```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   This script is coded for calculate the QTM for
%   1D XXX spin chain under exactly diagonalization
%   method. The input "beta" presents the inverse
%   temperature in the partition function.
%   copyright by ycyu@wipm.ac.cn

m2id=eye(2);
m2z=[1,0;0,-1];

m4id=zeros(2,2,2,2);
m4id(1,1,1,1)=1;
m4id(1,2,1,2)=1;
m4id(2,1,2,1)=1;
m4id(2,2,2,2)=1;

m4p=zeros(2,2,2,2);
m4p(1,1,1,1)=1;
m4p(1,2,2,1)=1;
m4p(2,1,1,2)=1;
m4p(2,2,2,2)=1;

beta = 0.4;

%%%%%%%%%%%%
%  to generate the exact hamiltonian
%%%%%%%%%%%%
Num_site = 4;
hamiltonian = fn_exchange(Num_site,1,Num_site);
for k=1:Num_site-1

    add_on = fn_exchange(Num_site,k,k+1);
    hamiltonian = hamiltonian + add_on;

end

identity = fn_identity(Num_site);
identity = permute(identity,[1:2:(2*Num_site-1),2:2:(2*Num_site)]);
hamiltonian = hamiltonian - Num_site*identity;

tm2 = reshape(hamiltonian,2^Num_site,2^Num_site);
tm2 = exp(-beta*tm2);
Z = trace(tm2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   begin to calculate the maxium eigenvalue

for Num = 2:2:12

    lambda = beta/Num;

    T1 = 1/(1-lambda)*m4p + lambda/(lambda-1)*m4id;
    T2 = permute(T1,[4,1,2,3]);

    T1 = permute(T1,[1,3,2,4]);
    T2 = permute(T2,[1,3,2,4]);

    mpo = cell(1,Num);
```

```
    for i=1:Num

        if mod(i,2) == 0
            mpo{i} = T1;
        else
            mpo{i} = T2;
        end

    end

    transform_matrix = T2;
    num = 4;
    for i=2:Num

        append = mpo{i};
        transform_matrix = fn_contract(transform_matrix,num,num,...
            append,4,3);
        tv = [1:(num-2),num,num+1,num-1,num+2];
        transform_matrix = permute(transform_matrix,tv);
        num = num+2;

    end

    transform_matrix = fn_contract(transform_matrix,num,[num-1,num],...
        eye(2),2,[1,2]);
    num = num - 2;

    transform_matrix = permute(transform_matrix,...
        [1:2:(num-1),2:2:num]);
    tm = reshape(transform_matrix,2^Num,2^Num);
    tv = real(eig(tm));
    disp(['Num=',num2str(Num),'  max eigen=', num2str(max(tv))]);

    tm = tm^Num_site;
    Z2 = trace(tm);
    max_eigen = max(tv);

    disp(['Num=',num2str(Num),'  Z=', num2str(Z),...
         '  Z2=',num2str(Z2), ' estimate=',num2str(max_eigen^Num_site)]);

end

%%%%%%%  TEST CODE HEAR  %%%%%%%%%%%
% tm = fn_exchange(4,3,1);
% tm2 = reshape(permute(tm,[4,3,2,1,8,7,6,5]),16,16);
% disp(tm2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%% End the script %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%  Functions for the scripts  %%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function result = fn_identity(number)

result = eye(2);
num = 2;
for k=2:number
```

```
    result = fn_contract(result,num+1,num+1,reshape(eye(2),[1,2,2]),3,1);
    num = num + 2;
end
% result = permute(result,[1:2:(num-1),2:2:num]);

end

function result = fn_exchange(Num,ind1,ind2)

if Num <=1
    error('cndy');
end

m4p=zeros(2,2,2,2);
m4p(1,1,1,1)=1;
m4p(1,2,2,1)=1;
m4p(2,1,1,2)=1;
m4p(2,2,2,2)=1;

if Num ==2

    result = m4p;
    return;

end

result = permute(m4p,[1,3,2,4]);
num = Num - 2;

T = fn_identity(num);
result = fn_contract(result,5,5,reshape(T,[1,2*ones(1,2*num)]),...
    2*num+1,1);

result = permute(result,[1:2:(2*Num-1),2:2:(2*Num)]);

p1 = min(ind1,ind2);
p2 = max(ind1,ind2);

tv = 3:(Num+2);
if p1<Num
    tv(p1+1:end) = tv(p1+1:end) - 1;
end
if p2<Num
    tv(p2+1:end) = tv(p2+1:end) - 1;
end

tv(p1) = 1;
tv(p2) = 2;
tvp = tv + Num;

result = permute(result,[tv,tvp]);

end
```