nmax] of boolean;
  c, t, d: array[1..nmax] of longint;
  u, v, i, j, temp, tmax: longint;
  k: longint = 0;
  resmin: longint = high(longint);
  kmax: longint = 0;

Procedure Duet(i: longint);
Var j: longint;
Begin
  fre[i] := False;
  inc(k);
  c[k] := i;
  for j := 1 to n do
    if (a[i, j] = 1) and fre[j] then
      Duet(j);
End;

Begin
  assign(input, fi); reset(input);
  assign(output, fo); rewrite(output);
  readln(n, m);
  for i := 1 to n do read(t[i]);
  fillchar(fre, sizeof(fre), True);
  for i := 1 to m do
    begin
      readln(u, v);
      a[u, v] := 1;
      a[v, u] := 1;
    end;
  for i := 1 to n do
    if fre[i] then
      begin
        Duet(i);
        temp := 0;
        tmax := low(longint);
        if k > kmax then
          begin
            kmax := k;
            for u := 1 to kmax do
              begin
                inc(temp, t[c[u]]);
                tmax := max(tmax, t[c[u]]); // d[u] := c[u];
              end;
            resmin := temp - tmax;
          end;
        k := 0;
      end;
  writeln(kmax, #32, resmin);
  //for u:=1 to kmax do write(d[u], #32);
  close(input); close(output);
End.
```

60:15

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

## Code 1 tham khảo:

```
*D:\THANG82020\bailam\tu\DFSTK.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSTK.cpp
1  #include <bits/stdc++.h>
2
3  using namespace std;
4  bool a[3009][3009], kt[3003];
5  int t[3003];
6  vector<int> b;
7  long long kq=999999999;
8  int n,m,dem,res=-1,dem1,maxx=0,s;
9  int dfs(int s)
10 {
11     kt[s]=true;
12     b.push_back(s);
13     for (int j=1;j<=n;j++)
14         if (a[s][j]==1 && kt[j]==false)
15             dfs(j);
16     return 0;
17 }
18
19 long long tinh(vector <int> b){
20
21     int tong=0,maxxx=0;
22     for (int i=0;i<b.size();i++){
23         tong+=t[b[i]];
24         maxxx=max(maxxx,t[b[i]]);
25     }
26     return tong-maxxx;
27 }
28
29 int main()
30 {
31     freopen("DFSTK.INP","r",stdin);
32     freopen("DFSTK.OUT","w",stdout);
33     ios_base::sync_with_stdio(false);
34     cin.tie(NULL);
35     cout.tie(NULL);
36     cin>>n>>m;
37     int tmp,tmp1;
38     for (int i=1; i<=n; i++) cin>>t[i];
39
40     for (int i=1; i<=m; i++)
41     {
42         cin>>tmp>>tmp1;
43         a[tmp][tmp1]=1;
44         a[tmp1][tmp]=1;
45     }
46     for (int i=1;i<=n;i++)
47     if (!kt[i]){
48         b.clear();
49         dfs(i);
50         if (b.size()>=maxx){
51             if (b.size()==maxx) kq=min(kq,tinh(b));
52             if (b.size()>maxx) {
53                 maxx=b.size();
54                 kq=tinh(b);
55             }
56         }
57     }
58 }
```

C++ source file length: 1

## Code 2 tham khảo:

```
D:\THANG82020\bailam\hai\DFSTK.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSTK.cpp
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSTK"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m,t[nmax];
23 Vi g[nmax];
24 Vi gg;
25 Vi LL;
26 bool check[nmax];
27 //bool visit[nmax];
28 void dfs(int u, int par){
29     gg.pb(u);
30     check[u] = 1;
31     // visit[u] = 0;
32     for(auto xx : g[u]){
33         if(xx != par && check[xx] == 0) dfs(xx,u);
34     }
35 }
36
37 void kurumi(){
38     cin >> n >> m;
39     for(int i = 1; i <= n; i++){
40         cin >> t[i];
41     }
42     for(int i = 1; i <= m; i++){
43         int u,v;
44         cin >> u >> v;
45         g[u].pb(v);
46         g[v].pb(u);
47         // visit[u] = 1;
48         // visit[v] = 1;
49     }
50     int Lelouch = 1e9+7;;
51     for(int i = 1; i <= n; i++){
52         if(check[i] == 0){
53             gg.clear();
54             dfs(i,0);
55             sort(gg.begin(), gg.end());
56             if(gg.size() >= LL.size()){
57                 int kt = 0;
58                 if(LL.size() == gg.size()) kt = 1;
59                 LL.clear();
60                 for(auto xx : gg){
61                     LL.pb(xx);
62                 }
63                 int Shido = 0;
64                 int ma = -1e9;
65                 for(auto xx : LL){
66                     Shido += t[xx];
67                 }
68             }
69         }
70     }
71 }
```

