Name: Shreeya Chatterji
Class: CO16
Roll Number: 102103447

EXPERIMENT 5

Q1.

$$\text{Maximize} \quad Z = x_1 + 2x_2$$

$$\begin{aligned} \text{s.t.} \quad -x_1 + x_2 &\leq 1, \\ x_1 + x_2 &\leq 2, \\ x_1, x_2 \geq 0 \end{aligned}$$

Code:

```
clc
clear all
Noofvariables=2;
variables={'x1','x2','s1','s2','sol'};
c=[1 2]; % cost of objective func
Abar=[-1 1;1 1];% const coeff
B=[1;2]; %RHS of constraints
s=eye(size(Abar,1));
A=[Abar s B];
Cost=zeros(1,size(A,2));
Cost(1:Noofvariables)=c;
% Contraints BV
BV=Noofvariables+1:1:size(A,2)-1;
% To calculate Zj-Cj
ZjCj=Cost(BV)*A-Cost;
% For printing 1st simplex table
ZCj=[ZjCj;A];
simplextable=array2table(ZCj);
simplextable.Properties.VariableNames(1:size(ZCj,2))=variables;
% Start simplex Algorithm
Run=true;
while Run
    if any(ZjCj<0) % to check if any negative value there
        fprintf('The current BFS is not optimal\n')
        fprintf('Next iteration required \n')
        disp('Old basic variable (BV)=')
        disp(BV)
        % For finding entering variable
        Zc=ZjCj(1:end-1);
        [Ent_col pvt_col]=min(Zc);
        fprintf('The most negative value in Zj-Cj row is %d and coresponding to 
column %d \n',Ent_col,pvt_col)
        fprintf('Entering variable is %d \n',pvt_col)
        %For finding the leaving variable
        sol=A(:,end);
        column=A(:,pvt_col);
        if all(column<=0)
            error('The LPP has unbounded solution \n since all enteries are <=0 in 
%d \n',pvt_col)
        else
            for i=1:size(column,1)
                if column(i)>0
                    ratio(i)=sol(i)./column(i)
                else
                    ratio(i)=inf
```

Name: Shreeya Chatterji
Class: CO16
Roll Number: 102103447

```
            end
        end
        % To finding minimmum ratio
        [minratio pvt_row]=min(ratio);
        fprintf('The minimum ratio corresponding to pivot row %d \n ',pvt_row)
        fprintf('leaving variable is %d \n ',BV(pvt_row))
        BV(pvt_row)=pvt_col;
        disp('New basic variable(BV)==')
        disp(BV)
        pvt_key=A(pvt_row,pvt_col)
        % To update 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
            ZjCj=ZjCj-ZjCj(pvt_col).*A(pvt_row,:);
        end

             end
    else
        Run= false;
        ZCj=[ZjCj;A]
        FinalTable=array2table(ZCj);
        FinalTable.Properties.VariableNames(1:size(ZCj,2))=variables
        FinalTable.Properties.RowNames(1:size(ZCj,1))={'Zj-Cj','x1','x2'}
        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))={'x1','x2','s1','s2','Op
t.Val of Z'}
        disp('Optimal sol is reached')
    end
end
```

Final Solution:

```
currentBFS =

  1×5 table

    x1     x2     s1    s2    Opt.Val of Z
    ___    ___    __    __    ____________

    0.5    1.5    0     0         3.5
```

Name: Shreeya Chatterji
Class: CO16
Roll Number: 102103447

Q2.

$$\text{Minimize} \quad Z = x_1 - 3x_2 + 2x_3$$

$$\begin{aligned} \text{s.t.} \quad 3x_1 - x_2 + 2x_3 &\leq 7, \\ -2x_1 + 4x_2 &\leq 12, \\ -4x_1 + 3x_2 + 8x_3 &\leq 10, \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

Code:

```
clc
clear all
Noofvariables=3;
variables={'x1','x2','x3','s1','s2','s3','sol'};
c=[-1 3 -2]; % cost of objective func
Abar=[3 -1 2;-2 4 0;-4 3 8];% const coeff
B=[7;12;10]; %RHS of constraints
s=eye(size(Abar,1));
A=[Abar s B];
Cost=zeros(1,size(A,2));
Cost(1:Noofvariables)=c;
% Contraints BV
BV=Noofvariables+1:1:size(A,2)-1;
% To calculate Zj-Cj
ZjCj=Cost(BV)*A-Cost;
% For printing 1st simplex table
ZCj=[ZjCj;A];
simplextable=array2table(ZCj);
simplextable.Properties.VariableNames(1:size(ZCj,2))=variables;
% Start simplex Algorithm
Run=true;
while Run
    if any(ZjCj<0) % to check if any negative value there
        fprintf('The current BFS is not optimal\n')
        fprintf('Next iteration required \n')
        disp('Old basic variable (BV)=')
        disp(BV)
        % For finding entering variable
        Zc=ZjCj(1:end-1);
        [Ent_col pvt_col]=min(Zc);
        fprintf('The most negative value in Zj-Cj row is %d and coresponding to 
column %d \n',Ent_col,pvt_col)
        fprintf('Entering variable is %d \n',pvt_col)
        %For finding the leaving variable
        sol=A(:,end);
        column=A(:,pvt_col);
        if all(column<=0)
            error('The LPP has unbounded solution \n since all enteries are <=0 in 
%d \n',pvt_col)
        else
            for i=1:size(column,1)
                if column(i)>0
                    ratio(i)=sol(i)./column(i)
                else
                    ratio(i)=inf
                end
```

