Pastebin Link: <https://paste.ubuntu.com/26148567/>

/\*

Add an edge between nodes u and v of capacity w

by making res[u][v]=w and adj[u][v]=adj[v][u]=true

\*/

**int** res[NODES][NODES], nxt[NODES][NODES];  
**bool** adj[NODES][NODES];  
**int** q[NODES], lvl[NODES];  
**int** src, sink;  
**inline** **void** init(){  
 memset(res, 0, **sizeof**(res));  
 memset(nxt, -1, **sizeof**(nxt));  
 memset(adj, **false**, **sizeof**(adj));  
}  
  
**inline** **void** process\_graph(){  
 **for**(**int** i=0; i<NODES; i++){  
 **for**(**int** j=NODES-2; j>=0; j--){  
 nxt[i][j]=(adj[i][j+1]) ? j+1 : nxt[i][j+1];  
 }  
 }  
}  
  
**bool** bfs(){  
 memset(lvl, -1, **sizeof**(lvl));  
 lvl[src]=0;  
 **int** st=0, ed=1;  
 q[st]=src;  
 **int** u, v;  
 **while**(st!=ed){  
 u=q[st];  
 st++;  
 v=(adj[u][0]) ? 0:nxt[u][0];  
 **for**(; v!=-1; v=nxt[u][v]){  
 **if**(lvl[v]==-1 && res[u][v]){  
 q[ed]=v;  
 lvl[v]=lvl[u]+1;  
 ed++;  
 }  
 }  
 }  
 **return** (lvl[sink]!=-1);  
}  
  
**int** dfs(**int** u, **int** flow){  
 **if**(u==sink) **return** flow;  
 **int** v=(adj[u][0])? 0:nxt[u][0];  
 **int** tmp;  
 **for**(; v!=-1; v=nxt[u][v]){  
 **if**(res[u][v] && lvl[v]==lvl[u]+1){  
 tmp=dfs(v, min(flow, res[u][v]));  
 **if**(tmp){  
 res[u][v]-=tmp;  
 res[v][u]+=tmp;  
 **return** tmp;  
 }  
 }  
 }  
 **return** 0;  
}  
  
**int** max\_flow(){  
 process\_graph();  
 **int** res=0, tmp;  
 **while**(bfs()){  
 **while**((tmp=dfs(src, INF))){  
 res+=tmp;  
 }  
 }  
 **return** res;  
}