```
clc
clear all

%%outside matlab functions called.
%function f = moody(ed,Re) % ed=relative roughness = epsilon / hd
%function [Co] =roundtorectangular(theta45,CA45)F

%Uvisc_Ci=airProp2(Tci,'my'); dynamic viscosity of air
%Vvisc_Ci=airprop2(Tci,'ny');kinematic viscosity of air
%hs01=XSteam('h_pT',P_Hi,T_Hi-273.15);
%% reference for airprop2 function
%col-#     prop.      units
%   ------------------------
%    1          T          K
%    2          rho        kg/m^3
%    3          cp         J/(kg K)
%    4          my         kg/ms dynamic viscosity (mew)
%    5          ny         m^2/s kinimatic viscosity (v)
%    6          k          W/(m K)
%    7          alpha m^2/s
%    8          Pr         -

% reference for tubeProp function
%col-#     prop.      units
%   ------------------------
%    1          Re         Renolds Number
%    2          F          f-effectivness
%    3          P          StPr^2/3 (Chilton + Colburn J factor)

% Test1=tubeProp(2050,'F')
% Test2=tubeProp(2050,'P')
%% DEFINITION OF CONSTANTS(DYNAMIC)  %%
% % OPTIMIZATION PARAMETERS (TO BE CHANGED PER OPTIMIZATION RUN)
start_V_C=4.6; %starting velocity(do not start @ zero
final_V_C=4.6; %incrimental changes in velocity
n=1;      %number of calculations/ incrimental increases in velocity.
fins=275*.5; %number of fins along tube, same for all tubes(275 per
meter, 7 per inch)
Nl=4; % number of rows. Nl
Nt=10;%number of tubes on first row(one less on even rows, staggard
arangement). Nt
%SSP=4;%Saturated steam pressure in BAR


iii=10000;
error=1/iii;

%   PARAMETERS AND DIMENTIONS OF THE HEAT EXCHANGER
%15":0.381m  18":0.4572m 24":0.6096m diameters of fan.
D_fan=.762;%tube diameter(determined by fan)
A_fan=(3.1416*(D_fan*.5)^2);
r12=.0762;%inside radius of flange: 3"(determined by incoming bell
housing)
L24=.3;%length of piping between fan and system boundary
L45=.3;%length of piping between fan and exchanger(round to square)
L78=.3;%length of duct work after exchanger



% TEMPORARY CONSTANTS(TO BE DYNAMICLY CALCULATED)
%hC=0;%convection coefficent of pipe to air
%hH=0;%convection coefficent of steam to pipe
deltavA=(final_V_C-start_V_C)/n;
Q_1=0;
Q_2=0;
Q_3=0;
T_Hi=450;%T of incoming steam
P_Hi=5.7;%P of incoming steam
T_Ho=410;%T of outgoing steam(INITIAL GUESS)
P_Ho=5.7;%P of outgoing steam(INITIAL GUESS)
T_sat=XSteam('Tsat_p',P_Hi)+273.15;
T_Ci=300; %incoming tempeture of air in Kelvin
T_Co=320; %(INITIAG GUESS)


Q=0;%Heat transfer rate
Qi=0;%Ideal heat transfer rate
hs01=XSteam('h_pT',P_Hi,T_Hi-273.15);
hs12=XSteam('h_px',((P_Hi+P_Ho)/2),1);
hs23=XSteam('h_px',((P_Hi+P_Ho)/2),0);
hs30=XSteam('h_pT',P_Ho,T_Ci-273.15);

QR_ZONE_1=(hs01-hs12)*.03*1000;%Energy needed to cool steam to T_sat
QR_ZONE_2=(hs12-hs23)*.03*1000;%Energy needed to condense steam
QR_ZONE_3=(hs23-hs30)*.03*1000;%Energy needed to cool steam to T_Ci
QR_Max=QR_ZONE_1+QR_ZONE_2+QR_ZONE_3;

% DYNAMIC VARIABLES(SET AT ZERO TO CLEAR CODE)
% ReA=0; %renolds number of atmosphere( >2700? Turbulent flow??)
% ReS=0; %renolds number of steam in tube
% StA=0; %Stanton number?
%% GIVEN CONSTANTS (DO NOT CHANGE!!!)