```

67         ma = max(ma,t[xx]);
68     }
69     //         cout << Shido << ' ' << ma << '\n';
70     if(kt == 1) Lelouch = min(Lelouch, Shido - ma);
71     else Lelouch = Shido - ma;
72     //         cout << Lelouch << ' ';
73 }
74 }
75 }
76 cout << LL.size() << ' ';
77 cout << Lelouch << '\n';
78 }
79
80 int main()
81 {
82     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
83     freopen(Kurumi".inp","r",stdin);
84     freopen(Kurumi".out","w",stdout);
85     kurumi();
86     return 0;
87 }
88

```

C++ source file

length: 2,232 lines: 88

## 15. Xây cầu 2.. DFSBRG2.\*

Tại quần đảo ZXY có N hòn đảo, ban đầu một đảo có cầu nối với nhau. Để thuận tiện cho các phương tiện đi lại, ban quản lý lên kế hoạch xây K cầu để từ một đảo ta có thể đi đến các đảo còn lại trong quần đảo bằng đường đi trực tiếp hoặc đi gián tiếp qua các đảo khác.

Ban quản lý nhận thấy rằng: có thể không nhất thiết phải xây hết các cầu theo trình tự trong kế hoạch đưa ra mà vẫn đảm bảo giao thông đi lại giữa các đảo.

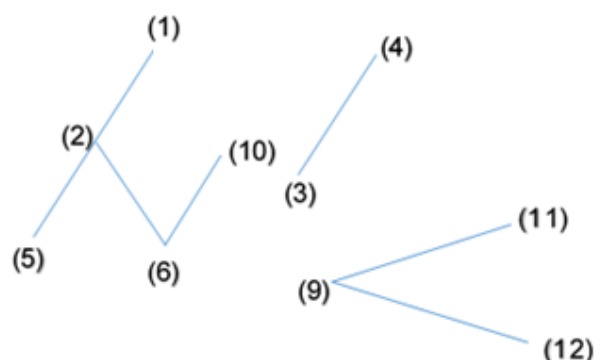
Cho kế hoạch có K cầu cần xây mới theo thứ tự đầu vào. Hãy cho biết, ban quản lý cần xây ít nhất bao nhiêu cầu mà vẫn đảm bảo giao thông đi lại giữa các đảo. Dữ liệu đảm bảo có nghiệm.

**Dữ liệu:** Vào từ file văn bản DFSBRG2.INP gồm:

- Dòng 1: Ghi số nguyên N, M và K ( $N, M, K \leq 3000$ ).
- M dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện đã có cầu nối đảo u và đảo v ( $u, v \leq N$ ).
- K dòng tiếp theo, mỗi dòng ghi số nguyên dương u và v thể hiện sẽ xây cầu nối đảo u và đảo v ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSBRG2.OUT số cầu ít nhất cần xây thêm.

DFSBRG2.INP	DFSBRG2.OUT
12 7 6	4
1 2	
2 5	
2 6	
6 10	
3 4	
9 11	
9 12	
<b>6 7</b>	
7 8	
3 8	
3 9	
4 10	
3 10	



**Thuật toán:** Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng a có N hàng, N cột với N là số đỉnh của đồ thị.

Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ  $u$  tới  $v$ .

$a[u,v] = a[v,u] = 1$  khi có đường đi trực tiếp từ  $u$  tới  $v$ .

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta Duyệt ( $j$ ).

Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

\* Thực hiện lần lượt đọc các cầu ( $u, v$ ) và ta cập nhật lại mảng  $a$ . Sau đó ta Duyệt đếm số vùng liên thông với mảng  $a$  mới này. Nếu số vùng liên thông bằng 1 hay từ một đảo có thể đi đến các đảo còn lại ta dừng lại và đưa số cầu cần xây ra.

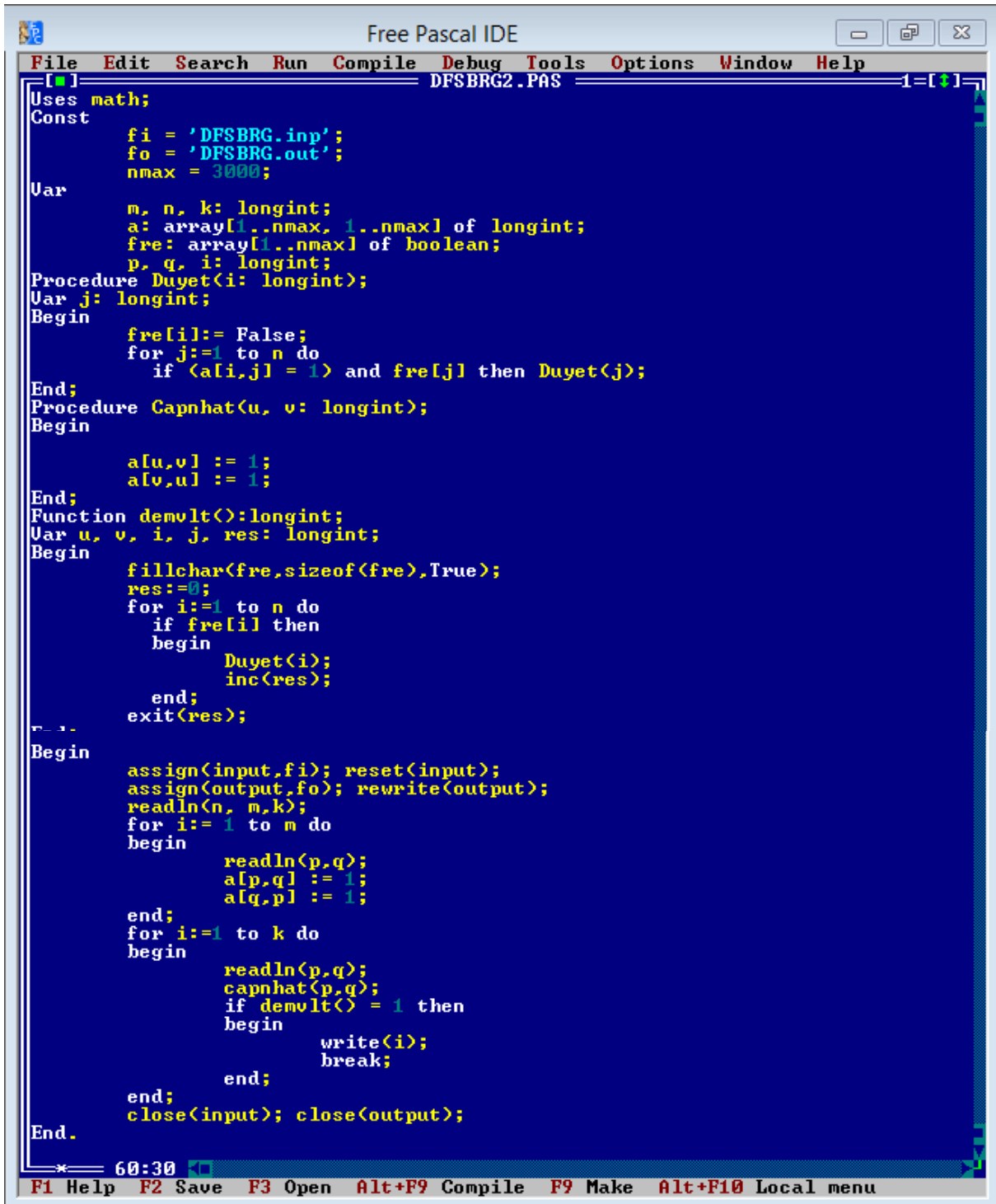
**Lưu ý:** Số cầu cần xây thêm không phải *số vùng liên thông - 1*, bởi lẽ dữ liệu cho có thể xây thêm các cầu trong một vùng liên thông nào đó.

Độ phức tạp thuật toán:  $O(M + N \times K)$

Cải tiến: Dùng Disjoint Set Union để làm bài này với chi phí tốt hơn.

Code chương trình tham khảo: Sửa trong code

Fi = 'dfsbrg2.inp'; fo = 'dfsdr2.out';



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSBRG2.PAS 1=[+]-
Uses math;
Const
  fi = 'DFSBRG.inp';
  fo = 'DFSBRG.out';
  nmax = 3000;
Var
  m, n, k: longint;
  a: array[1..nmax, 1..nmax] of longint;
  fre: array[1..nmax] of boolean;
  p, q, i: longint;
Procedure Duyet(i: longint);
Var j: longint;
Begin
  fre[i] := False;
  for j:=1 to n do
    if <a[i,j] = 1> and fre[j] then Duyet(j);
End;
Procedure Capnhat(u, v: longint);
Begin
  a[u,v] := 1;
  a[v,u] := 1;
End;
Function demvlt(): longint;
Var u, v, i, j, res: longint;
Begin
  fillchar(fre, sizeof(fre), True);
  res:=0;
  for i:=1 to n do
    if fre[i] then
      begin
        Duyet(i);
        inc(res);
      end;
  exit(res);
End;
Begin
  assign(input, fi); reset(input);
  assign(output, fo); rewrite(output);
  readln(n, m, k);
  for i:=1 to m do
    begin
      readln(p, q);
      a[p,q] := 1;
      a[q,p] := 1;
    end;
  for i:=1 to k do
    begin
      readln(p, q);
      capnhat(p, q);
      if demvlt() = 1 then
        begin
          write(i);
          break;
        end;
    end;
  close(input); close(output);
End.
*== 60:30
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```

## Code 1 tham khảo:

