信奥算法

有向无环图

Directed Acyclic Graph

DAG

先后
依赖
结构

topological sort

DAG的拓扑排序

1 Kahn算法

2 DFS实现

解锁关卡

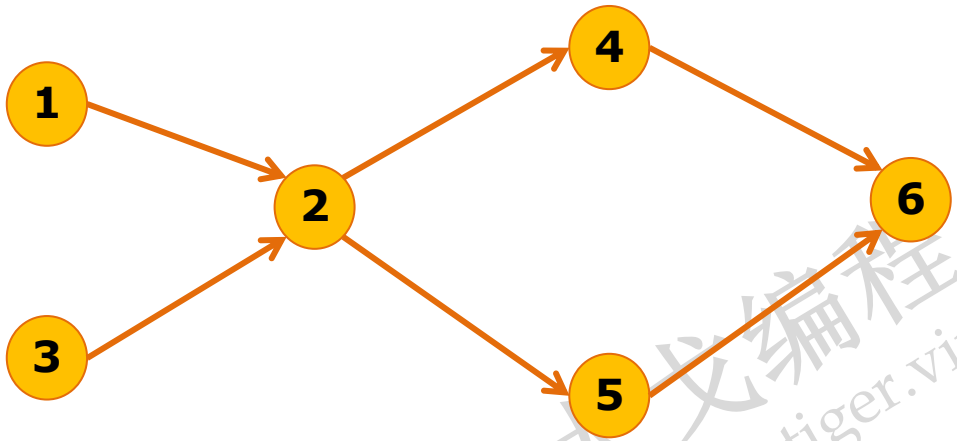
你是一个电子游戏高手，正在研究一款新的游戏。该游戏共有n种关卡有待解锁，编号1到n。关卡之间有m条依赖关系，第i条为：解锁关卡ai前必须先解锁关卡bi。请你为n个关卡设计一个可行的解锁顺序，若有多个解请输出字典序最小解。本题保证有解。

输入正整数n和m，n<=1000,m<=10000。接着共m行每行两个不同的数ai和bi，1<=ai,bi<=n。输出一个正整数。

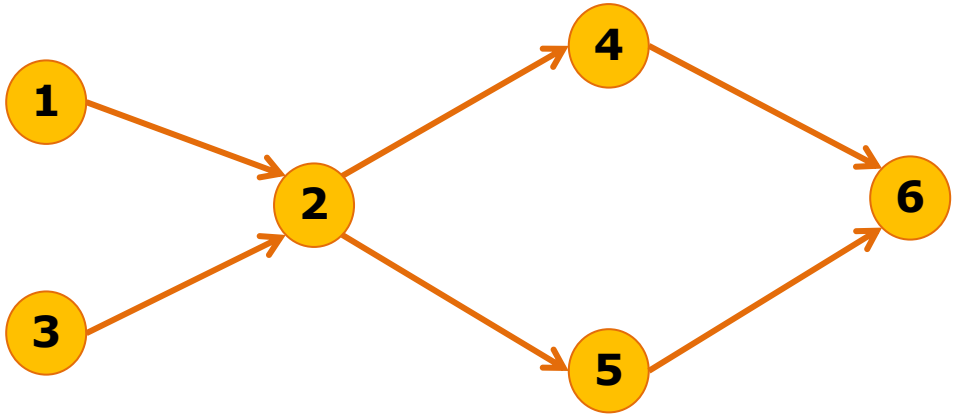
输入样例
6 6
4 2
5 2
6 4
6 5
2 3
2 1

输出样例
1 3 2 4 5 6

图的识别
点的含义
边的含义
DAG?



拓扑排序：Kahn算法



贪心策略	每次找入度为0的节点u，优先靠前安排u， 删除节点u，以及u出发的所有边
如何求出字典序最小的排序序列	多个入度为零的点里 选字典序最小的节点

```
 8      cin>>n>>m;
 9      for(int i=1;i<=m;i++){
10          int a,b;
11          cin>>a>>b;
12          if(d[b][a])continue;
13          d[b][a]=1;
14          in[a]++;
15      }
16      for(int k=1,i;k<=n;k++){
17          for(i=1;i<=n;i++)
18              if(!vst[i]&&in[i]==0)break;
19          topo[++cnt]=i;
20          vst[i]=cnt;
21          for(int j=1;j<=n;j++)
22              if(d[i][j]) d[i][j]=0,in[j]--;
23      }
24      for(int i=1;i<=n;i++)cout<<topo[i]<<" ";
```