V_cond=0;%Velocity that causes condensation;
T_Co=320; %outgoing tempeture of air in Kelvin (INITIAL GUESS)
epsilon=.0152;%(.0002~copper & aluminum)relative roughness of inside
of pipe)
k=60.5;%W/m*K:Aluminum(237):conduction value of copper 60.5 (plain
carbon from table A.1)
m_dot_H=.03;%%(kg/sec)of steam
OD=.01638;    %%(m)outside diameter of tube
ID=.0138;     %%(m)inside diameter of tube
WT=.0013;     %%(m)thickness of wall
finD=.0285;   %%(m)diameter of fin
```

```
tubespace=.0313; %(m)distance between tubes in a row, centroid to
centroid
rowspace=.0343; %(m)perpindicular distance between rows, centroid to
centroid
finlength=.00605; %(m)length of fin from outside diameter of fin
finT=.000254; %(m) thickness of fin
finspace=.003629; %distance between fins along tube, centroid to
centroid
c=1.7378;%ratio of fin to tube diameter
HD=.00668;%Hydraulic Diameter
tubeCSin=.00014957; %(m^2) Internal cross section area of tube
sigma=.449; % ratio of minimum(restricted)flow, over maximum(open)
flow.
%% SECONDARY STATIC VARIABLES(CALCULATED FROM GIVIN OPTIMIZATION
VARIABLES)
%optimization of variables before optimization loop.
% [SHOULD BE NO CONSTANTS]

%PROPERTIES OF TUBE DUE TO GIVEN GEOMETRY
length=finspace*(fins+1);%The length of each tube
%length=finspace*(fins);%The length of each tube
height=((tubespace*Nt)+.02);
%     tubecount=Nl*Nt-floor(Nl/2); % Total number of tubes in
exchanger,
%1cm added at top and bottom for additional pipe clearence
i=1;
rack=zeros(1,Nl);
for i=1:2:Nl
    rack(i)=Nt;
end
i=1;
for i=2:2:Nl
    rack(i)=Nt-1;
end
tubecount=sum(rack); % Total number of tubes in exchanger,
fincount=fins*tubecount; % Total number of fins in exchanger
Afr=height*length; %Maximum(unristricted) cross-section area of
exchanger(m^2)
finCS=finlength*2*finT*fins*Nt; %first row cross section of fins
perpindicular to movement of air
tubeCSpr=(length-fins*finT)*(OD*Nt); %first row cross section of tubes
perpindicular to movement of air
Aff=Afr-(tubeCSpr+finCS);% [plus fin]Minimum(constricted) cross-
section area of exchanger(m^2)
tube_Af=(((finD)^2-OD^2)*pi*.5+finT*(finD)*pi)*fins;
%[(Surface*2+Tip)*fins]Surface area of fins on each tube
Af=tube_Af*tubecount;
tube_SA=(pi*OD)*(length-fins*finT);%[length of tube-length of
fins]:Surface area of each tube
tube_A=tube_SA+tube_Af;%CombinedSurface area of each tube:
A_C=tube_A*tubecount;
A_H=(3.1416*length*ID*tubecount);
```

```
V_tube=((3.1416*(OD*.5)^2)*length)+((3.1416*(finD*.5)^2)*(fins*finT));
%Sigma according to book should be 0.449 (free flow/frontal area)
%    sigma=Aff/Afr; %the ratio of minimum flow area over maximum flow
area. (corresponds to change in velocity)
%According to Kays/London, Hydraulic diameter is infered to be a
constant
%     %WP=[lengthalongtubes*2sides+length along wall*2sides+ length
along fins*4sides*numberoffins]
%     WP=(2*length+2*(tubespace-OD)+finlength*4*fins)*tubes; %the
perimiter at the smallest section
%     HD=Aff*4/WP%Flow passage hydraulic diameter (4*area/wetted
perimiter)
%According to Kays/London.
% 216 (m^2/m^3) Heat Transfer area/total volume
%.862 Fin area/total volume
edS=epsilon/ID; %length of tube over diameter
m_dot_tube=m_dot_H/tubecount; %m_dot rate per tube, steam flow rate /
total number of tubes.