```
"D:\THANG82020\bailam\duong\DFSBRG2.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSBRG2.cpp
1  #include <bits/stdc++.h>
2  #define task "DFSBRG2"
3  #define pii pair<int,int>
4  using namespace std;
5  int m,n,k,u,v;
6  vector<int> g[3009];
7  int dad[3009];
8  int finddad(int u)
9  {
10     if (dad[u]<0) return u;
11     else return dad[u]=finddad(dad[u]);
12 }
13 vector< pair<int,int> > cau;
14 bool join(int u,int v)
15 {
16     int x,y;
17     x=finddad(u);
18     y=finddad(v);
19     if (x==y) return 0;
20     if (dad[x]<dad[y]) swap(x,y);
21     dad[y]+=dad[x];
22     dad[x]=y;
23
24     return 1;
25 }
26 int main()
27 {
28     freopen(task".inp","r",stdin);
29     freopen(task".out","w",stdout);
30     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
31
32     cin >>n>>m>>k;
33     int vlt=n;
34     for(int i=1;i<=n;i++) dad[i]=-1;
35     for (int i=1;i<=m;i++)
36     {
37         cin >>u>>v;
38         if(join(u,v)) vlt--;
39     }
40     if (vlt==1) { cout <<0; return 0;}
41     for (int i=1;i<=k;i++)
42     {
43         cin >>u>>v;
44         cau.push_back(make_pair(u,v));
45     }
46     // sort(cau.begin(),cau.end());
47     int ans=0;
48     for(pii a: cau)
49     {
50         u=a.first;
51         v=a.second;
52         ans++;
53         if (join(u,v) && vlt>1) vlt--;
54         if (vlt==1) break;
55     }
56     // if (join(u,v) && vlt>1) { vlt--; ans++; }
57     cout <<ans;
58     return 0;
59 }
```

C++ source file length: 1,240 line



## Code 2 tham khảo:

```
D:\THANG82020\bailam\thao\DFSBRG2.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSBRG2.cpp DFSBRG2.cpp
1  #include <bits/stdc++.h>
2
3  using namespace std;
4  int n,m,k,dem=0;
5  vector<int> a[3009];
6  map<int,bool> vi;
7  void dfs(int i)
8  {
9      vi[i]=true;
10     for(int j=0;j<a[i].size();j++)
11     {
12         if(!vi[a[i][j]]) dfs(a[i][j]);
13     }
14 }
15 int main()
16 {
17     freopen("DFSBRG2.inp","r",stdin);
18     freopen("DFSBRG2.out","w",stdout);
19     cin>>n>>m>>k;
20     while(m--)
21     {
22         int u,v;
23         cin>>u>>v;
24         a[u].push_back(v);
25         a[v].push_back(u);
26     }
27     while(k--)
28     {
29         vi.clear();
30         int u,v;
31         cin>>u>>v;
32         int ct=0;
33         dem++;
34         a[u].push_back(v);
35         a[v].push_back(u);
36         for(int i=1;i<=n;i++)
37             if(!vi[i])
38             {
39                 ct++;
40                 dfs(i);
41             }
42         if(ct==1)
43         {
44             cout<<dem;
45             return 0;
46         }
47     }
48     return 0;
49 }
50
C++ source file length
```

## 16. Đường đi trên đồ thị có hướng.. DFSORIEN.\*

Đồ thị có hướng là đồ thị mà tất cả các cạnh trong đồ thị có hướng. Cạnh  $(u, v)$  thể hiện chỉ có đường đi từ đỉnh  $u$  tới đỉnh  $v$ .

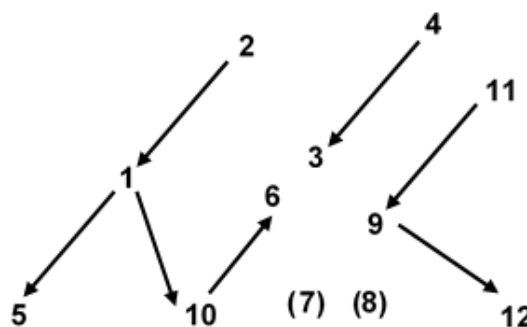
Cho đồ thị có hướng có  $N$  đỉnh,  $M$  cạnh. Hãy liệt kê các đường đi xuất phát từ đỉnh  $S$ .

**Dữ liệu:** Vào từ file văn bản DFSORIEN.INP gồm:

- Dòng 1: Ghi số nguyên dương  $N, M$  và  $S$  ( $M, N \leq 3000, S \leq N$ ).
- $M$  dòng tiếp theo, mỗi dòng ghi số nguyên dương  $u$  và  $v$  thể hiện có đường đi từ đỉnh  $u$  đến đỉnh  $v$  ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSORIEN.OUT các đường đi xuất phát từ đỉnh  $S$ . Mỗi đường đi trên một hàng. Mỗi số ghi cách nhau một dấu cách. Nếu từ  $S$  không có đường đi sang đỉnh khác thì ghi ra -1.

DFSORIEN.INP	DFSORIEN.OUT
12 7 2	1 2
2 1	5 1 2
1 5	10 1 2
1 10	6 10 1 2
10 6	(Truy vết ngược)
4 3	
11 9	
9 12	



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng  $a$  có  $N$  hàng,  $N$  cột với  $N$  là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp từ  $u$  tới  $v$ .

$a[u,v] = 1$  khi có đường đi trực tiếp từ  $u$  tới  $v$ .

Lưu ý:  $a[v,u]$  có thể khác  $a[u,v]$ .

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

```

{
    Lưu đường đi tới j là i vào mảng vt[j] := i;
    Truy vết, đưa hành trình từ đỉnh j về đỉnh i.
    Duyệt ( j ).
}

```

Tới đỉnh j, ta lặp lại việc duyệt như ở bước trên.

Khi không có đường đi từ S tới đỉnh khác ta ghi ra -1.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSORIE.nas
Const
    fi = 'DFSORIE.inp';
    fo = 'DFSORIE.out';
    nmax = 30000;
Var
    m, n, s: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j: longint;
Procedure trace(t: longint);
Begin
    while t <> s do
    begin
        write(t, #32);
        t := vt[t];
    end;
    writeln(s);
    vt[1] := 30000;
End;
Procedure Duyệt(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
        begin
            vt[j] := i;
            trace(j);
            Duyệt(j);
        end;
    end;
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s);
    fillchar(fre, sizeof(fre), True);
    for i := 1 to m do
    begin
        readln(u, v);
        a[u, v] := 1;
    end;
    duyet(s);
    if vt[1] = 0 then write(-1);
    close(input); close(output);
End.

```

Code 1 tham khảo:

```
D:\THANG82020\baillam\thao\DFSORIEN.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSORIEN.cpp
1  #include <bits/stdc++.h>
2  #define ft first
3  #define sc second
4  using namespace std;
5  int n,m,l,r;
6  int v[3009];
7  deque<int> b;
8  int dem=0;
9  int t[3009];
10 vector<int>a[3009];
11 void dfs(int i)
12 {
13     if(dem==1)
14     {
15         deque<int> c=b;
16         while(!c.empty())
17         {
18             cout<<c.back()<<" ";
19             c.pop_back();
20         }
21         cout<<"\n";
22     }
23     v[i]=1;
24     for(int j=0; j<a[i].size(); j++)
25     {
26         if(v[a[i][j]]==0&&t[a[i][j]]==0)
27         {
28             b.push_back(a[i][j]);
29             dem=1;
30             dfs(a[i][j]);
31             b.pop_back();
32         }
33     }
34 }
35 int main()
36 {
37     freopen("DFSORIEN.inp","r",stdin);
38     freopen("DFSORIEN.out","w",stdout);
39     cin>>n>>m>>l;
40     for(int i=1;i<=r;i++)
41     {
42         int u;
43         cin>>u;
44         t[u]=1;
45     }
46     while(m--)
47     {
48         int u,v;
49         cin>>u>>v;
50         a[u].push_back(v);
51     }
52     b.push_back(1);
53     dfs(1);
54     if(dem==0) cout<<-1;
55     return 0;
56 }
57
C++ source file
```

