```
%
clc;
clear;
strJobTitle='cdm_Tshape2_nouserx';
%save a copy of inp file, in case of wrong operation
strOdb=['cp ' strJobTitle '.inp ' strJobTitle '-old.inp'];
unix(strOdb);

%fdat = fopen([strJobTitle, '-old.inp']);
finp = fopen([strJobTitle '.inp'],'r');
aa='*NODE';

%read node information
while 1
    xtline=fgetl(finp);
    if ~ischar(xtline)
        fclose(finp);
        f=-1;
        break;
    else
        bComp=strcmp(xtline,aa);
        if(bComp)
            break;
        end
    end
end
blnIsNode=1;
cntNode=1;
while blnIsNode
    xtline=fgetl(finp);
    column=strread(xtline,'%s','delimiter',',');
    if ~strcmp(column(1),'*ELEMENT')
        feNodes(cntNode,:)=str2num(xtline);
        cntNode=cntNode+1;
    else
        blnIsNode=0;
        eleSet=strread(column{3},'%s','delimiter','=');
    end
end
maxNode=cntNode-1;
maxNodeID=feNodes(maxNode,1);

%read element information
blnIsElement=1;
cntEle=1;
eleSetName=[eleSet{2}];
eval(['eleSet' eleSetName '=[];']);
strSetNames=[eleSetName];

while blnIsElement
    xtline=fgetl(finp);
    column=strread(xtline,'%s','delimiter',',');
    if strcmp(column(1),'*ELEMENT')
        eleSet=strread(column{3},'%s','delimiter','=');
        eleSetName=[eleSet{2}];
        if isempty(findstr(strSetNames,eleSetName))
            strSetNames=[strSetNames ',' eleSetName];
            eval(['eleSet' eleSetName '=[];']);
        end
        %leave empty
        %get one more element section, is using both triangle and square
        %shape element
    elseif strcmp(column(1),'**') % end of element section
        blnIsElement=0;
    else
        column=str2num(xtline);
        if (length(column)==4)
            column(5)=-column(1);
        end
        feEles(cntEle,:)=column;
        cntEle=cntEle+1;
        eval(['eleSet' eleSetName '=[eleSet' eleSetName ',column(1)];']);
    end
end
maxEle=cntEle-1;
maxEleID=feEles(maxEle,1);
%close the input file
fclose(finp);

cntNewNode=0;
cntNewEle=0;
newNode=[];
newNodeStr=[];
newEle=[];
eleCoor=[]; %build the relation between old ele id and new ele id

cntReserveNode=0;
resNode=[];

%---these array declaration can be reduced by eval
setEleCPE3LOAD_SPHERE=[];
setEleCPE4ILOAD_SPHERE=[];
setEleCPE3PLASTIC_TOOTH=[];
setEleCPE4IPLASTIC_TOOTH=[];
setEleCPE3RESTORATION=[];
setEleCPE4IRESTORATION=[];
%---


setCDMtooth=[];
setCDMres=[];
setCDMinter=[];
setCDMuppertooth=[];
X=0;Y=0;

column=strread(strSetNames,'%s','delimiter',',');


for i=1:maxEle
    %in case of sphere element, keep the same node id, copy a new ele id
    sphereEleID=find(eleSetLOAD_SPHERE==feEles(i,1));
    if isempty(sphereEleID)
        %not sphere element
        %search the neighbour elements
        %numCurEle=(feEles(i,:));
        %copy a new element with new nodes id
        %array: [newNodeID, oldNodeID]
        cntNewNode=cntNewNode+1;
        [X,Y]=find(feNodes(:,1)==feEles(i,2));
        newNode(cntNewNode,:)=[maxNodeID+cntNewNode, feEles(i,2)];
        newNodeStr(cntNewNode,:)=[maxNodeID+cntNewNode feNodes(X,Y+1:Y+2)];
        cntNewNode=cntNewNode+1;

        [X,Y]=find(feNodes(:,1)==feEles(i,3));
        newNode(cntNewNode,:)=[maxNodeID+cntNewNode, feEles(i,3)];
        newNodeStr(cntNewNode,:)=[maxNodeID+cntNewNode feNodes(X,Y+1:Y+2)];
        cntNewNode=cntNewNode+1;

        [X,Y]=find(feNodes(:,1)==feEles(i,4));