m_dot_C=start_V_C*Afr*1.1774;
vS= m_dot_tube*.3312/tubeCSin;%Velocity of steam in each tube (.3312
density of steam @ incoming conditions)

% CODUCTION THOUGH PIPE (is a constant variable)
    h_F=((ID)*log(OD/ID))/(2*k*(.143));
% PROPERTIES OF AIR FROM INCOMING CONDITIONS
Pr_Ci=airProp2(T_Ci,'Pr'); %prussel(sp?) number
Cp_Ci=airProp2(T_Ci,'cp'); %specific heat
Uvisc_Ci=airProp2(T_Ci,'my'); %dynamic viscosity of air
Vvisc_Ci=airprop2(T_Ci,'ny');%kinematic viscosity of air
rho_Ci=airProp2(T_Ci,'rho'); %density kg/m^3
Cp_HV=XSteam('CpV_p',P_Hi)*1000;%watts/Kg
Cp_HL=XSteam('CpL_p',P_Hi)*1000;%watts/Kg
C_1H=m_dot_H*Cp_HV;
%C_2H=m_dot_H*((Cp_SV+Cp_SL)*.5);%Mathmatically invalid : C_2H
=infinity
C_3H=m_dot_H*Cp_HL;
%A1=.01;
i=1;
v_Ci=start_V_C;
v_Co=start_V_C;
out=zeros(30,n);
%Values used for global heat model transfer combination
Q_MAX_1H=QR_ZONE_1+QR_ZONE_2+QR_ZONE_3;
Q_MAX_2H=QR_ZONE_2+QR_ZONE_3;
Q_MAX_3H=QR_ZONE_3;

%Values used for heat transfer coefficent of steam to tube
        Pr_L=1.09; %From Book Table A.6
        K_l=XSteam('tcL_p',5.7); %
        rho_L=XSteam('rhoL_p',5.7);
        rho_G=XSteam('rhoV_p',5.7);
        Uvisc_L=XSteam('my_pT',5.7,150);%where 150=150 centigrade
1.8248e-004
        C=5.03;%RE<50,000
        nn=1/3;%RE<50,000

        x1=.99;
        x2=.5;
        x3=.01;
%% OPTIMIZATION LOOP
%%%%%%%%%%%%%%%%%%%%
while i <= n
    %% Variables to be recalculated for every run
    m_dot_C=v_Ci*Afr*rho_Ci; %m_dot rate of Air though exchanger,
density of air at 300K according to book
    flowrate_A=v_Ci*Afr;%m^3/s, Volumetric flow rate
    Pr_Co=airProp2(T_Co,'Pr'); %prussel(sp?) number
    Cp_Co=airProp2(T_Co,'cp'); %specific heat
    C_1C=m_dot_C*Cp_Ci;
    C_2C=m_dot_C*Cp_Ci;
    C_3C=m_dot_C*Cp_Ci;

    Q_MAX_1C=Cp_Ci*(T_Hi-T_Ci)*m_dot_C;
    Q_MAX_2C=Cp_Ci*(T_sat-T_Ci)*m_dot_C;
    Q_MAX_3C=Cp_Ci*(T_sat-T_Ci)*m_dot_C;

    % if Q_MAX_1H > Q_MAX_1C
    % Q_MAX_1=Q_MAX_1C
    % else
    % Q_MAX_1=Q_MAX_1H
    % end
    %
    % if Q_MAX_2H > Q_MAX_1C
    % Q_MAX=Q_MAX_2C
    % else
    % Q_MAX=Q_MAX_2H
    % end
    %
    % Q_MAX_2=Q_MAX_2C;%Due to Cr relation, Q_MAX always equal to air
side;
    % if Q_MAX_3H > Q_MAX_3C
    % Q_MAX_3=Q_MAX_3C
    % else
    % Q_MAX_3=Q_MAX_3H
    % end
    %% SOLVING FOR CONVECTION (Pipe to Air) h_c
    %Heat transfer from the fins surface to the air
    %Alternative methods for calculating 'G'
    G=rho_Ci*v_Ci/sigma;
    % PrA=.708;
    % Cp=1.0057;
    % viscA= 1.9830e-005;
    % rhoA=1.1774;
    %StA=hC/(G*Cp)%statons number air
    %jH=StA*PrA^(2/3)%Colburn j factor
    %jH=hC/(G*CP)*PrA^(2/3)
    Re_Ci=G*HD/Uvisc_Ci; %reynolds number air calculation
    %jHold=tubeProp(ReA,'P'); %Function interpretation, from table in
book unique to exchanger
    jH=0.1464*Re_Ci^-0.337;
    %Stantons number and j factor are used to estimate hC.
    %hC is the heat convection heat transfer coeficent for the tubes
and fins.
