# (1) SIMPLEX CODE :-

```
clc
clear all
%% defining parameters
c = [4 6 3 1];
a = [1 4 8 6; 4 1 2 1; 2 3 1 2];
b = [11; 7; 2]
s = eye(size(a, 1))
variables = size(a, 2)
%% creating table
var = {'x1', 'x2', 'x3', 'x4', 's1', 's2', 's3', 'sol'}
A = [a s b]
cost = zeros(1, size(A,2))
cost(1: variables) = c
bv = variables+1: size(A, 2)-1
zjcj = cost(bv)*A - cost
zcj = [zjcj; A]
ST = array2table(zcj, VariableNames=var)
%% finding solution
RUN = true
while RUN
    if any(zjcj(1:end-1) < 0)
        fprintf('Not optimal solution')
        zc = zjcj(1:end-1)
        [enter_val, pvt_col] = min(zc)
        if all(zcj(:, pvt_col)) <= 0
            fprintf('LPP is unbounded')
        else
            sol = A(:, end)
            column = A(:, pvt_col)
            for i=1:size(A, 1)
                if column(i) > 0
                    ratio(i) = sol(i)./column(i)
                else
                    ratio(i) = inf
                end
            end
            [leaving_val, pvt_row] = min(ratio)
        end
        bv(pvt_row) = pvt_col
        pvt_key = A(pvt_row, pvt_col)
        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
        end
        zjcj = cost(bv)*A - cost
        next_table = [zjcj; A]
        ST = array2table(next_table, VariableNames=var)
    else
        RUN = false
        fprintf("The final optimal value is ")
         zjcj(end)
   end
end
```

# (2) Big M :-

```
% Big M method
clc
clear all
M=1000;
%---1---
 cost=[-3 -5 0 0 -M -M 0];
 a=[1 3 -1 0 1 0; 1 1 0 -1 0 1];
 b=[3;2];
 A=[a b];
 bv=[5 6];
 Var={'x1','x2','x3','s1','s2','s3','A1','A2','sol'};
 %cost=[20 10 0 0 0 -M -M 0];
 %a=[1 2 1 0 0 0 0;3 1 0 -1 0 1 0;4 3 0 0 -1 0 1];
 %b=[40;30;60];
artifical_var=[7 8];
% bv=[3 6 7];
% cost=[1 1 0 0 -M 0];
% a=[2 1 -1 0 1;1 7 0 1 0];
% b=[4;7];
% artifical_var=[5];
% bv=[4 5];
%cost=[5 3 0 0 0 -M 0];
%a=[1 1 1 0  0  0;5 2 0 1  0  0;2 8 0 0 -1  1];
%b=[2;10;12];
%artifical_var=[6];
%bv=[3 4 6];
%----3---
% cost=[3 2 0 0 -M 0];
% artifical_var=5
% bv=[3 5];
% a=[2 1 1 0 0; 3 4 0 -1 1];
% b=[2;12];
A=[a b];
Var={'x1','x2','s1','s2','s3','A1','sol'};
% %----2--
% cost=[4 -3 0 0 -M -M 0];
% a=[2 1 -1 0 1 0; 2 -1 0 -1 0 1];
% b=[0;2];
% artifical_var=[5 6]
% A=[a,b];
% Var={'x1','x2','s1','s2','A1','A2','sol'};
% bv=[5 6];
zjcj=cost(bv)*A-cost
% Display initial simplex table
simplex_table=[zjcj; A];
array2table(simplex_table,'VariableNames',Var)
RUN=true;
 while RUN
if any(zjcj(1:end-1)<0) % check for negative value
 fprintf(' The current BFS is not optimal \n');
 zc=zjcj(1:end-1);
 [Enter_val, pvt_col]= min(zc) ;
 if all(A(:,pvt_col)<=0)
  error('LPP is Unbounded');
 else
 sol=A(:,end);
 column=A(:,pvt_col);
```

```
     for i=1:size(A,1)
    if column(i)>0
    ratio(i)= sol (i)./column(i);
    else
    ratio(i)=inf;
    end
     end
     [leaving_value,pvt_row]=min(ratio);
    end
    bv(pvt_row)=pvt_col;
    pvt_key=A(pvt_row, pvt_col);
    A(pvt_row,:)=A (pvt_row,:)./pvt_key;
    % row operation
   for i=1:size(A,1)
    if i~=pvt_row
    A(i,:)=A(i,:)-A (i, pvt_col).*A(pvt_row,:);
    end
   end
    zjcj=cost(bv)*A-cost;
    next_table=[zjcj; A]
   array2table(next_table,'VariableNames',Var)

   else
       RUN=false;
       if any(bv==artifical_var(1))
           error('Infeasible solution');
       else
       fprintf('The table is optimal \n');
       end
       z=input(' Enter 0 for minimization and 1 for max \n');

       if z==0
           Obj_value=-zjcj(end);
       else
           Obj_value=zjcj(end);
       end
       fprintf('The final optimal value is % f \n',Obj_value);
   end
    end
```

