```
pn = [1 1 -1 -1  1  1;
    -1  1  1  1  1 -1;
    -1 -1 -1 -1 -1  1];
% a.
w = memstor(pn)
```

```
w = 6×6
     0     1    -1    -1     1     1
     1     0     1     1     3    -1
    -1     1     0     3     1    -3
    -1     1     3     0     1    -3
     1     3     1     1     0    -1
     1    -1    -3    -3    -1     0
```

```
% b.
g = goodness(w);
```

```
ans = 64×7
    -1    -1    -1    -1    -1    -1     3
     1    -1    -1    -1    -1    -1     1
    -1     1    -1    -1    -1    -1    -7
     1     1    -1    -1    -1    -1    -5
    -1    -1     1    -1    -1    -1     1
     1    -1     1    -1    -1    -1    -5
    -1     1     1    -1    -1    -1    -5
     1     1     1    -1    -1    -1    -7
    -1    -1    -1     1    -1    -1     1
     1    -1    -1     1    -1    -1    -5
     ⋮
```

```
g(1)
```

```
ans =
     3
```

```
% c.
[~, index] = max(g);
index
```

```
index =
    31
g = 64×1
     3
     1
    -7
    -5
     1
    -5
    -5
    -7
     1
    -5
     ⋮
```

```
% pattern == [-1 1 1 1 1 -1]
```

Attachements

```
function mem=memstor(pats)
% each row of the matrix pats is a pattern
[np nd]= size(pats) ;
mem=zeros(nd) ;
for i=1:nd
    for j=1:nd
        if (i~=j)
            for k=1:np
                mem(i,j)=mem(i,j)+pats(k,i)*pats(k,j) ;
            end
        end
    end
end
end



function gvals = goodness( hopnet )
%calculates goodness for all patterns in a Hopfield Network
gvals=[];
pmat=[] ;
netsize=size(hopnet,1) ;
for k=0:(2^netsize-1)
    pvec=2*de2bi(k,netsize)-1; % need package
    %pvec=pvec([end:-1:1]) ;
    pmat=[pmat;pvec];
    g=0;
    for i=1:(netsize-1)
        for j=(i+1):netsize
            g=g+hopnet(i,j)*pvec(i)*pvec(j) ;
        end
    end
    gvals=[gvals, g];

end
gvals=gvals';
[pmat,gvals]
end
```