    %By solving for the J factor, the Statton factor is known, the
Statton
    %factor can then be solved fro the heat transfer coefficent.
    h_C=jH*Cp_Ci*G/(Pr_Ci^(2/3)); %Watts/m^2*K(Cold side between
exchanger and air)
    %% SOLVING FOR CONVECTION (STEAM TO PIPE) h_h %%
    %Constants needed for Akers
    if i==1

    end
    %Equation is divided into 3 regions
    %%Zone 1:SUPER HEATED STEAM%%
    mdot_1H=  (m_dot_tube/(3.1416*(ID^2)*.25))*((1-
x1)+x1*(rho_L/rho_G)^.5);
    Re_1H=mdot_1H*ID/Uvisc_L;
    h_1H=C*(Re_1H^nn)*(Pr_L^(1/3))*K_l*(1/ID);

    %%Zone 2:CONDENSING ZONE%%
    mdot_2H=  (m_dot_tube/(3.1416*(ID^2)*.25))*((1-
x2)+x2*(rho_L/rho_G)^.5);
    Re_2H=mdot_2H*ID/Uvisc_L;
    h_2H=C*(Re_2H^nn)*(Pr_L^(1/3))*K_l*(1/ID);

    %%Zone 3:FULLY CONDENSED STEAM%%
    mdot_3H=  (m_dot_tube/(3.1416*(ID^2)*.25))*((1-
x3)+x3*(rho_L/rho_G)^.5);
    Re_3H=mdot_3H*ID/Uvisc_L;
    h_3H=C*(Re_3H^nn)*(Pr_L^(1/3))*K_l*(1/ID);
    %% SOLVING FOR PRESSURE LOSS OF AIR OVER PIPES%%
    %Hloss=H34=12H+H23+H45+H67+H78+H89 (in Pascals)

    %Static Variables
    if i==1;
        DH68=(4*height*length)/(2*(height+length)); %Hydraulic
Diameter of fan region
        CA45=(Afr/(3.1416*(D_fan*.5)^2));%Change of cross section area
between fan and exchanger
        if CA45 < .1;
            CA45=.1;
        end
        theta45h=atan((.5*length-
D_fan*.5)/L45)*2*(180/3.1415);%horizontal angle of slope of ductwork
between fan and heat exchanger
        theta45v=((.5*height-D_fan*.5)/L45)*2*(180/3.1415);%vertical
angle of slope of ductwork
        visc69=airProp2(T_Co,'my'); %dynamic viscosity of air
        rho69=airProp2(T_Co,'rho'); %density
        edA=epsilon/HD;%friction coefficent of fin surface/hydraulic
diameter of fin, used for moody diagram to calculate Renolds number
        if theta45h>theta45v;
            theta45=theta45h;
        else
            theta45=theta45v;
        end
    end
    theta45=45;%[REMOVE LATER: BENS CONSTANT]

    %H12:Atmosphere over incoming duct
    v12=v_Ci*(height*length)/(.25*3.1415*D_fan^2);%Function is
calculating the velocity of the air at the fan as a ratio(CrossSection
Fan/Exchanger)*(Velocity at exchanger)
    Kl=-.126*log(r12/D_fan)-.1554;%Function is logritmic function
developed in excell
    H12=0.5*rho_Ci*v12^2*Kl;

    %H23:incoming duct
    Re23=rho_Ci*L24*v12/Uvisc_Ci;
    ed23=epsilon/D_fan;
    f23 = moody(ed23,Re23);
    H23=f23*(L24/D_fan)*(v_Ci^2/(2*9.81));

    %H34:Head pressure of fan(not used)

    %H45:duct between fan and pipes
    Co=roundtorectangular(theta45,CA45);%Co referes to external
function interpolating a table
    v45=v12;%(variable needs to be redefined? ASK BEN!)
    H45=Co*.5*rho_Ci*v45^2;

    %H56:Delta pressure over pipes (NEEDS REVISION! DISCUSS WITH BEN!)