## Code 2 tham khảo:

```
D:\THANG82020\bailam\duong\DFSORIEN.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

DFSORIEN.cpp
1  #include <bits/stdc++.h>
2  #define task "DFSORIEN"
3  using namespace std;
4  vector <int> ke[3009];
5  bool check[3009];
6  int nb[3009];
7  int u,v,m,n,s;
8  vector <int> ans;
9  void dfs(int u)
10 {
11     check[u]=1;
12     for(int t: ke[u])
13     if (check[t]==0)
14     {
15         ans.push_back(t);
16         nb[t]=u;
17         dfs(t);
18     }
19 }
20 int main()
21 {
22     freopen(task".inp","r",stdin);
23     freopen(task".out","w",stdout);
24     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
25     cin >>n>>m>>s;
26     for(int i=1;i<=m;i++)
27     {
28         cin >>u>>v;
29         ke[u].push_back(v);
30     }
31     dfs(s);
32     // sort (ans.begin(),ans.end());
33     if (ke[s].size()==0) cout <<-1;
34     else
35     for(int t: ans)
36     {
37         vector<int> in;
38         int j=t;
39         while (nb[j]!=s)
40         {
41             in.push_back(j);
42             j=nb[j];
43         }
44         in.push_back(j);
45         in.push_back(s);
46         for(int i: in) cout <<i<<' ';
47         cout <<'\\n';
48     }
49     return 0;
50 }
51

C++ source file length: 998 lines: 51
```

### 17. Đường đi từ S tới T và ngược lại.. DFSOST.\*

Đồ thị có hướng là đồ thị mà tất cả các cạnh trong đồ thị có hướng. Cạnh  $(u, v)$  thể hiện chỉ có đường đi từ đỉnh  $u$  tới đỉnh  $v$ .

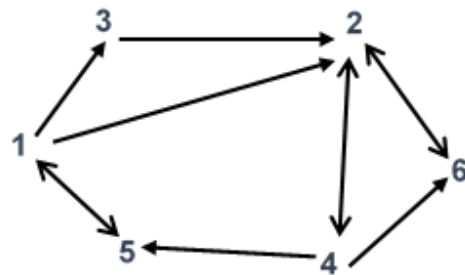
Cho đồ thị có hướng có  $N$  đỉnh,  $M$  cạnh. Jame đang ở đỉnh  $S$ , Alisa đang ở đỉnh  $T$ . Họ hẹn gặp vào cuối tuần ở  $S$  hoặc  $T$ . Hãy cho biết hành trình di chuyển của Jame hoặc Alisa để họ có thể gặp nhau tại  $S$  hoặc  $T$ .

**Dữ liệu:** Vào từ file văn bản DFSOST.INP gồm:

- Dòng 1: Ghi số nguyên dương  $N, M, S, T$  ( $M, N \leq 3000, S, T \leq N, S \neq T$ ).
- $M$  dòng tiếp theo, mỗi dòng ghi số nguyên dương  $u$  và  $v$  thể hiện có đường đi từ đỉnh  $u$  đến đỉnh  $v$  ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSOST.OUT các đường đi xuất phát từ đỉnh  $S$  tới  $T$  và ngược lại. Mỗi đường đi trên một hàng. Mỗi số ghi cách nhau một dấu cách. Nếu Jame và Alisa không có đường đi để gặp nhau tại  $S$  hoặc  $T$  thì ghi ra -1.

DFSOST.INP	DFSOST.OUT
6 11 1 6	6 4 2 1
1 3	6 2 1
1 2	1 5 4 2 6
1 5	(Truy vết ngược)
5 1	
2 6	
6 2	
3 2	
2 4	
4 2	
4 6	
4 5	



#### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng  $a$  có  $N$  hàng,  $N$  cột với  $N$  là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp hướng từ  $u$  tới  $v$ .

$a[u,v] = 1$  khi có đường đi trực tiếp hướng từ  $u$  tới  $v$ .

Lưu ý:  $a[v,u]$  có thể khác  $a[u,v]$ .

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

{

Lưu đường đi tới  $j$  là  $i$  vào mảng  $vt[j] := i$ ;

Nếu  $j = T$  thì truy vết, đưa hành trình từ đỉnh  $j$  về đỉnh  $i$ .

Duyệt ( $j$ ).

}

Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Ở chương trình chính ta gọi  $duyetst(s)$ ; để tìm các đường đi từ  $S$  tới  $T$ .

$duyetts(t)$ ; để tìm các đường đi từ  $T$  về  $S$ .

Khi không có đường đi từ  $S$  tới  $T$  hoặc từ  $T$  tới  $S$  ta ghi ra -1.

Để tổ chức truy vết đơn giản hơn ta viết thành hai thủ tục truy vết.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSOST.pas 1=[+]-
Const
    fi = 'DFSOST.inp';
    fo = 'DFSOST.out';
    nmax = 3000;
Var
    m, n, s, t: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j: longint;
Procedure tracest(u: longint);
Begin
    while u<>s do
    begin
        write(u, #32);
        u:=vt[u];
    end;
    writeln(s);
    vt[s]:=3000;
End;
Procedure tracets(u: longint);
Begin
    while u<>t do
    begin
        write(u, #32);
        u:=vt[u];
    end;
    writeln(t);
    vt[t]:=3000;
End;
Procedure Duyetst(i: longint);
Var j: longint;
Begin
    fre[i]:= False;
    for j:=1 to n do
        if <a[i,j] = 1> and fre[j] then
            begin
                vt[j]:=i;
                if j = t then tracest(j)
                else Duyetst(j);
            end;
    end;
End;
Procedure Duyetts(i: longint);
Var j: longint;
Begin
    fre[i]:= False;
    for j:=1 to n do
        if <a[i,j] = 1> and fre[j] then
            begin
                vt[j]:=i;
                if j = s then tracets(j)
                else Duyetts(j);
            end;
    end;
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s, t);

    for i:= 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
        end;
    fillchar(fre, sizeof(fre), True);
    duyetst(s);
    fillchar(fre, sizeof(fre), True);
    duyetts(t);
    if <vt[s]=0> and <vt[t]=0> then write(-1);
    close(input); close(output);
End.

*== 76:1 ==
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```



## Code 1 tham khảo:

```
D:\THANG82020\baillam\duong\DFSOST.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

DFSOST.cpp
1  #include <bits/stdc++.h>
2  #define task "DFSOST"
3  using namespace std;
4  vector<int> ke[3009];
5  bool check[3009];
6  int u,v,m,n,s,t;
7  vector<int> ans;
8  int road[3009];
9  void dfs(int u,int tt)
10 {
11     road[tt]=u;
12     check[u]=1;
13     if (u==t)
14     {
15         check[t]=0;
16         for (int i=tt;i>=1;i--) cout <<road[i]<<' ';
17         cout <<'\n';
18         return;
19     }
20     for(int t: ke[u])
21     if (check[t]==0)
22     {
23         dfs(t,tt+1);
24         check[t]=0;
25     }
26 }
27 void _dfs(int u,int tt)
28 {
29     road[tt]=u;
30     check[u]=1;
31     if (u==s)
32     {
33         check[t]=0;
34         for (int i=tt;i>=1;i--) cout <<road[i]<<' ';
35         cout <<'\n';
36         return;
37     }
38     for(int t: ke[u])
39     if (check[t]==0)
40     {
41         _dfs(t,tt+1);
42         check[t]=0;
43     }
44 }
45 int main()
46 {
47     freopen(task".inp","r",stdin);
48     freopen(task".out","w",stdout);
49     ios_base::sync_with_stdio(0); cin.tie(NULL); cout.tie(NULL);
50     cin >>n>>m>>s>>t;
51     for(int i=1;i<=m;i++)
52     {
53         cin >>u>>v;
54         ke[u].push_back(v);
55     }
56     dfs(s,1);
57     for (int i=1;i<=n;i++) check[i]=0;
58     _dfs(t,1);
59     return 0;
60 }
```

