Program for Predicting the Temperature of Satellite

This program is a built around MATLAB® language. It is a result of months of relentless hard work. We will now see what it took to build the program, the theory behind it, the logic, the code and a few screen shots when the program is running.

# The theory and the code:

The temperature of the satellite is a result of heat exchange between the ambient and the satellite. The primary source of heat is the Sun which contributes the maximum heat flux, the other sources are the Albedo, which is the heat reflected from the planet Earth, and the other source is heat in the infrared range from the planet Earth.

Heat exchange occurs between the plates of the satellite by conduction and also by radiation, which we will refer as internal radiation.

The internal radiation should be considered by considering the view factors between the plates. Fundamentals of heat transfer can be found from any good textbook, thus we will spare the reader from going all over the fundamentals here again.

This version of program is capable of solving a simple model of the satellite and then solving it numerically.

We will now go through various experiences that the developer had while getting the version 1.0 Build22 of the program, correcting various parts of the code, trying out different things, getting around obstacles found around building the program. We will use the words program and program referring to the current MATLAB® program we are talking about.

# The capability of program :

This program takes various inputs which we will see quite soon. First we will see the extent of possibilities where the program can be used.

The naming of the satellite in the program: While using the program, the user has to enter the dimensions, material and surface properties of the plates, which form the structure of the satellite. The following diagram (Figure 1) shows how the satellite is named in the program, and the user has to enter the properties etc accordingly.

[Figure 1: The naming of satellite plates]

The axis of rotation is chosen through plates 2 and plate 4 because; the direction of movement is through plate 1 and plate 6, as shown by figure 2. The prime reason to chose this way is because, had the axis was chosen through plate 1 through plate 6, the satellite would retain its inertia and would not rotate around the desired path and would become disoriented after a while, as it goes round the Earth.

[Figure 2: The figure showing movement of satellite around Earth]

The satellite goes through solar flux and Albedo loading and unloading. However, it enjoys a continuous input of Earth’s Infrared radiation. The Albedo and Earth variation according to the cloud cover are neglected, because the Albedo hardly varies between 0.33 ± 0.05 and thus it is taken as 0.33 times the incoming solar radiation by default. Of course the user has the freedom to edit this value when using the program.

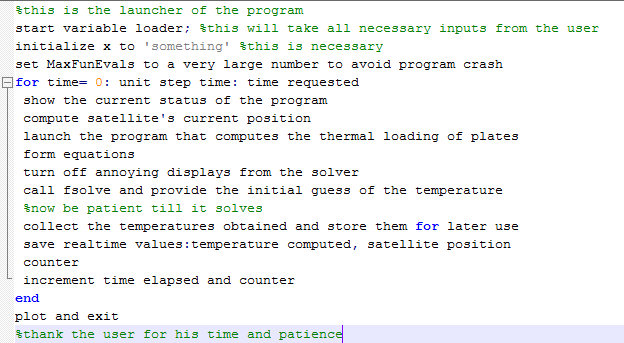
The satellite takes in the position of the satellite from where the analysis is to be begun, the orbit position by its height and the angle made with the Sun vector and the latitude position. The program will compensate for the tilt in the Earth’s axis by subtracting 23.5 degrees from the given value, converts into a positive value if the answer turned out to be negative upon subtraction. The unit of angles used in the program is degrees. Units are mentioned in which the properties are to be fed in the program while using it.

The next figure (Figure 3) shows the above explained concept.

[Figure 3: The orbital position of Satellite]

As it can be seen in the full source code of the program that can be found at the end of this document, comments have been provided in a liberal manner which should help any developer who wishes to improve the program.

# The logic behind the program:



[Figure 4: The logic behind the program]

Well, the developer assumes the above description is self explanatory and needs no further explanation.

The graph plotted will be temperature versus time from where the analysis was requested. The temperature is the temperature of the plate at the mid-section of the plate.

The program takes care of solar radiation change with days in the year and the contact resistance and the internal radiation heat exchange between the plates. However the program neglects the radiation output from the darkened surfaces that are found in the Figure 1 and also on the corresponding sides on the far side of the satellite.

