Pastebin Link: http://paste.ubuntu.com/24627641/

*/\**  
*adj keeps initial graph*  
*isBridge marks if the edge is a bridge*  
*invPos keeps the index of the opposite edge (v, u) in the adjacency list of v*  
*adj\_bridge\_tree keeps the bridge tree*  
*\*/*  
vector <**int**> adj[NODES], isBridge[NODES], invPos[NODES], adj\_bridge\_tree[NODES];

**int** tme=0, n, m; *//n, m is the number of nodes and edges respectively*  
**bool** vis[NODES];  
**int** d[NODES], f[NODES], low[NODES], paren[NODES], comp\_no[NODES];  
**int** comp\_cnt;*//Number of nodes in bridge tree*

**void** dfs(**int** node){ *//Identifies the bridges in the graph*  
 tme++;  
 d[node]=tme;  
 low[node]=d[node];  
 vis[node]=**true**;  
 **for**(**int** i=0; i<adj[node].size(); i++){  
 **int** v=adj[node][i];  
 **if**(!vis[v]){  
 paren[v]=node;  
 dfs(v);  
 **if**(low[v]>d[node]){  
 isBridge[node][i]=1;  
 isBridge[v][invPos[node][i]]=1;  
 }  
 low[node]=min(low[node], low[v]);  
 }  
 **else** **if**(v!=paren[node]){  
 low[node]=min(low[node], d[v]);  
 }  
 }  
 tme++;  
 f[node]=tme;  
}  
  
**void** dfs2(**int** nd, **int** comp){ *//Marks the component number of each node in a bridge tree*  
 vis[nd]=**true**;  
 comp\_no[nd]=comp;  
 **for**(**int** i=0; i<adj[nd].size(); i++){  
 **if**(isBridge[nd][i]==0){  
 **int** v=adj[nd][i];  
 **if**(!vis[v]){  
 dfs2(v, comp);  
 }  
 }  
 }  
}  
  
**void** init(){ *//Main function that builds the bridge tree*  
 tme=0;  
 memset(vis, **false**, **sizeof**(vis));  
 paren[1]=1;  
 dfs(1);  
 memset(vis, **false**, **sizeof**(vis));  
 comp\_cnt=0;  
 **for**(**int** i=1; i<=n; i++){  
 **if**(!vis[i]){  
 dfs2(i, comp\_cnt);  
 comp\_cnt++;  
 }  
 }  
 **for**(**int** i=1; i<=n; i++){  
 **for**(**int** j=0; j<adj[i].size(); j++){  
 **if**(comp\_no[i]!=comp\_no[adj[i][j]]){  
 adj\_bridge\_tree[comp\_no[i]].push\_back(comp\_no[adj[i][j]]);  
 adj\_bridge\_tree[comp\_no[adj[i][j]]].push\_back(comp\_no[i]);  
 }  
 }  
 }  
}  
  
**void** take\_input(){  
 */\*Takes the graph as input\*/*  
   
 scanf("%d %d", &n, &m);  
 **int** u, v;  
 **for**(**int** i=0; i<m; i++){  
 scanf("%d %d", &u, &v);  
 adj[u].push\_back(v);  
 adj[v].push\_back(u);  
 isBridge[u].push\_back(0);  
 isBridge[v].push\_back(0);  
 invPos[u].push\_back(adj[v].size()-1);  
 invPos[v].push\_back(adj[u].size()-1);  
 }