    Re56=Re_Ci;
    f56=0.2287*Re56^-0.224;
    H56=(G^2*(1/rho_Ci)*.5)*((1+sigma^2)*((v_Ci/v_Co)-
1)+f56*(A_C/Aff)*((.5*(v_Ci+v_Co))/v_Ci));
    %H67:Louvers?(Not Used)
    H67=0;

    %H78:Exiting duct work
    v78=v_Ci;
    Re78=rho_Ci*L78*v78/Uvisc_Ci;
    ed78=epsilon/(DH68);
    f78 = moody(ed78,Re78);
    H78=f78*(L78/DH68)*(v78^2/(2*9.81));
    %H89:end duct work to atmosphere
    Kl89=1;%VALUE IS ONE FOR EXITING AIR
    v89=v_Ci;
    H89=0.5*Kl89*rho_Ci*v89^2;

    deltaPA=H12+H23+H45+H56+H67+H78+H89;%Combination of all pressures
except fan H34
    %% GLOBAL HEAT MODEL COMBINATION
    %U=1/r+1/r+1/r
    eff_f=.89; %effectivness of fin From figure 3.19[TO BE OPTIMIZED
INTO EQUATION]
    eff_h=1-(Af/A_C)*(1-eff_f);%effictivness of exchanger
    % Cr_ZONE1=(C_3C/C_H3)
    % if Cr_ZONE1 >1
    %     Cr_ZONE1=(C_1C/C_1H)
    % end
    % Cr_ZONE3=(C_3C/C_H3)
    % if Cr_ZONE3 > 1
    %     Cr_ZONE3=1/Cr_ZONE3;
    %% ZONE 1 ZONE 1 ZONE 1 ZONE 1 ZONE 1 ZONE 1 ZONE 1 ZONE 1 ZONE 1
    % SOLVE FOR FUNCTION ASSUMING C_MIN IS AIR : cold
    ii=1;
    if i ==1;
        Z_1C=.2;
    else
        Z_1C=out(29,i-1);
    end

    while ii < iii;%Solves for minimum length assuming C_1min is C_C1

U_1C=(1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_1H*A_H*eff_h))))*Z_1C;
        Cr_ZONE_1C=C_1C/(C_1H*Z_1C);
        NTU_ZONE_1C=U_1C*A_C/C_1C;
        e_ZONE_1C=1-exp((1/Cr_ZONE_1C)*((NTU_ZONE_1C)^.22)*(exp(-
Cr_ZONE_1C*(NTU_ZONE_1C)^.78)-1));
        Q_1C=Q_MAX_1C*e_ZONE_1C*Z_1C;
        %Optimization loop for Z_1C
        if Q_1C > QR_ZONE_1*(1+error); % When Z_1C needs to be reduced
            Z_1C=Z_1C*(1-(QR_ZONE_1/Q_1C)*(1/20))
            ii=ii+1;
        elseif Q_1C < QR_ZONE_1*(1-error); % When Z_1C needs to be
increased,
            Z_1C=Z_1C*(1+(Q_1C/QR_ZONE_1)*(1/20))
            ii=ii+1;
        else Q_1C >= QR_ZONE_1*(1-error) && Q_1C <=
QR_ZONE_1*(1+error) % When value has been optimized.
            ii=iii+ii;
            Z_1=Z_1C;
            Q_MAX_1=Q_MAX_1C
            Q_1=Q_1C;
        end
    end
    ii_count(1,i)=ii;

    % logic statement to determine which C[C_H1, C_C1] is minimum.

    if m_dot_C*Z_1C*Cp_Ci <= m_dot_H*Cp_HV; %[C_C1 < C_CH1]
        Z_1=Z_1C;
        Q_MAX_1=Q_MAX_1C
        Q_1=Q_1C;
        Z_1H=0;
    else %[C_H1 < C_C ]C1%Logic statements solves for length for C_
MIN as C_H1 STEAM
        ii=1;
        if i ==1;
            Z_1H=.2;
        else
            Z_1H=out(30,i-1);
        end
        ii=1;

        while ii < iii; % SOLVE FOR FUNCTION ASSUMING C_MIN IS STEAM :
hot

U_1H=(1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_1H*A_H*eff_h))))*Z_1H;
            Cr_ZONE_1H=C_1H/C_1C*Z_1H;
            NTU_ZONE_1H=U_1H*A_C*Z_1H/(C_1H);
            e_ZONE_1H=1-exp((1/Cr_ZONE_1H)*((NTU_ZONE_1H)^.22)*(exp(-
Cr_ZONE_1H*(NTU_ZONE_1H)^.78)-1));
            Q_1H=Q_MAX_1H*e_ZONE_1H;
            %Optimization loop for Z_1H
            if Q_1H > QR_ZONE_1*(1+error); % When Z_1H needs to be
reduced
                Z_1H=Z_1H*(1-(QR_ZONE_1/Q_1H)*(1/20))
                ii=ii+1
            elseif Q_1H < QR_ZONE_1*(1-error); % When Z_1H needs to be
increased,
                Z_1H=Z_1H*(1+(Q_1H/QR_ZONE_1)*(1/20))
                ii=ii+1
            else
                % Q_1H >= QR_ZONE_1*(1-error) && Q_1H <=
QR_ZONE_1*(1+error) % When value has been optimized.