d[b][a]代表b要在a之前完成
解锁a依赖于解锁b

易错点：重边处理

in[a]代表有a的入度
即a当前依赖几个前序关卡

vst[i]代表关卡i是否已解锁

cnt记录已解锁几个关卡

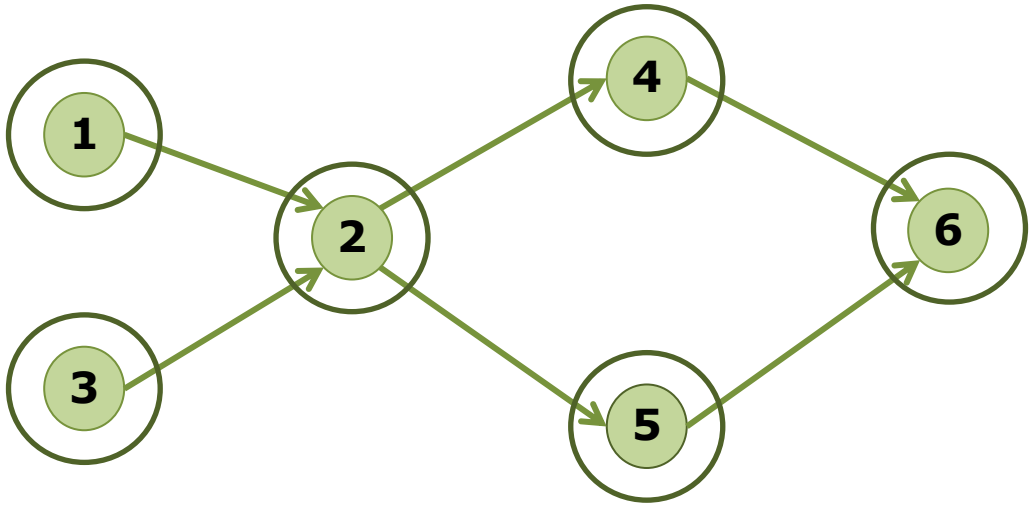
topo[x]是第x个解锁哪一关

时空复杂度	$O(N^2)$

DAG的拓扑排序

1	Kahn算法
2	DFS实现

拓扑排序：DFS实现



最先完成DFS的节点
在拓扑顺序里最靠后

每次找当前
出度为0节点

DFS只保证拓扑顺序正确，无法保证字典序最小

```cpp
vector<int> to[N];
int n,m,topo[N],cnt;
bool vst[N];
void dfs(int x){
    vst[x]=1;
    for(int i=0;i<to[x].size();i++)
        if(!vst[to[x][i]])
            dfs(to[x][i]);
    topo[n-cnt]=x; cnt++;
}
int main(){
    cin>>n>>m;
    for(int i=0;i<m;i++){
        int u,v;
        cin>>u>>v;
        to[u].push_back(v);
    }
    for(int i=1;i<=n;i++)if(!vst[i])dfs(i);
    for(int i=1;i<=n;i++)cout<<topo[i]<<" ";
    return 0;
}
```

从x开始继续深度优先遍历

vst[i]代表关卡i是否已访问

查看依赖于i的所有关卡

若未访问则继续深入

x及x后续均完成DFS时
放入拓扑序列最后位置

cnt记录已有几个关卡
确定了排序位置

u到v有条有向边

时空复杂度	O(N+M)

DAG

偏序关系

边的方向隐藏某种内在的顺序

先后的传递性

工作依赖关系

时间先后关系

高度高低关系

尺寸嵌套关系

编号大小关系

能力碾压关系

拓扑排序：对n个节点先后排列
在一维实数轴上排布n个节点
所有的边方向一致

记忆化搜索

DFS

DP

DAG

记忆化搜索

Depth First Search

Dynamic Programming

Directed Acyclic Graph

最长雪道

在n*m格子的雪山地图里，每格都有高度值。地图是四向连通的，请你找一条从高到低的下降雪道，使得长度最长。该雪道的起点和终点也由你来选择。1<=n,m<=100

输入样例
```
5 5
1 2 3 4 5
16 17 18 19 6
15 24 25 20 7
14 23 22 21 8
13 12 11 10 9
```
输出样例
```
25
```

输入样例
```
3 3
3 9 2
5 8 10
4 12 11
```
输出样例
```
6
```

此题是DAG吗?

节点代表什么?

有向边代表什么?