        newNode(cntNewNode,:)=[maxNodeID+cntNewNode, feEles(i,4)];
        newNodeStr(cntNewNode,:)=[maxNodeID+cntNewNode feNodes(X,Y+1:Y+2)];
        if feEles(i,5)>0 %square element
            cntNewNode=cntNewNode+1;

            [X,Y]=find(feNodes(:,1)==feEles(i,5));
            newNode(cntNewNode,:)=[maxNodeID+cntNewNode, feEles(i,5)];
            newNodeStr(cntNewNode,:)=[maxNodeID+cntNewNode feNodes(X,Y+1:Y+2)];
        end

        %format of new element array: [newEleID, newNodeID1, newNodeID2,
        newNodeID3, newNodeID4]
        cntNewEle=cntNewEle+1;
        if feEles(i,5)>0
            newEle(cntNewEle,:)=[maxEleID+cntNewEle, maxNodeID+cntNewNode-3,
            maxNodeID+cntNewNode-2, maxNodeID+cntNewNode-1,
            maxNodeID+cntNewNode];
            %eleCoor=[eleCoor; [feEles(i,1) maxEleID+cntNewEle]];%relate the
            old ele id and new ele id
            eleType=['CPE4I'];
            for k=1:length(column)
                eval(['eleIndex=find(eleSet' column{k} '==feEles(i,1));']);
                if ~isempty(eleIndex)
                    break;
                end
            end
            eval(['setEle' eleType column{k} '=[setEle' eleType column{k}
            ';newEle(cntNewEle,:)];']);

        else%triangle element
            newEle(cntNewEle,:)=[maxEleID+cntNewEle, maxNodeID+cntNewNode-2,
            maxNodeID+cntNewNode-1, maxNodeID+cntNewNode, -1];
            %eleCoor=[eleCoor; [feEles(i,1) maxEleID+cntNewEle]];
            eleType=['CPE3'];
            for k=1:length(column)
                eval(['eleIndex=find(eleSet' column{k} '==feEles(i,1));']);
                if ~isempty(eleIndex)
                    break;
                end
            end
            eval(['setEle' eleType column{k} '=[setEle' eleType column{k}
            ';newEle(cntNewEle,1:4)];']);

        end
    else
        %sphere element
        %push old node id into node list
        %cntOldNode=cntOldNode+1;
        %reorganize the node id of sphere?
        for k=2:5
            if feEles(i,k)>0 & isempty(find(resNode==feEles(i,k)))
                cntReserveNode=cntReserveNode+1;
                resNode=[resNode; feEles(i,k)];
            end
        end
        %copy a new element id with old nodes
        cntNewEle=cntNewEle+1;
        if feEles(i,5)>0
            % 4 nodes
            newEle(cntNewEle,:)=[maxEleID+cntNewEle, feEles(i,2:5)];
            eleType=['CPE4I'];
            for k=1:length(column)
                eval(['eleIndex=find(eleSet' column{k} '==feEles(i,1));']);
                if ~isempty(eleIndex)
                    break;
                end
            end
            eval(['setEle' eleType column{k} '=[setEle' eleType column{k}
            ';newEle(cntNewEle,:)];']);

        else
            % 3 nodes
            newEle(cntNewEle,:)=[maxEleID+cntNewEle,feEles(i,2:4), -1];
            eleType=['CPE3'];
            for k=1:length(column)
                eval(['eleIndex=find(eleSet' column{k} '==feEles(i,1));']);
                if ~isempty(eleIndex)
                    break;
                end
            end
            eval(['setEle' eleType column{k} '=[setEle' eleType column{k}
            ';newEle(cntNewEle,1:4)];']);

        end
    end
end

%re-organize the reserve sphere nodes
resNode=sort(resNode);
resNodeStr=[];
for i=1:length(resNode)