                ii=iii+ii
                Z_1=Z_1H;
                Q_MAX_1=Q_MAX_1H
                Q_1=Q_1H;
            end
        end
        Z_1=Z_1H;
        Q_MAX_1=Q_MAX_1H
        Q_1=Q_1H;
    end
    ii_count(2,i)=ii;
    %% ZONE 2 ZONE 2 ZONE 2 ZONE 2 ZONE 2 ZONE 2 ZONE 2 ZONE 2 ZONE 2
    % SOLVE FOR FUNCTION ASSUMING CR_MIN IS AIR : cold
    % Function will always solve for cold side as C_minimum
    % C_H is always infinite due to definition of C
    % Q_max is always defined by the maximum capacity of the air.
    ii=1;
    Z_2C=1-Z_1;

    %Initial run @ max possible tube length for Z_2
    U_2C=1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_2H*A_H*eff_h)))*Z_2C;
    NTU_ZONE_2C=U_2C*A_H/C_2C;
    e_ZONE_2C=1-exp(-NTU_ZONE_2C);
    Q_2C=Q_MAX_2C*e_ZONE_2C*Z_2C;
    if Q_2C <= QR_ZONE_2;%Consensation is not long enouph to condense
steam
        display('Exchanger is unable to condense steam')
        Q_2=Q_2C;
        Z_2=1-Z_1;

    else % Funtion to calculate needed length for condensate to form
        if ii ==1;
        Z_2C=1-Z_1;
        else
        Z_2C=out(27,i-1)
        end
        while ii < iii; % solve with C_H= infinity, C_C = C_min.

U_2C=1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_2H*A_H*eff_h)))*Z_2C;
            Cr_ZONE2=0;
            NTU_ZONE_2C=U_2C*A_H/C_2C
            e_ZONE_2C=1-exp(-NTU_ZONE_2C)
            Q_2C=Q_MAX_2C*e_ZONE_2C*Z_2C

            %Test to determine if tube is long enouph to condense
steam
            if Q_2C < QR_ZONE_2 && Z_2C >1-Z_1;

                Z_2=1-Z_1;
                ii=iii+ii;
                Z_2=Z_2C;

                Q_2=Q_2C;
            end;
            % Logic loop to iterate length of section 2
            if Q_2C > QR_ZONE_2*(1+error); % When Z_2C needs to be
reduced
                Z_2C=Z_2C*(1-(QR_ZONE_2/Q_2C)*(1/10));
                ii=ii+1;
            elseif Q_2C < QR_ZONE_2*(1-error); % When Z_2C needs to be
increased,
                Z_2C=Z_2C*(1+(Q_2C/QR_ZONE_2)*(1/10));
                ii=ii+1;
            else  %When value has been optimized.