算法：最长雪道

算法1

拓扑排序：格子从低到高

依次DP计算从每格开始的最长雪道

算法2

记忆化搜索

搜索顺序中隐含着高低顺序

算法1：排序+DP

从低到高依次填表

3	9	2
5	8	10
4	12	11

h[i][j]表示
(i,j)格的高度

1	4	1
2	3	4
1	6	5

f[i][j]表示
从(i,j)格开始下降
的最长雪道长度

算法2：记忆化搜索

从任意格(i,j)	开始DFS	直到确定f[i][j]

3	9	2
5	8	10
4	12	11

1	4	1
2	3	4
1	6	5

h[i][j]表示
(i,j)格的高度

f[i][j]表示
从(i,j)格开始下降
的最长雪道长度

算法2：记忆化搜索

h[i][j]表示 (i,j)格的高度
f[i][j]表示 从(i,j)格开始下降 的最长雪道长度

ok[i][j]表示 f[i][j]是否计算完
F(i,j)函数返回值 为(i,j)格开始下降 的最长雪道长度

```
25      for(ll i=1;i<=n;i++)
26          for(ll j=1;j<=m;j++)
27              ans=max(ans,F(i,j));
28      cout<<ans<<endl;
```

算法2：记忆化搜索

h[i][j]表示 (i,j)格的高度	ok[i][j]表示 f[i][j]是否计算完
f[i][j]表示 从(i,j)格开始下降 的最长雪道长度	F(i,j)函数返回值 为(i,j)格开始下降 的最长雪道长度

```
9  ll F(ll x,ll y){
10     if(ok[x][y])return f[x][y];
11     ok[x][y]=1; f[x][y]=1;
12     for(ll k=0;k<4;k++){
13         ll nx=x+dx[k],ny=y+dy[k];
14         if(nx>=1&&nx<=n&&ny>=1&&ny<=m&&h[nx][ny]<h[x][y])
15             f[x][y]=max(f[x][y],F(nx,ny)+1);
16     }
17     return f[x][y];
18 }
```

算法2：记忆化搜索

h[i][j]表示 (i,j)格的高度
f[i][j]表示 从(i,j)格开始下降 的最长雪道长度

ok[i][j]表示 f[i][j]是否计算完
F(i,j)函数返回值 为(i,j)格开始下降 的最长雪道长度

```
9  ll F(ll x,ll y){
10     若(x,y)状态已算过    返回记忆中的结果
11     设置(x,y)状态已算过  准备计算(x,y)状态
12     for(ll k=0;k<4;k++){
13         查看/转移到所有相关状态
14         比较出最优解
15             f[x][y]=max(f[x][y],F(nx,ny)+1);
16     }
17     返回结果
18 }
```

算法对比

算法1

拓扑排序+DP

若拓扑排序用DFS实现
解法步骤就和算法2类似

算法2

DFS记忆化搜索

每个节点最多计算1次

时间复杂度

O(DAG节点数+边数)=O(n*m)

算法对比

算法1

拓扑排序：格子从低到高

依次DP：算以每格开始的最长雪道

必需预先理清状态间整体的先后依赖关系

算法2

记忆化搜索
搜索顺序中隐含着高低顺序

无需预先理清状态间整体的先后

DAG各类问题

DAG最长路径

DAG最短路径

DAG最优路径

DAG路径计数

两种算法

拓扑顺序 + DP

DFS + 记忆化

强力推荐

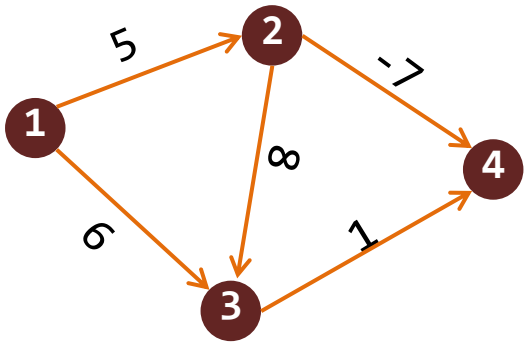
有权图的储存

邻接矩阵
adjacency matrix

邻接表
adjacency list

能节约空间和时间

邻接矩阵



	1	2	3	4
1	∞	5	6	∞
2	∞	∞	8	-7
3	∞	∞	∞	1
4	∞	∞	∞	∞

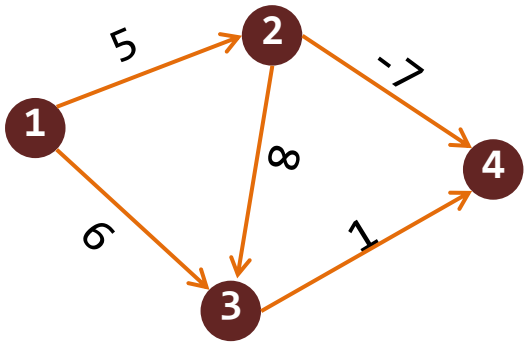
邻接表

1	2	3		
2	3	4		
3	4			
4				

1	5	6		
2	8	-7		
3	1			
4				

能节约空间和时间

邻接表



1号节点出边能到2,3号节点

2号节点出边能到3,4号节点

3号节点出边能到4号节点

4号节点没有出边

1	2	3		
2	3	4		
3	4			
4				

1号节点出边的边长依次是5,6

2号节点出边的边长依次是8,-7

3号节点出边的边长依次是1

4号节点没有出边

1	5	6		
2	8	-7		
3	1			
4				

```cpp
#include<iostream>
#include<vector>
#define N 109
using namespace std;
vector<int> to[N],w[N];
```