C++ source file length: 1,145 lines:

## Code 2 tham khảo:

```
D:\THANG82020\bailam\bacl\DFSOST.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSOST.cpp
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  vector<int> p[3005];
6  map<int, bool> checked;
7  int n, maxx, k, m, s;
8  vector<int> res;
9  int dem = 0;
10
11 void input()
12 {
13     cin >> n >> m >> s >> k;
14     for(int i=0; i<m; i++)
15     {
16         int a, b;
17         cin >> a >> b;
18         p[a].push_back(b);
19     }
20 }
21
22 void dfs(int i, int t)
23 {
24     checked[i] = true;
25     res.push_back(i);
26     if(i == t)
27     {
28         dem++;
29         for(int i = res.size() - 1; i >= 0; i--)
30         {
31             cout << res[i] << ' ';
32         }
33         cout << '\n';
34     }
35     for(int j = 0; j < p[i].size(); j++)
36         if(checked[p[i][j]] == false)
37             dfs(p[i][j], t);
38     checked[i] = false;
39     res.pop_back();
40 }
41
42 void solve()
43 {
44     dfs(s, k);
45     dfs(k, s);
46     if(dem == 0)
47         cout << -1;
48 }
49
50 int main()
51 {
52     freopen("DFSOST.inp", "r", stdin);
53     freopen("DFSOST.out", "w", stdout);
54     input();
55     solve();
56     return 0;
57 }
58
C++ source file | lengt
```

## 18. Tìm điểm hẹn.. DFSOST2.\*

Đồ thị có hướng là đồ thị mà tất cả các cạnh trong đồ thị có hướng. Cạnh  $(u, v)$  thể hiện chỉ có đường đi từ đỉnh  $u$  tới đỉnh  $v$ .

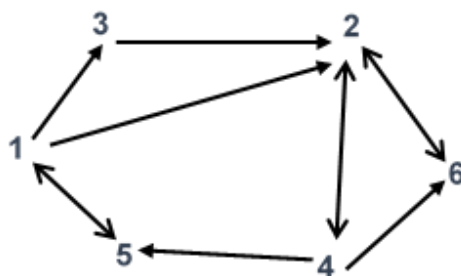
Cho đồ thị có hướng có  $N$  đỉnh,  $M$  cạnh. Jame đang ở đỉnh  $S$ , Alisa đang ở đỉnh  $T$ . Họ hẹn gặp vào cuối tuần ở đâu đó trong đồ thị. Hãy tìm các đỉnh mà Jame và Alisa có thể di chuyển tới để họ có thể gặp nhau. Tất nhiên, họ có thể gặp nhau ở tại  $S$  hoặc  $T$ .

**Dữ liệu:** Vào từ file văn bản DFSOST2.INP gồm:

- Dòng 1: Ghi số nguyên dương  $N, M, S, T$  ( $M, N \leq 3000, S, T \leq N, S \neq T$ ).
- $M$  dòng tiếp theo, mỗi dòng ghi số nguyên dương  $u$  và  $v$  thể hiện có đường đi từ đỉnh  $u$  đến đỉnh  $v$  ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSOST2.OUT các điểm hẹn tìm được. Các đỉnh được sắp xếp thành dãy tăng. Mỗi số ghi cách nhau một dấu cách. Nếu Jame và Alisa không có đường đi để gặp nhau thì ghi ra -1.

DFSOST2.INP	DFSOST2.OUT
6 11 1 6	1 2 4 6
1 3	
1 2	
1 5	
5 1	
2 6	
6 2	
3 2	
2 4	
4 2	
4 6	
4 5	



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng  $a$  có  $N$  hàng,  $N$  cột với  $N$  là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp hướng từ  $u$  tới  $v$ .

$a[u,v] = 1$  khi có đường đi trực tiếp hướng từ  $u$  tới  $v$ .

Lưu ý:  $a[v,u]$  có thể khác  $a[u,v]$ .

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

```
{  
    Lưu đường đi tới  $j$  là  $i$  vào mảng  $vt[j] := i$ ;  
    Nếu  $j = T$  thì truy vết, đưa hành trình từ đỉnh  $j$  về đỉnh  $i$ .  
    Duyệt (  $j$  ).  
}
```

Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Ở chương trình chính ta gọi  $duyetst(s)$ ; để tìm các đường đi từ  $S$  tới  $T$ .

$duyetts(t)$ ; để tìm các đường đi từ  $T$  về  $S$ .

Để tổ chức truy vết đơn giản hơn ta viết thành hai thủ tục truy vết. Mỗi lần tìm được một đường đi, ta đánh dấu các đỉnh trên đường đi đó.

Các đỉnh trên đường từ  $S$  tới  $T$  đánh dấu vào  $ck1[i]$ .

Các đỉnh trên đường từ  $T$  tới  $S$  đánh dấu vào  $ck2[i]$ .

Duyệt từ 1 tới  $N$ , nếu đỉnh nào được đánh dấu trên  $ck1$  và  $ck2$  thì đưa đỉnh đó ra. Nếu không có đỉnh nào được đánh dấu, tức là không có đường đi từ  $S$  tới  $T$  hoặc từ  $T$  tới  $S$  ta ghi ra -1.

Độ phức tạp thuật toán:  $O(M + N)$

```

Free Pascal IDE
DFSOST2.pas
1=1

Const
    fi = 'DFSOST2.inp';
    fo = 'DFSOST2.out';
    nmax = 3000;

Var
    m, n, s, t: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre, ck1, ck2: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j, p: longint;

Procedure tracest(u: longint);
Begin
    while u <> s do
    begin
        ck1[u] := true; //write(u, #32);
        u := vt[u];
    end;
    vt[s] := 3000;
End;

Procedure tracets(u: longint);
Begin
    while u <> t do
    begin
        ck2[u] := true; //write(u, #32);
        u := vt[u];
    end;
    vt[t] := 3000;
End;

Procedure Duyetst(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
            begin
                vt[j] := i;
                if j = t then tracest(j)
                else Duyetst(j);
            end;
    end;
End;

Procedure Duyetts(i: longint);
Var j: longint;
Begin
    fre[i] := False;
    for j := 1 to n do
        if (a[i, j] = 1) and fre[j] then
            begin
                vt[j] := i;
                if j = s then tracets(j)
                else Duyetts(j);
            end;
    end;
End;

Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s, t);
    for i := 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
        end;
    fillchar(ck1, sizeof(ck1), false);
    fillchar(ck2, sizeof(ck2), false);
    ck1[s] := true;
    ck2[t] := true;
    fillchar(fre, sizeof(fre), True);
    duyetst(s);
    duyetts(t);
    p := 0;
    for i := 1 to N do
        if ck1[i] and ck2[i] then
            begin
                write(i, #32);
                p := i;
            end;
    if p = 0 then write(-1);
    close(input); close(output);
End.

