Dual Simplex

```
clc
clear all
n=4;
m=2;
b=[-2;-4]
A=[-1 -1 1 0;-4 -1 0 1];
BV=[3 4];
c=[-5 -6 0 0]
for i=1:50
    B=A(:,BV)
    cb=c(BV)
    Xb=inv(B)*b
    z=cb*Xb
    Y=inv(B)*A
    zjcj=cb*Y-c
     if(Xb>=0)
       disp('Feasibilty Achieved')
       Xb
     break
     else
        [a,LV]=min(Xb)
           for j=1:n
             if(Y(LV,j)<0)
                 ratio(j)=zjcj(j)/Y(LV,j)
             else
                ratio(j)=inf
            end
           end
        [l,EV]=min(abs(ratio))
        BV(LV)=EV
     end
end
zjcj=[inf zjcj z;BV' A Xb]
```

Transportation Problem

```
cost=[2 7 4;3 3 1;5 5 4;1 6 2];
cc=cost
n=size(cost,1);
m=size(cost,2);
supply=[5 8 7 14];
demand=[7 9 18];
X=zeros(n,m)
if(sum(supply)==sum(demand))
    disp('balanced problem')
elseif sum(supply)<sum(demand)
    disp('unbalanced problem')
    cost=[cost;zeros(1,m)]
    supply=[supply sum(demand)-sum(supply)]
else
    disp('unbalanced problem')
    cost=[cost zeros(n,1)]
    supply=[supply sum(supply)-sum(demand)]
end
for i=1:n
    for j=1:m
        temp=min(cost(:));
        %cost(:) will convert 2d to vector
        [r,c]=find(temp==cost)
        y=min(supply(r),demand(c))
        [val,index]=max(y)
        pos_row=r(index)
        pos_col=c(index)
        X(pos_row,pos_col)=val
        supply(pos_row)=supply(pos_row)-val
        demand(pos_col)=demand(pos_col)-val
        cost(pos_row,pos_col)=inf
    end
end
fc=cc.*X
final_cost=sum(fc(:))
X
```

Fibonacci

```
clc
clear all
f=@(x) x^2
a=-5;
b=15;
n=7;
F(1)=1;
F(2)=1;
for i=3:n+1
    F(i)=F(i-1)+F(i-2);
end
table=[];
for k=1:n
    x1=a+(F(n-2+1)/F(n+1))*(b-a);
    x2=b-(F(n-2+1)/F(n+1))*(b-a);
    fx1=f(x1);
    fx2=f(x2);
    table=[table; k x1 x2 fx1 fx2];
```

```
    if(fx1<fx2)
        b=x2
    else
        a=x1
    end
end
opt=(a+b)/2
optval=f(opt)
disp(table)
```

Weighted Sum:

```
clc
clearvars

mocost=[3 2 0;2 3 0]
w=1/size(mocost,1)
socost=sum(mocost.*w)

C=socost
A=[-1 -1 1 -2]
BV=[3]
answer=[0 0 0]

zjcj=[0 0 0 0]

RUN=true;
while RUN
    for i=1:size(A,2)-1
        zjcj(i)=sum(C(BV)*A(:,i))-C(i);
    end
    if any(A(:,size(A,2))<0)
        disp('the current BFS is not feasible')
        [lval, pvt_row]=min(A(:,size(A,2)))
        for i=1:size(A,2)-1
            if A(pvt_row,i)<0
                m(i)=zjcj(i)/A(pvt_row,i)
            else
                m(i)=-inf
            end
        end
        [ent_val, pvt_col]=max(m)
        A(pvt_row,:)=A(pvt_row,:)/A(pvt_row,pvt_col)
        for i=1:size(A,1)
            if i~=pvt_row
                A(i,:)=A(i,:)-A(i,pvt_col).*A(pvt_row,:)
            end
        end
        BV(pvt_row)=pvt_col;
    else
        RUN=false;
        disp('current BFS is Feasible and Optimal\n')
        answer(BV)=A(:,size(A,2))
    end
end
```