输入样例

```
4 7
1 2 2
2 1 2
2 3 1
3 2 1
1 3 4
3 4 1
2 4 -4
```

输入储存

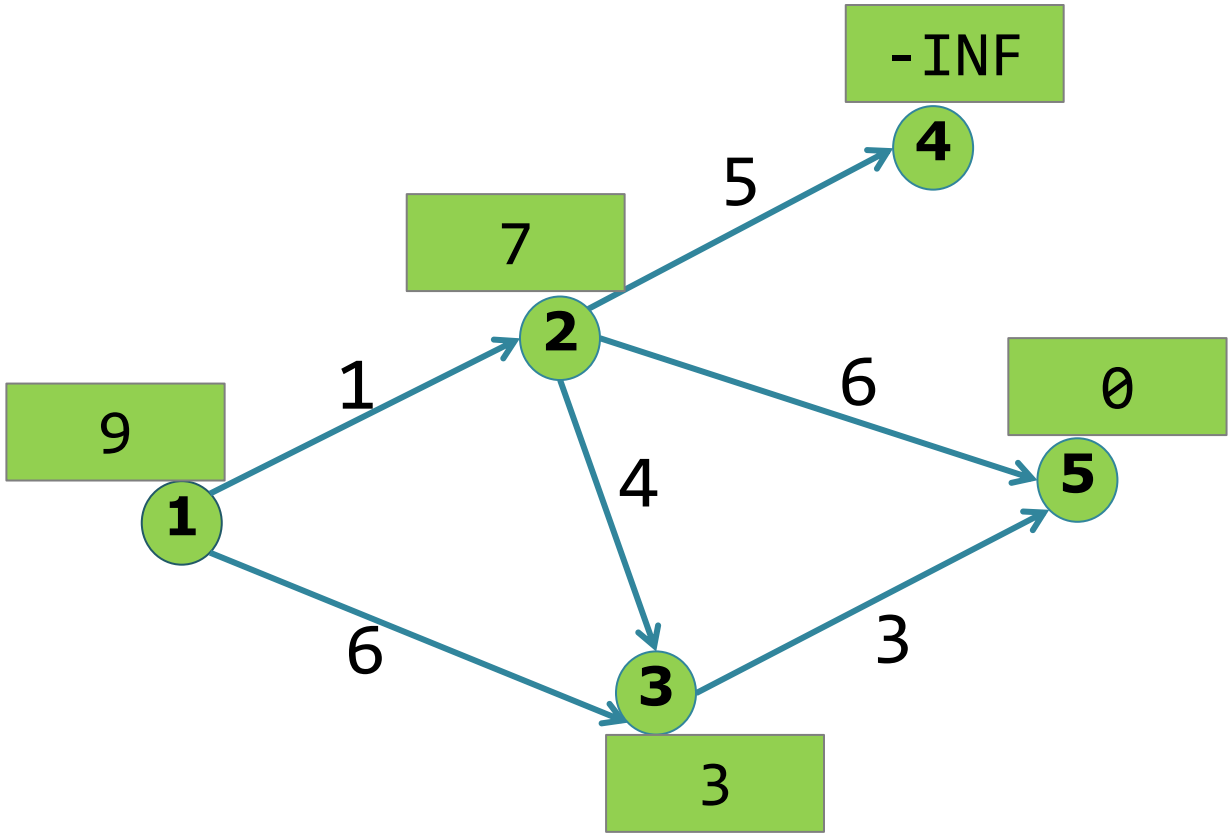
```cpp
    for(int i=0;i<m;i++){
        int a,b,c;
        cin>>a>>b>>c;
        to[a].push_back(b); w[a].push_back(c);
    }
```

输出遍历

```cpp
    for(int i=1;i<=n;i++)
        for(int k=0;k<to[i].size();k++)
            cout<<i<<" "<<to[i][k]<<" "<<w[i][k]<<endl;
```

现场挑战
158

n=5



u	f[u]
1	9
2	7
3	3
4	-INF
5	0

F(u)计算从u到n的路径最长长度

f[u]记录当前从u到n的路径最长长度

如果u出发到n没有路 f[u]和F(u)数值是-INF	例如 1e9	可否 改为0

此题是DAG最长路问题 初始值都设置-INF
若改成DAG最短路问题 初始值都设置为几?