```

Code 1 tham khảo:

```
*D:\THANG82020\bailam\bac\DFSOST2.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSOST2.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3  vector <int> p[3005];
4  map <int, bool> checked;
5  int n, maxx, k, m, s;
6  vector <int> res;
7  int dem = 0;
8  void input()
9  {
10     cin >> n >> m >> s >> k;
11     for(int i=0; i<m; i++)
12     {
13         int a, b;
14         cin >> a >> b;
15         p[a].push_back(b);
16     }
17 }
18 bool ok;
19 void dfs(int i, int t)
20 {
21     checked[i] = true;
22     if(i == t)
23         ok = 1;
24     for(int j = 0; j < p[i].size(); j++)
25         if(checked[p[i][j]] == false)
26             dfs(p[i][j], t);
27     checked[i] = false;
28 }
```

```

33 void solve()
34 {
35     for(int i = 1; i <= n; i++)
36     {
37         ok = 0;
38         dfs(s, i);
39         if(ok)
40         {
41             ok = 0;
42             dfs(k, i);
43             if(ok)
44             {
45                 dem++;
46                 cout << i << ' ';
47             }
48         }
49     }
50     if(dem == 0)
51         cout << -1;
52 }
53
54 int main()
55 {
56     freopen("DFSOST2.inp", "r", stdin);
57     freopen("DFSOST2.out", "w", stdout);
58     input();
59     solve();
60     return 0;
61 }
62

```

C++ source file

Code 2 tham khảo:

```
D:\THANG82020\bailam\loc\DFSOST2.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSOST2.cpp
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 int a[3001][3001];
5
6 int n, m, Free[3001], f[3001], u, v, s, k, ma=INT_MIN, check=0, p=0;
7 vector <int> g;
8 void DFS(int u)
9 {
10     if (f[u]==0)
11         f[u]=1;
12     else
13     {
14         check=1;
15         g.push_back(u);
16     }
17     Free[u] = false;
18     for (int v=1;v<=n;v++)
19         if (a[u][v] == 1 && Free[v])
20             DFS(v);
21 }
22
23 int main()
24 {
25     freopen("DFSOST2.INP", "r", stdin);
26     freopen("DFSOST2.OUT", "w", stdout);
27     ios_base::sync_with_stdio(false);
28     cin.tie(NULL);
29     cout.tie(NULL);
30     cin >> n >> m >> s >> k;
31
32     for (int i = 1; i <= n; i++)
33         Free[i] = 1;
34     for (int i = 1; i <= n; i++)
35         for (int j = 1; j <= n; j++)
36             a[i][j] = 0;
37     for (int i = 1; i <= m; i++)
38     {
39         cin >> u >> v;
40         a[u][v]=1;
41     }
42     DFS(s);
43     for (int i=1;i<=n;i++)
44         Free[i]=1;
45     DFS(k);
46     sort (g.begin(),g.end());
47     for (auto i:g)
48         cout << i << " ";
49     if (check==0)
50     {
51         cout << -1;
52         return 0;
53     }
54     return 0;
55 }
56
C++ source file length : 1,105 lines : 56
```



## 19. Đường truyền an toàn.. DFSSAFE.\*

Đồ thị có hướng là đồ thị mà tất cả các cạnh trong đồ thị có hướng. Cạnh  $(u, v)$  thể hiện chỉ có đường đi từ đỉnh  $u$  tới đỉnh  $v$ .

Cho đồ thị có hướng có  $N$  đỉnh,  $M$  cạnh. Jame đang ở đỉnh  $S$ , muốn truyền tin đến đỉnh  $T$ . Đường truyền được gọi an toàn khi có ít nhất hai đường truyền khác nhau từ đỉnh  $S$  đến đỉnh  $T$ . Hai đường truyền được gọi là khác nhau khi đi có ít nhất một đỉnh trên đường đi khác nhau.

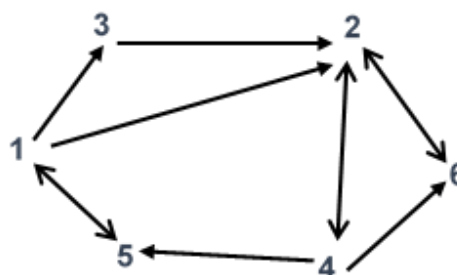
Hãy cho biết đường truyền từ  $S$  tới  $T$  có an toàn hay không. Nếu có ghi ra YES và số lượng đường truyền từ  $S$  tới  $T$ , ngược lại ghi ra NO.

**Dữ liệu:** Vào từ file văn bản DFSSAFE.INP gồm:

- Dòng 1: Ghi số nguyên dương  $N, M, S, T$  ( $M, N \leq 3000, S, T \leq N, S \neq T$ ).
- $M$  dòng tiếp theo, mỗi dòng ghi số nguyên dương  $u$  và  $v$  thể hiện có đường đi từ đỉnh  $u$  đến đỉnh  $v$  ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSSAFE.OUT nếu đường truyền từ  $S$  tới  $T$  là an toàn thì ghi ra YES và số lượng đường truyền từ  $S$  tới  $T$ , ngược lại ghi ra NO.

DFSSAFE.INP	DFSSAFE.OUT
6 11 1 6 1 3 1 2 1 5 5 1 2 6 6 2 3 2 2 4 4 2 4 6 4 5	YES 2
6 11 2 5 1 3 1 2 1 5 5 1 2 6 6 2 3 2 2 4 4 2 4 6 4 5	NO



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng  $a$  có  $N$  hàng,  $N$  cột với  $N$  là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp hướng từ  $u$  tới  $v$ .

$a[u,v] = 1$  khi có đường đi trực tiếp hướng từ  $u$  tới  $v$ .

*Lưu ý:  $a[v,u]$  có thể khác  $a[u,v]$ .*

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

{

Lưu đường đi tới  $j$  là  $i$  vào mảng  $vt[j] := i$ ;

Nếu  $j = T$  thì tăng biến đếm lên.

Duyệt ( $j$ ).

}

Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Ở chương trình chính ta gọi  $duyet(S)$ ; để tìm các đường đi từ  $S$  tới  $T$ .

Nếu số lượng đường đi từ  $S$  tới  $T$  là đếm  $\geq 2$  thì đưa ra YES và đếm,

Ngược lại ghi ra NO.

Độ phức tạp thuật toán:  $O(M + N)$

Code chương trình tham khảo:

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
DFSSAFE.pas 1=[+]-
Const
    fi = 'DFSSAFE.inp';
    fo = 'DFSSAFE.out';
    nmax = 3000;
Var
    m, n, s, t: longint;
    a: array[1..nmax, 1..nmax] of longint;
    fre: array[1..nmax] of boolean;
    vt: array[1..nmax] of longint;
    u, v, i, j: longint;
    res: longint = 0;
Procedure Duet<i: longint>;
Var j: longint;
Begin
    fre[i] := False;
    for j:=1 to n do
        if (a[i,j] = 1) and fre[j] then
            begin
                if j = t then inc(res)
                else Duet(j);
            end;
    end;
End;
Begin
    assign(input, fi); reset(input);
    assign(output, fo); rewrite(output);
    readln(n, m, s, t);
    for i:= 1 to m do
        begin
            readln(u, v);
            a[u, v] := 1;
        end;
    fillchar(fre, sizeof(fre), True);
    duet(s);
    if res >= 2 then write('YES', #32, res)
    else write('NO');
    close(input); close(output);
End.

```

Code 1 tham khảo:

```

D:\THANG82020\bailam\loc\DFSSAFE.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSSAFE.cpp
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int a[3001][3001];
5
6  int n, m, Free[3001], f[3001], u, v, s, k, ma=INT_MIN, check=0, p=0;
7  vector<int> g[3001];
8  void DFS(int u, int t, int cnt)
9  {
10     f[cnt]=u;
11     if (u==t)
12     {
13         p++;
14         check=1;
15     }
16     Free[u] = false;
17     for (auto i:g[u])
18         if (i != u && Free[i])
19             DFS(i, t, cnt+1);
20     Free[u]=1;
21 }
22
23 int main()
24 {
25     freopen("DFSSAFE.INP", "r", stdin);
26     freopen("DFSSAFE.OUT", "w", stdout);
27     ios_base::sync_with_stdio(false);
28     cin.tie(NULL);
29     cout.tie(NULL);
30     cin >> n >> m >> s >> k;

```

C++ source file length : 1,059 lines : 54