    idxNode=find(feNodes==resNode(i));
    if ~isempty(idxNode)
        resNodeStr=[resNodeStr; feNodes(idxNode,:)];
    end
end

%search the common pair to create new CDM elements
cntShareNode=0;
cntNewPair=maxEleID+cntNewEle+1;
oldNode=[];
oldPairs=[];
newPair=[];
%cdmTooth=[];
%cdmRestoration=[];
%cdmInterface=[];
for i=1:maxEle
    %exclude the sphere elements
    sphereEleID=find(eleSetLOAD_SPHERE==feEles(i,1));
    if isempty(sphereEleID)
        % not sphere element, find the neighbour
        numCurEle=(feEles(i,2:5));
        %find possible neighbour elements, which must share same nodes
        %
        aryNeighbour=[];
        for k=2:5
            for nk=2:5
                if feEles(i,k)>0
                    idxEle=find(feEles(:,nk)==feEles(i,k));
            if (~isempty(idxEle))
                for idxRepeat=1:length(idxEle)
                    if idxEle(idxRepeat)~=i &
                    isempty(find(aryNeighbour==idxEle(idxRepeat)))
                        aryNeighbour=[aryNeighbour; idxEle(idxRepeat)];
                    end
                end
            end
        end
    end
end
for j=1:length(aryNeighbour)

%end

%for j=1:maxEle

    %if (j~=i)
        cntShareNum=0;
        cNode=[];
        for k=2:5
            numNeighbourNode=(feEles(aryNeighbour(j),k));
            hasComonNode=find(numCurEle==numNeighbourNode);
            if length(hasComonNode)~=0 & feEles(aryNeighbour(j),k)>0
            %has common node
                %suppose two neighbour elements can only share one pair
                %of common nodes
                cntShareNum=cntShareNum+1;
                cNode(cntShareNum)=numNeighbourNode;

            end
        end

        %has two share nodes
        %has problem in logic, need update
        isNewPair=0;
        pairNodes=[];
        if length(cNode)==2
            pairNodes=(sort(cNode));
            if isempty(oldPairs)
                oldPairs=[oldPairs; pairNodes];
                isNewPair=1;
            else
                idxPair=find(oldPairs(:,1)==pairNodes(1));
                if isempty(idxPair)
                    oldPairs=[oldPairs; pairNodes];
                    isNewPair=1;
                else
                    nRepeatPair=0;
                    for n=1:length(idxPair)
                        if oldPairs(idxPair(n),2)==pairNodes(2)
                            nRepeatPair=nRepeatPair+1;
                        end
                    end
                    if nRepeatPair==0
                        oldPairs=[oldPairs; pairNodes];
                        isNewPair=1;
                    end
                end
            end
        end
```

```
if isNewPair
    cntNewPair=cntNewPair+1;
    %specially, the same rows of feEle and newEle have the same
    %array index here
    %the neighbour elements: maxEleID+i  maxEleID+j
    %find the new node id according to the old id
    cntShareNum=0;
    orderNode=[];
    for k=2:5

        [X,Y]=ind2sub(size(newNode),find(newNode==newEle(i,k)))
        ;
        curNode=newNode(X,Y+1);
        for m=2:5

            [X2,Y2]=ind2sub(size(newNode),find(newNode==newEle(
            aryNeighbour(j),m)));
            nbNode=newNode(X2,Y2+1);
            if curNode==nbNode & curNode~=-1
                cntShareNum=cntShareNum+1;
                nNodes(cntShareNum,:)=[newEle(i,k)
