【算法分析】

因为输入保证前面的城市比较靠前（西边），所以不存在a在b的西边，b也在a的西边啦，所以这是一个有向无环图，可以拓扑排序后动规。

到达v这一点可能有多种走法，所以要找最大值 Ans[v]=max(Ans[v],Ans[u]+1)。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | #include <bits/stdc++.h>  using namespace std;  const int N=100010;  int du[N],Ans[N];  vector <int> f[N];  stack <int> s;  int main()  {  int m,n,u,v;  cin>>n>>m;//输入城市和道路数  while(m--)  {  cin>>u>>v;  f[u].push\_back(v);  ++du[v];//入度加1  }  for(int i=1; i<=n; i++)  if(du[i]==0) //当入度为0存入堆栈  Ans[i]=1,s.push(i);  while(!s.empty())//边拓扑边动规  {  u=s.top();  s.pop();  int k=f[u].size()-1;  for(int i=0; i<=k; i++)//查找所有u东面的连线城市v  {  v=f[u][i];  --du[v];  Ans[v]=max(Ans[v],Ans[u]+1);//动规找最长路径  if(du[v]==0)//如果入度为0则入栈  s.push(v);  }  }  for(int i=1; i<=n; i++)  cout<<Ans[i]<<endl;  return 0;  } |