```

31     for (int i = 1; i <= n; i++)
32         Free[i] = 1;
33     for (int i = 1; i <= n; i++)
34         for (int j = 1; j <= n; j++)
35             a[i][j] = 0;
36     for (int i = 1; i <= m; i++)
37     {
38         cin >> u >> v;
39         g[u].push_back(v);
40     }
41     DFS(s,k,1);
42     if (p>=2)
43         cout << "YES " << p;
44     else
45         cout << "NO";
46     //     if (check==0)
47     //     {
48     //         cout << -1;
49     //         return 0;
50     //     }
51     return 0;
52 }
53
54

```

C++ source file      length : 1,059    lines : 54    Ln :

Code 2 tham khảo:

```

1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSSAFE"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m,s,kt = 0,t;
23 ll res = 0;
24 Vi g[nmax];
25 Vi gg;
26 int a[nmax];
27 int check[nmax];
28 int visit[nmax];
29
30

```

C++ source file      length :

```

31 void dfs(int u, int par, int t, int cnt){
32     a[cnt] = u;
33     if(u == t){
34         res++;
35     }
36     check[u] = 1;
37     for(auto xx : g[u]){
38         if(xx != par && check[xx] == 0) dfs(xx,u,t,cnt + 1);
39     }
40     check[u] = 0;
41 }
42
43
44 void kurumi(){
45     cin >> n >> m >> s >> t;
46
47     for(int i = 1; i <= m; i++){
48         int u,v;
49         cin >> u >> v;
50         g[u].pb(v);
51     }
52     dfs(s,0,t,1);
53     if(res >= 2){
54         cout << "YES" << ' ' << res << '\n';
55     }
56     else cout << "NO";
57 }
58
59 int main()
60 {
61     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
62     freopen(Kurumi".inp","r",stdin);
63     freopen(Kurumi".out","w",stdout);
64     kurumi();
65     return 0;
66 }
67

```

C++ source file

length : 1,403 lines : 67

## 20. Điểm đến an toàn.. DFSSAFE2.\*

Đồ thị có hướng là đồ thị mà tất cả các cạnh trong đồ thị có hướng. Cạnh  $(u, v)$  thể hiện chỉ có đường đi từ đỉnh  $u$  tới đỉnh  $v$ .

Cho đồ thị có hướng có  $N$  đỉnh,  $M$  cạnh. Jame đang ở đỉnh  $S$ , muốn truyền tin đến đỉnh  $T$  nào đó trong đồ thị. Đường truyền được gọi an toàn khi có ít nhất hai đường truyền khác nhau từ đỉnh  $S$  đến đỉnh  $T$ . Hai đường truyền được gọi là khác nhau khi đi có ít nhất một đỉnh trên đường đi khác nhau.

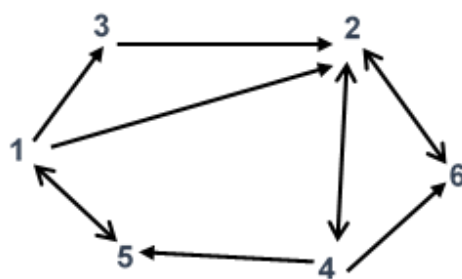
Hãy đưa ra các điểm đến an toàn.

**Dữ liệu:** Vào từ file văn bản DFSSAFE2.INP gồm:

- Dòng 1: Ghi số nguyên dương  $N, M, S$  ( $M, N \leq 3000, S \leq N$ ).
- $M$  dòng tiếp theo, mỗi dòng ghi số nguyên dương  $u$  và  $v$  thể hiện có đường đi từ đỉnh  $u$  đến đỉnh  $v$  ( $u, v \leq N$ ).

**Kết quả:** Ghi ra file văn bản DFSSAFE2.OUT các điểm đến an toàn. Mỗi số cách nhau một dấu cách. Nếu không có điểm đến an toàn đưa ra 0.

DFSSAFE2.INP	DFSSAFE2.OUT
6 11 1 1 3 1 2 1 5 5 1 2 6 6 2 3 2 2 4 4 2 4 6 4 5	2 5 6
6 11 6 1 3 1 2 1 5 5 1 2 6 6 2 3 2 2 4 4 2 4 6 4 5	0



### Thuật toán:

\* Ta biểu diễn đồ thị theo ma trận kề: Lưu các cạnh vào mảng  $a$  có  $N$  hàng,  $N$  cột với  $N$  là số đỉnh của đồ thị. Khi đó,

$a[u,v] = 0$  khi không đường đi trực tiếp hướng từ  $u$  tới  $v$ .

$a[u,v] = 1$  khi có đường đi trực tiếp hướng từ  $u$  tới  $v$ .

Lưu ý:  $a[v,u]$  có thể khác  $a[u,v]$ .

\* Tư tưởng giải thuật Duyệt theo chiều sâu:

Ban đầu các đỉnh  $fre[i] = \text{True}$ .

Duyệt đến đỉnh thứ  $i$ , đánh dấu  $i$  đã đi đến ( $fre[i] = \text{False}$ ). Ta thực hiện duyệt tất cả các đỉnh  $j$  đang free và có đường đi trực tiếp từ  $i$  đến  $j$  thì ta:

```
{  
    Lưu đường đi tới  $j$  là  $i$  vào mảng  $vt[j] := i$ ;  
    Nếu  $j = T$  thì tăng biến đếm lên.  
    Duyệt ( $j$ ).  
}
```

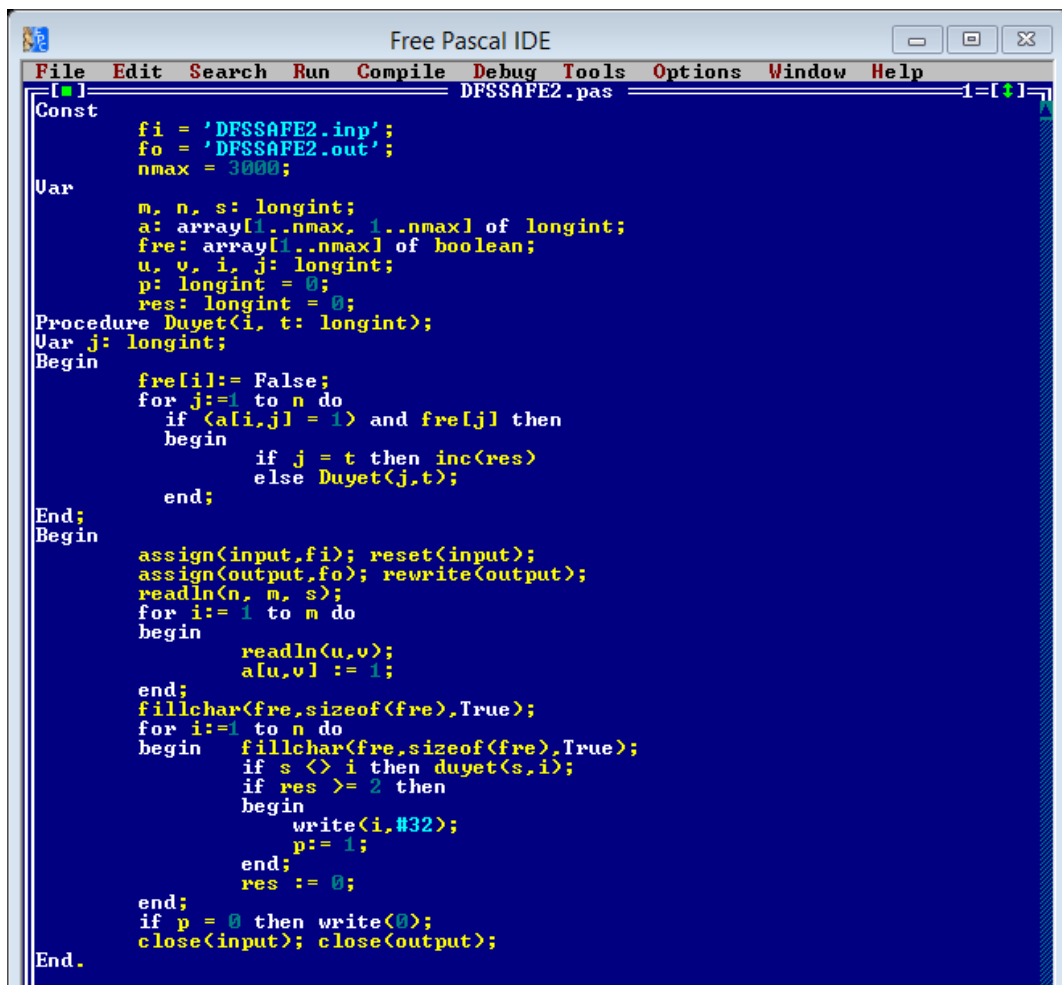
Tới đỉnh  $j$ , ta lặp lại việc duyệt như ở bước trên.

Ở chương trình chính ta:

Duyệt qua tất cả các đỉnh  $t$ , ta thực hiện đếm số đường đi từ  $S$  tới  $T$ , nếu đếm  $\geq 2$  thì đưa  $t$  ra.

Nếu không có đỉnh  $t$  nào thỏa mãn, ghi ra 0.

Độ phức tạp thuật toán:  $O(M + N)$



