```
clc
clear all
format short
%simplex
C=[-1 3 -2];
info=[3 -1 2; -2 4 0; -4 3 8];
b=[7; 12; 10];
NOVariables=size(info,2);
s=eye(size(info,1));
A=[info s b];
Cost=zeros(1,size(A,2));
Cost(1:NOVariables)=C;
%Constraint Basic Variable
BV=NOVariables+1:size(A,2)-1;
%calculate Zj-Cj row
ZRow=Cost(BV)*A-Cost;
%To print the table
ZjCj=[ZRow;A];
SimpTable=array2table(ZjCj);
SimpTable.Properties.VariableNames(1:size(ZjCj,2))={'x_1','x_2','x_3','s_1'
,'s_2','s_3','Sol'}
%Simplex Table starts
Run=true;
while Run
if any(ZRow<0) %To check any negative value is there
fprintf('The current BFS is not Optimal \n')
fprintf('\n ============The Next Iteration Results========\n' )
disp('Old Basic Variable (BV)=')
disp(BV)
%To find the entering variable
ZC=ZRow(1:end-1);
[EnterCol,Pvt_Col]=min(ZC);
fprintf('The most negative element in ZRow is %d Corresponding to Column %d
\n' ,EnterCol, Pvt_Col)
fprintf('Entering Variable is %d \n', Pvt_Col)
%To find the leaving variable
sol=A(:,end);
Column=A(:,Pvt_Col);
if all(Column<=0)
    error('LPP has unbounded solution.All entries <=0 in Column
%d',Pvt_Col)
else
    % To check minimus ration is with positive entering column entries
    for i=1:size(Column,1)
if Column(i)>0
ratio(i)=sol(i)./Column(i);
else
    ratio(i)=inf;
end
end
%To Finding the minimum Ratio
[MinRatio, Pvt_Row]=min(ratio);
fprintf('Minimum ratio corresponding to pivot row is %d \n' ,Pvt_Row)
fprintf('Leaving Variable is %d \n', BV(Pvt_Row))
end
BV(Pvt_Row)=Pvt_Col;
disp('New Basic Variables (BV) =')
disp(BV)
```

```
%Pivot Key
Pvt_Key=A(Pvt_Row,Pvt_Col);
%Update the Table for next Iteration
A(Pvt_Row,:)=A(Pvt_Row,:)./Pvt_Key;
for i=1:size(A,1)
if i~=Pvt_Row
A(i,:)=A(i,:)-A(i,Pvt_Col).*A(Pvt_Row,:);
end
ZRow=ZRow-ZRow(Pvt_Col).*A(Pvt_Row,:);
%To print the table
ZjCj=[ZRow;A];
SimpTable=array2table(ZjCj);
SimpTable.Properties.VariableNames(1:size(ZjCj,2))={'x_1','x_2','x_3','s_1'
,'s_2','s_3','Sol'}
BFS=zeros(1,size(A,2));
BFS(BV)=A(:,end);
BFS(end)=sum(BFS.*Cost);
CurrentBFS=array2table(BFS);
CurrentBFS.Properties.VariableNames(1:size(CurrentBFS,2))=s{'x_1','x_2','x_
3','s_1','s_2','s_3','Sol'}
end
else
Run=false;
fprintf('======**============\n')
fprintf('The current BFS is optimal and Optimality is reached \n' )
fprintf('======**============\n')
end
end
```