                ii=iii+ii;
                Z_2=Z_2C;
                Q_MAX_2=Q_MAX_2C
                Q_2=Q_2C;
            end
        end
    end
    ii_count(3,i)=ii;
    % SOLVE FOR FUNCTION ASSUMING CR_MIN IS STEAM : hot
    % ii=1;
    % Z_2H=.5;
    %     while ii < 100;
    %
U_2H=(1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_2H*A_H*eff_h))))*Z_2H;
    %         NTU_ZONE_2H=U_2H*A_H*Z_2H/(C_2C);
    %         e_ZONE_2H=1-exp(-NTU_ZONE_2H);
    %         Q_2H=Q_MAX_2H*e_ZONE_2H;
    %
    %         if Q_2H > QR_ZONE_2*1.1;
    %             Z_2H=Z_2H-.01
    %             ii=ii+1;
    %         end
    %         if Q_2H < QR_ZONE_2*0.9;
    %             Z_2H=Z_2H+.01;
    %             ii=ii+1;
    %         end
    %         if Q_2H >= QR_ZONE_2*0.9 && Q_2H <= QR_ZONE_2*1.1
    %             ii=100;
    %         end
    %         if Z_2C >= 1-Z_1;
    %             display('Exchanger is unable to condense steam')
    %             ii=100;
    %             Z_2C=1-Z_1;
    %         end
    %         %A_out(2,ii)=Z_1H
    %         ii=ii+1;
    %     end
    %
    %     %if Q_MAX_1H > Q_MAX_1C
    %     %   Z_2=Z_2C;
    %     %   Q_MAX_2=Q_MAX_2C
    %     %   Q_2=Q_2H;
    %     %else
    %     Z_2=Z_2H;
    %     Q_MAX_2=Q_MAX_2H;
    %     Q_2=Q_2H;
    %     %end
    %% ZONE 3 ZONE 3 ZONE 3 ZONE 3 ZONE 3 ZONE 3 ZONE 3 ZONE 3
    ii=1;
    if Z_1+Z_2 < 1
        Z_3C=1-(Z_1+Z_2);
        Z_3H=Z_3C;
        Z_3=Z_3C;
        %if m_dot_C*Cp_Ci*Z_3 < m_dot_H*Cp_HL % C_C < C_H
        if 2+2 == 5
            %C_min is cold side

U_3C=(1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_3H*A_H*eff_h))))*Z_3C;
            Cr_ZONE_3C=C_3C*Z_3C/(C_3H);
            NTU_ZONE_3C=U_3C*A_C/C_3C;
            e_ZONE_3C=1-exp((1/Cr_ZONE_3C)*((NTU_ZONE_3C)^.22)*(exp(-
Cr_ZONE_3C*(NTU_ZONE_3C)^.78)-1));
            Q_3C=Q_MAX_3C*e_ZONE_3C;
            Z_3=Z_3C;
            Q_MAX_3=Q_MAX_3C
            Q_3=Q_3C;
        else
            %C_min is hot side
            ii=1;

U_3H=(1/((1/(h_C*(A_C/A_H)))+(h_F*A_H)+(1/(h_3H*A_H*eff_h))))*Z_3H;
            Cr_ZONE_3H=C_3H/C_3C*Z_3H;
            NTU_ZONE_3H=U_3H*A_C*Z_3H/(C_3H);
            e_ZONE_3H=1-exp((1/Cr_ZONE_3H)*((NTU_ZONE_3H)^.22)*(exp(-
Cr_ZONE_3H*(NTU_ZONE_3H)^.78)-1));
            Q_3H=Q_MAX_3H*e_ZONE_3H;
            Z_3=Z_3H;
            Q_MAX_3=Q_MAX_3H
            Q_3=Q_3H;
        end
    else
        Q_3=0;
        Z_3=0;
    end
    Q_3=abs(Q_3);
    %% FAN CURVE
    %Fan 2
    if i == n-1

FAN_CFM_2=start_V_C*Afr*rho_Ci*2118.88:deltavA*Afr*rho_Ci*2118.88:fina
l_V_C*Afr*rho_Ci*2118.88;
    FAN_Presure_2=-6E-12*FAN_CFM_2.^3+6E-08*FAN_CFM_2.^2-
0.0003*FAN_CFM_2+1.2706
    end
    %% Final calculations
    Q=Q_1+Q_2+Q_3;
    T_Co=T_Ci+Q/(m_dot_C*.5*(Cp_Co+Cp_Ci));
    T_Ho=T_Hi-(Q_3/C_3H);
    %% ENDING MISILANIOUS FUNCTIONS FOR OPTIMIZATION LOOP
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %out=(velocity though
    %exchanger(m/s:f/s),massflowrate(kg/s:lbm/s),pressure
drop(pascal:inches of water), )
```

```
    out(1,i)=v_Ci;%velocity at exchanger:meters/sec
    out(2,i)=v_Ci*3.2808399;%velocity at exchanger:feet/sec
    out(3,i)=m_dot_C;