Name: Shreeya Chatterji
Class: CO16
Roll Number: 102103447

```
            end
            % To finding minimmum ratio
            [minratio pvt_row]=min(ratio);
            fprintf('The minimum ratio corresponding to pivot row %d \n ',pvt_row)
            fprintf('leaving variable is %d \n ',BV(pvt_row))
            BV(pvt_row)=pvt_col;
            disp('New basic variable(BV)==')
            disp(BV)
            pvt_key=A(pvt_row,pvt_col)
            % To update 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
                ZjCj=ZjCj-ZjCj(pvt_col).*A(pvt_row,:);
            end

                 end
    else
        Run= false;
        ZCj=[ZjCj;A]
        FinalTable=array2table(ZCj);
        FinalTable.Properties.VariableNames(1:size(ZCj,2))=variables
        FinalTable.Properties.RowNames(1:size(ZCj,1))={'Zj-Cj','x1','s2','x3'}
        BFS=zeros(1,size(A,2));
        BFS(BV)=A(:,end)
        BFS(end)=0-sum(BFS.*Cost);
        currentBFS=array2table(BFS);

currentBFS.Properties.VariableNames(1:size(currentBFS,2))={'x1','x2','x3','s1','s2
','s3','Opt.Val of Z'}
        disp('Optimal sol is reached')
    end
end
```

Final Solution:

```
currentBFS =

  1×7 table
```

x1	x2	x3	s1	s2	s3	Opt.Val of Z
4	5	0	0	0	11	-11

Name: Shreeya Chatterji
Class: CO16
Roll Number: 102103447

Q3.

$$\text{Maximize } Z = 5x_1 + 3x_2$$

$$\begin{aligned} \text{s.t.}\quad 3x_1 + 5x_2 &\le 15, \\ 5x_1 + 2x_2 &\le 10, \\ x_1, x_2 &\ge 0 \end{aligned}$$

Code:

```
clc
clear all
Noofvariables=2;
variables={'x1','x2','s1','s2','sol'};
c=[5 3]; % cost of objective func
Abar=[3 5;5 2];% const coeff
B=[15;10]; %RHS of constraints
s=eye(size(Abar,1));
A=[Abar s B];
Cost=zeros(1,size(A,2));
Cost(1:Noofvariables)=c;
% Contraints BV
BV=Noofvariables+1:1:size(A,2)-1;
% To calculate Zj-Cj
ZjCj=Cost(BV)*A-Cost;
% For printing 1st simplex table
ZCj=[ZjCj;A];
simplextable=array2table(ZCj);
simplextable.Properties.VariableNames(1:size(ZCj,2))=variables;
% Start simplex Algorithm
Run=true;
while Run
    if any(ZjCj<0) % to check if any negative value there
        fprintf('The current BFS is not optimal\n')
        fprintf('Next iteration required \n')
        disp('Old basic variable (BV)=')
        disp(BV)
        % For finding entering variable
        Zc=ZjCj(1:end-1);
        [Ent_col pvt_col]=min(Zc);
        fprintf('The most negative value in Zj-Cj row is %d and coresponding to column %d \n',Ent_col,pvt_col)
        fprintf('Entering variable is %d \n',pvt_col)
        %For finding the leaving variable
        sol=A(:,end);
        column=A(:,pvt_col);
        if all(column<=0)
            error('The LPP has unbounded solution \n since all enteries are <=0 in %d \n',pvt_col)
        else
            for i=1:size(column,1)
                if column(i)>0
                    ratio(i)=sol(i)./column(i)
                else
                    ratio(i)=inf
                end
```

Name: Shreeya Chatterji
Class: CO16
Roll Number: 102103447

```
            end
            % To finding minimmum ratio
            [minratio pvt_row]=min(ratio);
            fprintf('The minimum ratio corresponding to pivot row %d \n ',pvt_row)
            fprintf('leaving variable is %d \n ',BV(pvt_row))
            BV(pvt_row)=pvt_col;
            disp('New basic variable(BV)==')
            disp(BV)
            pvt_key=A(pvt_row,pvt_col)
            % To update 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
                ZjCj=ZjCj-ZjCj(pvt_col).*A(pvt_row,:);
            end

                 end
    else
        Run= false;
        ZCj=[ZjCj;A]
        FinalTable=array2table(ZCj);
        FinalTable.Properties.VariableNames(1:size(ZCj,2))=variables
        FinalTable.Properties.RowNames(1:size(ZCj,1))={'Zj-Cj','x1','x2'}
        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))={'x1','x2','s1','s2','Op
t.Val of Z'}
        disp('Optimal sol is reached')
    end
end
```

Final Solution:

```
currentBFS =

  1×5 table

     x1         x2       s1    s2    Opt.Val of Z
    ______    ______    __    __    ____________

    1.0526    2.3684    0     0        12.368

Optimal sol is reached
```