```
Free Pascal IDE  
File Edit Search Run Compile Debug Tools Options Window Help  
DFSSAFE2.pas  
Const  
    fi = 'DFSSAFE2.inp';  
    fo = 'DFSSAFE2.out';  
    nmax = 3000;  
Var  
    n, m, s: longint;  
    a: array[1..nmax, 1..nmax] of longint;  
    fre: array[1..nmax] of boolean;  
    u, v, i, j: longint;  
    p: longint = 0;  
    res: longint = 0;  
Procedure Duyet(i, t: longint);  
Var j: longint;  
Begin  
    fre[i] := False;  
    for j := 1 to n do  
        if (a[i, j] = 1) and fre[j] then  
            begin  
                if j = t then inc(res)  
                else Duyet(j, t);  
            end;  
    end;  
End;  
Begin  
    assign(input, fi); reset(input);  
    assign(output, fo); rewrite(output);  
    readln(n, m, s);  
    for i := 1 to m do  
        begin  
            readln(u, v);  
            a[u, v] := 1;  
        end;  
    fillchar.fre, sizeof.fre, True);  
    for i := 1 to n do  
        begin  
            fillchar.fre, sizeof.fre, True);  
            if s <> i then duyet(s, i);  
            if res >= 2 then  
                begin  
                    write(i, #32);  
                    p := 1;  
                end;  
            res := 0;  
        end;  
    if p = 0 then write(0);  
    close(input); close(output);  
End.
```

## Code 1 tham khảo:

```
D:\THANG82020\bailam\hai\DFSSAFE2.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DFSSAFE2.cpp
1  ///Kurumi
2  #include <bits/stdc++.h>
3  #define ll long long
4  #define ull unsigned long long
5  #define Pii pair < int, int >
6  #define Piii pair < int, Pii >
7  #define Vi vector < int >
8  #define Vii vector < Pii >
9  #define pb push_back
10 #define mp make_pair
11 #define reset(x) memset(x, 0, sizeof(x))
12 #define fori(i, a, b) for(int i = a; i <= b; i++)
13 #define ford(i, a, b) for(int i = a; i >= b; i--)
14 #define fora(xx, gg) for(auto xx : gg)
15 #define fi first
16 #define se second
17 #define Kurumi "DFSSAFE2"
18 #define nmax 50005
19 const ll OO = 1e15+7 ;
20 const ll mod = 1e9+7;
21 using namespace std;
22 int n,m,s,kt = 0;
23 ll res = 0;
24 Vi g[nmax];
25 Vi gg;
26 int a[nmax];
27 int check[nmax];
28 int visit[nmax];
29 void dfs(int u, int par, int t, int cnt){
30     a[cnt] = u;
31     if(u == t){
32         res++;
33         if(res >= 2){
34             kt = 1;
35             cout << t << ' ';
36             res = -OO;
37         }
38     }
39     check[u] = 1;
40     for(auto xx : g[u]){
41         if(xx != par && check[xx] == 0) dfs(xx,u,t,cnt + 1);
42     }
43     check[u] = 0;
44 }
45
46 void kurumi(){
47     cin >> n >> m >> s;
48
49     for(int i = 1; i <= m; i++){
50         int u,v;
51         cin >> u >> v;
52         g[u].pb(v);
53     }
54
55     for(int i = 1; i <= n; i++){
56         if(i != s){
57             res = 0;
58         }
59     }
60 }
C++ source file length : 1,567 lines :
```



```

59         reset(check);
60         dfs(s,0,i,1);
61     }
62 }
63 if(kt == 0) cout << 0;
64 }
65
66 int main()
67 {
68     ios_base::sync_with_stdio(false);cin.tie(NULL);cout.tie(NULL);
69     freopen(Kurumi".inp","r",stdin);
70     freopen(Kurumi".out","w",stdout);
71     kurumi();
72     return 0;
73 }
74

```

C++ source file length : 1,567 lines : 74

Code 2 tham khảo:

```

1  #include <bits/stdc++.h>
2  #define ft first
3  #define sc second
4  using namespace std;
5  int n,m,l,r;
6  int v[3009];
7  deque<int> b;
8  int dem=0;
9  int ct=0;
10 int ck=0;
11 int t[3009];
12 vector<int>a[3009];
13 void dfs(int i,int r)
14 {
15     v[i]=1;
16     if(b.back()==r)
17     {
18         ct++;
19     }
20     for(int j=0; j<a[i].size(); j++)
21     {
22         if(v[a[i][j]]==0)
23         {
24             b.push_back(a[i][j]);
25             dfs(a[i][j],r);
26             b.pop_back();
27         }
28     }
29     v[i]=0;
30 }

```

C++ source file

```

31 }
32 int main()
33 {
34     freopen("DFSSAFE2.inp", "r", stdin);
35     freopen("DFSSAFE2.out", "w", stdout);
36     cin >> n >> m >> l;
37     while (m--)
38     {
39         int u, v;
40         cin >> u >> v;
41         a[u].push_back(v);
42     }
43     for (int j = 1; j <= n; j++)
44     {
45         ct = 0;
46         if (j != 1)
47         {
48             b.push_back(1);
49             dfs(1, j);
50             if (ct >= 2) {
51                 cout << j << " ";
52                 ck = 1;
53             }
54         }
55     }
56     if (ck == 0) cout << 0;
57     return 0;
58 }
59
60

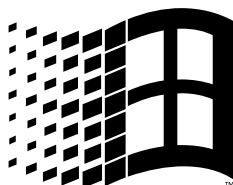
```

C++ source file

len

---Hết---

"Mỗi ngày tôi chọn một niềm vui,  
Chọn những bông hoa, chọn những nụ cười..."



Các bạn tìm đọc tài liệu của cùng tác giả:



Hỗ trợ qua zalo: 0854518333