```cpp
/*** Articulation Point & Bridges:
In a graph, a vertex is called an articulation point if removing it and all the edges
associated with it results in the increase of the number of connected components in the graph.
An edge in a graph between vertices say u and v is called a Bridge, if after removing it,
there will be no path left between u and v. here Articulation Point & Bridges Basic Code: ***/
#include<bits/stdc++.h>
using namespace std;
pair<int,int>pii;
int tim,root,dis[10005],low[10005],par[10005];
vector<int>ed[10005];
vector< pair<int,int> >bridge;
bool ap[10005],vis[10005];
void dfs(int u){
    vis[u]=true;
    dis[u]=++tim;
    low[u]=tim;
    int child=0;

    for(int i=0; i<ed[u].size(); i++){
        int v = ed[u][i];
        if(vis[v]==false){
            par[v]=u;
            child+=1;
            dfs(v);

            low[u]=min(low[u],low[v]);

            if(u==root && child>1)ap[u]=true;
            if(u!=root && dis[u]<=low[v])ap[u]=true;
            if(dis[u]<low[v]) bridge.push_back(make_pair(u,v));
        } else if(v!=par[u]) { // backedge
            low[u]=min(low[u],dis[v]);
        }
    }
}
int main(){
    int tt; scanf("%d",&tt);
    for(int ks=1; ks<=tt; ks++){
        int n,m; scanf("%d%d",&n,&m);
        for(int i=1; i<=m; i++){
            int u,v; scanf("%d%d",&u,&v);
            ed[u].push_back(v);
            ed[v].push_back(u);
        }

        memset(ap,false,sizeof(ap));
        tim=0;
        root=1;
        dfs(1);

        printf("Articulation Points: ");
        for(int i=1; i<=n; i++){
            if(ap[i])printf("%d ",i);
        }
        printf("\n\n");

        printf("Articulation Bridges:\n");
        for(int i=0; i<bridge.size(); i++){
            pii = bridge[i];
            printf("%d %d\n",pii.first,pii.second);
        }
        printf("\n");

        for(int i=1; i<=n; i++)ed[i].clear();
    }

    return 0;
}
```