# (3) Two Phase :-

```
% Two phase method
clc
clear all
%-----Problem 1----
Variables={'x1','x2','s1','s2','A1','A2','sol'};
OVariables={'x_ 1','x_ 2','s_ 1','s_ 2','sol'}; % original variables
OrigC=[-3 -5 0 0 -1 -1 0];
a=[1 3 -1 0 1 0; 1 1 0 -1 0 1];
b=[3;2];
A=[a b];

%-----Problem 2----
% Variables={'x1','x2','s1','s2','A1','A2','sol'};
% OVariables={'x_ 1','x_ 2','s_ 1','s_ 2','sol'};
% OrigC=[-3 4 0 0 -1 -1 0];
% a=[1 -1 -1 0 1 0; 2 -1 0 -1 0 1];
% b=[0;2];
% A=[a b];
% PHASE-1
fprintf('********** PHASE-1 ********** \n')
%----Problem 1-----
cost=[0 0 0 0 -1 -1 0]
Artifical_var=[5 6]
bv=[5 6];

 %----Problem 2----
%  cost=[ 0 0 0 0 -1 -1 0];
%  Artifical_var=[5 6];
%  bv=[5 6];

zjcj=cost(bv)*A-cost
simplex_table=[zjcj;A]
array2table(simplex_table,'VariableNames',Variables)
RUN=true;
while RUN
if any(zjcj(1:end-1)<0) % check for negative value
 fprintf(' the current BFS is not optimal \n')
 zc=zjcj(1:end-1);
 [Enter_val, pvt_col]= min(zc);
 if all(A(:,pvt_col)<=0)
 error('LPP is Unbounded all enteries are <=0 in column % d',pvt_col);
 else
 sol=A(:,end);
 column=A(:,pvt_col);
 for i=1:size(A,1)
 if column(i)>0
 ratio(i)= sol (i)./column(i);
 else
 ratio(i)=Inf;
 end
 end
 [leaving_val, pvt_row]=min(ratio);
 end
bv(pvt_row)=pvt_col;
pvt_key=A(pvt_row, pvt_col);
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
end
%  zjcj=zjcj-zjcj (pvt_col).*A(pvt_row,:);
zjcj=cost(bv)*A-cost;
 next_table=[zjcj;A];
 table=array2table(next_table,'VariableNames',Variables)
else
 RUN=false;
 if any(bv==Artifical_var(1)) || any(bv==Artifical_var(2))
     error('Infeasible solution');
 else
  fprintf('optimal table of phase-1 is achieved \n');
 end
end
end

% PHASE-2
fprintf('********** PHASE-2 ********** \n')
A(:,Artifical_var)=[]; % Removing Artificial var by giving them empty value
OrigC(:,Artifical_var)=[]; % Removing Artificial var cost by giving them empty
value
cost=OrigC;
zjcj=cost(bv)*A-cost;
simplex_table=[zjcj;A];
array2table(simplex_table,'VariableNames',OVariables)

RUN=true;
while RUN
if any (zjcj(1:end-1)<0) % check for negative value
 fprintf('The current BFS is not optimal \n')
 zc=zjcj(1:end-1);
 [Enter_val, pvt_col]= min(zc);
 if all(A(:,pvt_col)<=0)
 error('LPP is Unbounded, all entries are <=0 in column %d',pvt_col);
 else
 sol=A(:,end);
 column=A(:,pvt_col);
 for i=1:size(A,1)
 if column(i)>0
 ratio(i)= sol (i)./column(i);
 else
 ratio(i)=inf;
 end
 end
 [leaving_val, pvt_row]=min(ratio);
 end
bv(pvt_row)=pvt_col;
pvt_key=A(pvt_row, pvt_col);
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
end
%  zjcj=zjcj-zjcj (pvt_col).*A(pvt_row,:);
zjcj=cost(bv)*A-cost;
```

```
next_table=[zjcj;A];
 table=array2table(next_table,'VariableNames',OVariables)
else
 RUN=false;
  fprintf('The current BFS is optimal \n');
  z=input(' Enter 0 for minimization and 1 for max \n');
    if z==0
        Obj_value=-zjcj(end);
    else
        Obj_value=zjcj(end);
    end
    fprintf('The final optimal value is %f \n',Obj_value);
 end
end
```

# (4) Dual Simplex :-

```
clc
clear all
format short

cost = [-2 0 -1 0 0 0];
A = [-1 -1 1 1 0 -5; -1 2 -4 0 1 -8];
bv = [4 5]
%S = eye(size(A1,1))


%cost = zeros(1,size(A,2))
%variables = size(A1,1)
%cost(1:variables)= C

%bv = variables+1:size(A,2)-1

zjcj = cost(bv)*A - cost
zc = [zjcj;A]
simplex_table = array2table(zc,'VariableNames',{'x1','x2','x3','s1','s2','sol'})

run = true
while(run)
    sol = A(:,end)
    if any(sol <0)
        fprintf('table is not faesible')
        fprintf('\n')
        %fprintf('old basic variables %d \n',bv)
        z = zjcj(1:end-1)
        [leaving_val pvt_row]= min(sol)
        fprintf('leaving variable is %d',pvt_row)

        solu =A(:,end)
        row = A(:,pvt_row)
        if(row<=0)
            fprintf('problem is unbounded')
        else
            for i=1:size(A,2)-1
                if A(pvt_row , i)<0
                    ratio(i) = abs(zjcj(i) /A(pvt_row , i));
                else
```

```
                    ratio(i)= inf
                end
            end
        end
        [entering_val pvt_col]=min(ratio)
        fprintf('entering variable %d',pvt_col)
        bv(pvt_row) = pvt_col
        %fprintf('new basic variables are ')
        %display(bv)
        pvt_key=A(pvt_row,pvt_col)

        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
        end
        zjcj=zjcj-zjcj(pvt_col)*A(pvt_row,:);
        zcj = [zjcj ; A]
simplest = array2table(zcj,'VariableNames',{'x1','x2','x3','s1','s2','sol'})
        bfs = zeros(1,size(A,2))
        bfs(bv)= A(:,end)
        bfs(end) = sum(bfs .* cost);
currentbfs = array2table(bfs,'VariableNames',{'x1','x2','x3','s1','s2','sol'})
    else
        run = false
        fprintf('table is feasible')
    end
end
```