```
/// Sample Code(dinic nafis saddique)
#define xx         first
#define yy         second
#define pb         push_back
#define mp         make_pair
#define LL         long long
#define inf        INT_MAX/3
#define mod        1000000007ll
#define PI         acos(-1.0)
#define linf       (1ll<<60)-1
#define FOR(I,A,B) for(int I = (A); I < (B); ++I)
#define REP(I,N)   FOR(I,0,N)
#define ALL(A)     ((A).begin(), (A).end())
#define set0(ar)   memset(ar,0,sizeof ar)
#define vsort(v)   sort(v.begin(),v.end())
#define setinf(ar) memset(ar,126,sizeof ar)

struct dinic_maxflow
{
    struct edge {
        int a, b, cap, flow ;
        edge(int _a,int _b,int _c,int _d) {
            a=_a,b=_b,cap=_c,flow=_d;
        }
    } ;

    int INF = 1500000001 ;

    int n,s,t,d[30001],ptr[3001],q[3001*10];
    vector < edge > e ;
    vector < int > g [ 3001 ] ;

    void add_edge ( int a, int b, int cap ) {
        edge e1 =edge( a, b, cap, 0 ) ;
        edge e2 =edge( b, a, 0 , 0 ) ;
        g [ a ] . push_back ( ( int ) e. size ( ) ) ;
        e. push_back ( e1 ) ;
        g [ b ] . push_back ( ( int ) e. size ( ) ) ;
        e. push_back ( e2 ) ;
    }

    bool bfs ( ) {
        int qh = 0 , qt = 0 ;
        q [ qt ++ ] = s ;
        memset ( d, -1 , sizeof d ) ;
        d [ s ] = 0 ;
        while ( qh < qt && d [ t ] == - 1 ) {
            int v = q [ qh ++ ] ;
            for(size_t i=0; i<g[v].size(); ++i) {
                int id = g [ v ] [ i ] ,
                    to = e [ id ] . b ;
                if(d[to]==-1 && e[id].flow<e[id].cap){
                    q [ qt ++ ] = to ;
                    d [ to ] = d [ v ] + 1 ;
                }
            }
        }
        return d [ t ] != - 1 ;
    }

    int dfs ( int v, int flow ) {
        if ( ! flow )  return 0 ;
        if ( v == t )  return flow ;
        for ( ; ptr[v]<(int)g[v].size(); ++ptr[v]){
            int id = g [ v ] [ ptr [ v ] ] ,
                to = e [ id ] . b ;
            if ( d [ to ] != d [ v ] + 1 )  continue ;
            int pushed = dfs(to,min(flow,e[id].cap-e[id].flow));
```

```cpp
            if ( pushed ) {
                e [ id ] . flow += pushed ;
                e [ id ^ 1 ] . flow -= pushed ;
                return pushed ;
            }
        }
        return 0 ;
    }

    int dinic ( ) {
        int flow = 0 ;
        for ( ;; ) {
            if ( !bfs () )  break ;
            memset ( ptr, 0 , sizeof ptr ) ;
            while ( int pushed = dfs ( s, INF ) ) {
                flow += pushed ;
                if(pushed == 0)break;
            }
        }
        return flow ;
    }

    dinic_maxflow( int _n, int _s, int _t ){
        n = _n; s = _s; t = _t;
    }
};
int A[101], B[101], out[101][101];
int main(){
    ios_base::sync_with_stdio(0); cin.tie(0);
    int n, m, sum = 0, sum1 = 0;
    cin >> n >> m;
    FOR(i, 1, n+1) {
        cin >> A[i];
        sum += A[i];
    }
    FOR(i, 1, n+1) {
        cin >> B[i];
        sum1 += B[i];
    }

    dinic_maxflow dm = dinic_maxflow(n+n+5, 0, n+n+1);

    FOR(i, 1, n+1){
        dm.add_edge(0, i, A[i]);
        dm.add_edge(n+i, n+n+1, B[i]);
        dm.add_edge(i, n+i, 1000);
    }

    REP(i, m){
        int p, q; cin >> p >> q;
        dm.add_edge(p, n+q, 1000);
        dm.add_edge(q, n+p, 1000);
    }

    int res = dm.dinic();
    if(res != sum || sum != sum1){
        cout << "NO" << endl;
        return 0;
    }
    cout << "YES" << endl;
    REP(i, dm.e.size()){
        if(dm.e[i].a>=1&&dm.e[i].a<=n&&dm.e[i].b>n&& dm.e[i].b<=n+n){
            out[dm.e[i].a][dm.e[i].b-n] = dm.e[i].flow;
        }
    }
    FOR(i, 1, n+1){
        FOR(j, 1, n+1) cout << out[i][j] << " ";
        cout << endl;
    }
}
```