F(u)计算从u到n的路径最长长度
f[u]记录当前从u到n的路径最长长度

作业要求	DP类程序开头用注释写明 状态定义的含义+手算表格

```
1  /*
2  F(u)计算从u到n的路径最长长度
3  f[u]记录当前从u到n的路径最长长度
4  输入节点数n=5,边数m=6,边集:
5  1 2 1
6  1 3 6
7  2 4 5
8  2 5 6
9  2 3 4
10 3 5 3
11    u=  1,  2,  3,   4, 5
12 f[u]=  9,  7,  3,-INF, 0
13 */
```

现场完成后
给老师检查

补全程序

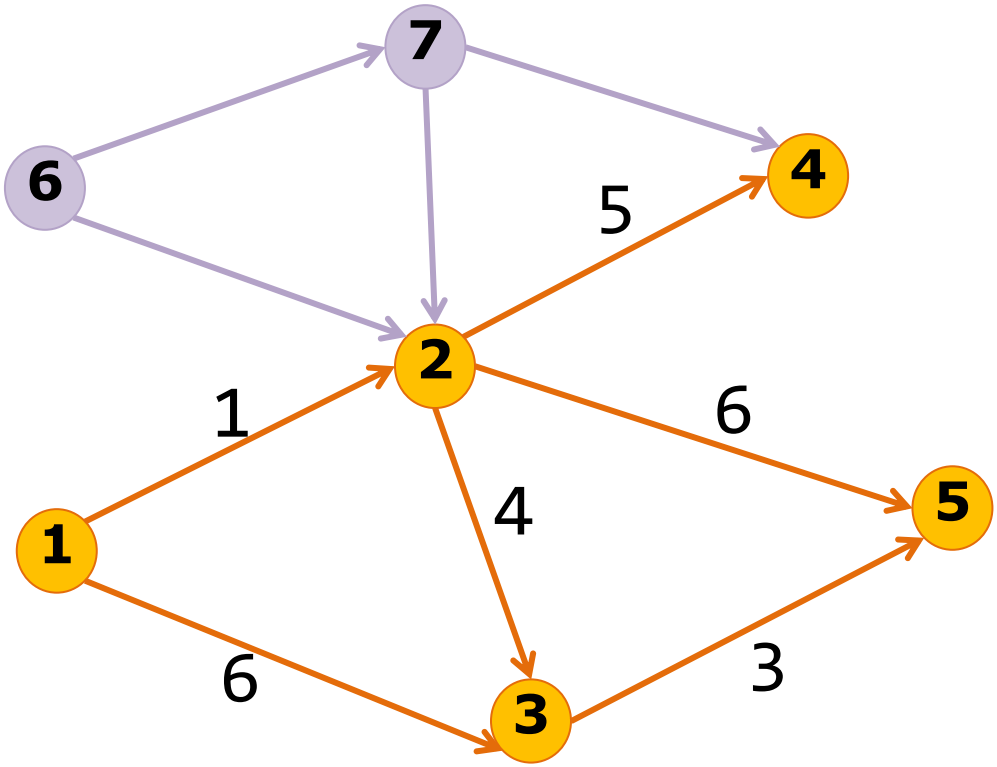
```
35      cin>>n>>m;
36      for(int i=0;i<m;i++){
37          int u,v,cost;
38          cin>>u>>v>>cost;
39          to[u].push_back(v);
40          w[ ].push_back(      );
41      }

42      for(int u=1;u<=n;u++)f[u]=-INF;
43      ok[n]=1; f[n]=0;
44      if(                  )
45          cout<<-1<<endl;
46      else
47          cout<<f[1]<<endl;
```

F(u)计算从u到n的路径最长长度
f[u]记录当前从u到n的路径最长长度
ok[u]代表f[u]是否更新完毕

补全程序

```
21 int F(int u){
22     if(ok[u])return f[u];
23     ok[u]=1;
24     for(int i=0;i<to[u].size();i++){
25         int v=to[u][i];
26         int cost=w[u][i];
27         if(                    )
28             f[u]=max(f[u],f[v]+cost);
29     }
30     return f[u];
31 }
```



记忆化搜索优点
无效状态 自动剔除 不访问

6号和7号节点
不会被DFS访问

如何查错

打印图的储存情况(如地图,邻接表)

打印f[]数组核对手算表格

太戈编程

	877
	158
	586
拓展题	587,588,205,495