    out(4,i)=v_Ci*Afr*rho_Ci*2118.88%Mass flow rate
    out(5,i)=deltaPA;%Pressure drop in pascals
    out(6,i)=deltaPA*0.004014742;%Pressure drop in inches of water
    out(7,i)=Q/1000;%k,watts
    out(8,i)=T_Co;
    out(9,i)=T_Ho;
    out(10,i)=Q_1/1000;
    out(11,i)=Q_2/1000;
    out(12,i)=Q_3/1000;
    out(13,i)=Q/1000;
    out(14,i)=H12*0.004014742;
    out(15,i)=H23*0.004014742;
    out(16,i)=H45*0.004014742;
    out(17,i)=H56*0.004014742;
    out(18,i)=H67*0.004014742;
    out(19,i)=H78*0.004014742;
    out(20,i)=Q_MAX_1H/1000;
    out(21,i)=Q_MAX_2H/1000;
    out(22,i)=Q_MAX_3H/1000;
    out(23,i)=Q_MAX_1C/1000;
    out(24,i)=Q_MAX_2C/1000;
    out(25,i)=Q_MAX_3C/1000;
    out(26,i)=Z_1;
    out(27,i)=Z_2;
    out(28,i)=Z_3;
    out(29,i)=Z_1C;
    out(30,i)=Z_1H;
    out(31,i)=-6E-12*out(4,i).^3+6E-08*out(4,i).^2-
0.0003*out(4,i)+1.2706;%Fan Curve
    out(32,i)=jH;
    out(33,i)=h_1H/1000;
    out(34,i)=h_1H/1000;
    out(35,i)=h_1H/1000;
    out(36,i)=Afr;
    out(37,i)=Aff;

    v_Ci=v_Ci+deltavA;
    v_Co=v_Ci;
    progress=(i/n)*100
    display('percent complete ')
    display(progress)
    display('%')
    i=i+1;
    if Q_3 > 0 && V_cond == 0
        V_cond=v_Ci;
        m_dot_cond=m_dot_C;
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOTTING OF DATA AND OUTPUT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%PRESSURE DROP
figure(1)
%subplot(2,2,1)
plot(out(4,:),out(6,:),'-
',out(4,:),out(14,:),out(4,:),out(15,:),out(4,:),out(16,:),out(4,:),ou
t(17,:),out(4,:),out(18,:),out(4,:),out(19,:),out(4,:),out(31,:));
hleg1 = legend('pressure drop in inches of water','incoming
bell','incoming duct','fan to exh','tubes','exit','exit bell','Fan
Curve 2');
ylabel('inches of water')
xlabel('Cubic Feet a minite')
axis([0 out(4,n)+100 0 out(31,1)+.5])



%Q EXCHANGED
figure(2)
%subplot(2,2,2)
%plot(out(1,:),out(10,:),'-',out(1,:),out(11,:),'-
',out(1,:),out(12,:),'-',out(1,:),out(13,:),'-',out(1,:),out(20,:),'--
',out(1,:),out(21,:),'--',out(1,:),out(22,:),'--
',out(1,:),out(23,:),'--',out(1,:),out(24,:),'--
',out(1,:),out(25,:),'--')
%hleg1 =
legend('Q1','Q2','Q3','QTotal','Q1maxh','Q2maxh','Q3maxh','Q1maxc','Q2
maxc','Q3maxc');
plot(out(1,:),out(10,:),'-',out(1,:),out(11,:),'-
',out(1,:),out(12,:),'-',out(1,:),out(13,:),'-')
hleg1 = legend('Q1','Q2','Q3','QTotal')

refline(0,(QR_ZONE_1+QR_ZONE_2)/1000)
refline(0,(QR_ZONE_1+QR_ZONE_2+QR_ZONE_3)/1000)

ylabel('KW of energy removal')
xlabel('speed though exchanger m^2/s')
display('V_Cond')

axis([0 out(1,n)+.5 0 (Q_MAX_1H/1000)+.5])

%TEMPETURE OUTPUT)
figure(3)
%subplot(2,2,3)
plot(out(1,:),out(8,:),'-',out(1,:),out(9,:),'-')
hleg1 = legend('T_Co','T_Ho');
ylabel('Kelvin')
xlabel('speed though exchanger m^2/s')
line([V_cond V_cond],[0 500])

% TUBE GEOMETRY
```

```
figure(4)
%subplot(2,2,4)
cla
i=1;
for i=1:2:Nl
    for ii=1:1:Nt
        rectangle('position',[rowspace*i, tubespace*ii ,.0285,
.0285],'curvature',[1,1])
    end
end
for i=2:2:Nl
    for ii=2:1:Nt
        rectangle('position',[rowspace*i, tubespace*ii-.01715 ,.0285,
.0285],'curvature',[1,1])
    end
end
axis equal

cla
figure(5)

plot(out(1,:),out(26,:),out(1,:),out(27,:),out(1,:),out(28,:))
hleg1 = legend('L_ZONE_1','L_ZONE_2','L_ZONE_3');
```