                newEle(aryNeighbour(j),m)];
                orderNode=[orderNode; k curNode];
            end
        end
    end
    %rotation direction
    %find the original direction of one side
    %follow that side direction to construct the node order
    %of new element

    if cntShareNum==2
        if (orderNode(2,1)-orderNode(1,1))==1
            newPair(cntNewPair,:)=[cntNewPair [nNodes(2,1)
            nNodes(1,1) nNodes(2,2) nNodes(1,2)]];
        else
            newPair(cntNewPair,:)=[cntNewPair [nNodes(1,1)
            nNodes(2,1) nNodes(1,2) nNodes(2,2)]];
        end
        %newPair(cntNewPair,:)=[cntNewPair [nNodes(1,1)
        nNodes(2,1) nNodes(1,2) nNodes(2,2)]];
        idxEle1=find(eleSetPLASTIC_TOOTH==feEles(i,1));

        idxEle2=find(eleSetRESTORATION==feEles(aryNeighbour(j),
        1));

        if (isempty(idxEle1) & isempty(idxEle2)) |
        ((~isempty(idxEle1)) & (~isempty(idxEle2)))
            %interface CDM
            setCDMinter=[setCDMinter; newPair(cntNewPair,:)];
        elseif (isempty(idxEle1) & (~isempty(idxEle2)))
            %tooth CDM
            setCDMres=[setCDMres; newPair(cntNewPair,:)];
        elseif ((~isempty(idxEle1)) & isempty(idxEle2))
            %restoration CDM

            yCoorNodes=(newNodeStr(find(newNodeStr(:,1)==nNodes
            (1,1)),3)+newNodeStr(find(newNodeStr(:,1)==nNodes(2
            ,1)),3))/2;
            if yCoorNodes>10.65
                setCDMuppertooth=[setCDMuppertooth;
                newPair(cntNewPair,:)];
```

```
                            else
                                setCDMtooth=[setCDMtooth;
                                newPair(cntNewPair,:)];
                            end
                            %setCDMtooth=[setCDMtooth; newPair(cntNewPair,:)];

                        end
                    end
                end
            %end
        end
        %
    %end of not sphere element
    end
end
%fprintf(fid,'Coef 0 to 90: a=%8.4f  b=%8.4f\n',coefC1,coefC2);

%%data to output
%Node information, newNodeStr
%Ordinary Node information, newEle
%CDM Element information, newPair, format: [elementID, pair1_1, pair1_2,
pair2_1, pair2_2]
%the elementID of CDM element starts from 1 in array, so need to shift
%before output,
%pair1_1 and pair1_2 are the same pair, & pair2_1 <==> pair2_2
%write the output according the required abaqus inp file format:
dlmwrite('load_sphere_Nodes.dat',resNodeStr,'delimiter',',');
dlmwrite('newNodeInfo.dat',newNodeStr,'delimiter',',');

dlmwrite('load_sphere_Eles3.dat',setEleCPE3LOAD_SPHERE,'delimiter',',');
dlmwrite('load_sphere_Eles4.dat',setEleCPE4ILOAD_SPHERE,'delimiter',',');
dlmwrite('plastic_tooth_Eles3.dat',setEleCPE3PLASTIC_TOOTH,'delimiter',',');
dlmwrite('plastic_tooth_Eles4.dat',setEleCPE4IPLASTIC_TOOTH,'delimiter',',');
dlmwrite('restoration_Eles3.dat',setEleCPE3RESTORATION,'delimiter',',');
dlmwrite('restoration_Eles4.dat',setEleCPE4IRESTORATION,'delimiter',',');
dlmwrite('cdm_tooth.dat',setCDMtooth,'delimiter',',');
dlmwrite('cdm_uppertooth.dat',setCDMuppertooth,'delimiter',',');
dlmwrite('cdm_restoration.dat',setCDMres,'delimiter',',');
dlmwrite('cdm_interface.dat',setCDMinter,'delimiter',',');
%dlmwrite('newEleInfo.dat',newEle,'delimiter',',');
%newPair(:,1)=newPair(:,1)+max(newEle(:,1));
%dlmwrite('newCDMEle.dat',newPair,'delimiter',',');

s=['the end']
```