<#The Source Code#>

**[LAUNCHER MODULE]: THIS LAUNCHES THE PROGRAM**

%THIS IS THE LAUNCHER FOR THE SATELLITE PROGRAM

%variables acceptor (manual/auto)

clc; label; clear all;

msgbox('PLEASE READ THE TITLE OF THE INPUT DIALOGUE BOXES YOU WIL BE SHOWN BEFORE ENTERING VALUES! (resize the window, if req)');

pause(2);

var\_loader();

close('all');

clc; label;

fprintf('\n System Paused: Hit any Key to Continue:: ');

pause;

load('f\_run\_var.mat');

warning('off','all');

%computing part

% options = optimset('Display','off');

options = optimset('Display','off','MaxFunEvals',100000');

cntr =1; t1 = s1t1; t2 = s1t2; t3 = s1t3; t4 = s1t4; t5 = s1t5; t6 = s1t6;

tot\_steps = t\_req/utime; time\_elapsed = 0;temp\_mat = zeros();

x = 'something';

MaxFunEvals = 10000;

for loop\_run = 0:utime:t\_req

clc; label;

fprintf('\n Step number : %d / %d ',cntr-1,tot\_steps);

%determine satellite pos around earth

sat\_pos\_orbit = sat\_pos\_orbit + sat\_pos\_orbit\_u;

% increment by step and determine plate orientations

sat\_pos\_axis = sat\_pos\_axis + sat\_pos\_axis\_u;

if sat\_pos\_axis >= 360 %convert to angle b/w 0 to 360

sat\_pos\_axis = rem(sat\_pos\_axis,360);

end

%defining a new angle zenith; abbr: zen

%takes care of orientation change along orbit

if sat\_pos\_orbit >= 0 && sat\_pos\_orbit <= 90

zen = sat\_pos\_orbit;

else

zen = rem(sat\_pos\_orbit,90);

end

%%% EQUATION FORMING PART

main\_mod(x);

options = optimset(options,'diagnostics','off','display','off');

fprintf('\n Please wait.. Solving equations...');

% options = optimset('Display','iter');

[x]= fsolve(@main\_mod,x0,options);

temp\_mat(1,cntr) = x(1); temp\_mat(2,cntr) = x(2); temp\_mat(3,cntr)=x(3); temp\_mat(4,cntr)=x(4); temp\_mat(5,cntr) = x(5); temp\_mat(6,cntr)=x(6);

rtym = 'rtime.mat';

save(rtym,'temp\_mat','sat\_pos\_orbit','sat\_pos\_axis','cntr')

cntr = cntr+1;time\_elapsed = time\_elapsed+utime;

clc; label;

end

%plot and exit

hold on

breadth\_tm = size(temp\_mat);

timat = linspace(0,time\_elapsed,breadth\_tm(2));

plot(timat,temp\_mat(1,:),'r');plot(timat,temp\_mat(2,:),'g');

plot(timat,temp\_mat(3,:),'c');plot(timat,temp\_mat(4,:),'b');

plot(timat,temp\_mat(5,:),'k');plot(timat,temp\_mat(6,:),'m');

title('Temperature of Plates with Time');

xlabel('Time');

ylabel('Plate Temperatures');

legend('Plate 1','Plate 2','Plate 3','Plate 4','Plate 5','Plate 6',2);

set(gcf,'toolbar','none','menubar','none');

imshow('thank\_you\_for\_your\_time.png');

**[VARIABLE LOADER]: THIS IS WHAT LOADS THE VARIABLES**

%var\_loader: LOADS VARIABLES INTO THE WORKSPACE

options.Resize='on';

options.WindowStyle='normal';

options.Interpreter='tex';

fprintf('\nLoad satellite parameters...Launch relevant input dialogues');

set(gcf,'toolbar','none','menubar','none');

imshow('sate\_image.jpg');

%PLATE 1

plate1\_prompt ={'LENGTH(mm)::','BREADTH(mm)','THICKNESS(mm)',...

'ABSORPTIVITY(internal)','EMISSIVITY(internal)','ABSORPTIVITY(external)','EMISSIVITY(external)'...

'DENSITY(kg/m3)','SPECIFIIC HEAT (Cp)(kJ/kg K)','THERMAL CONDUCTIVITY(W/mK)','INITIAL TEMP(K)'};

dlg\_title = '::PLATE 1::';

num\_lines = 1;

%aluminium surface property is given 0.8 because of prescence of solar cell

%internal surf prop due to normally black coating on the inside

def\_ans = {'100','100','6','0.8','0.8','0.8','0.8','2700','0.91','167','285'};

ans1 = str2num(char(inputdlg(plate1\_prompt,dlg\_title,num\_lines,def\_ans,options)));

fprintf('\nPlate 1 done..');

%PLATE 2

plate2\_prompt ={'LENGTH(mm)::','BREADTH(mm)','THICKNESS(mm)',...

'ABSORPTIVITY(internal)','EMISSIVITY(internal)','ABSORPTIVITY(external)','EMISSIVITY(external)'...

'DENSITY(kg/m3)','SPECIFIIC HEAT (Cp)(kJ/kg K)','THERMAL CONDUCTIVITY(W/mK)','INITIAL TEMP(K)'};

dlg\_title = '::PLATE 2::';

num\_lines = 1;

%aluminium surface property is given 0.8 because of prescence of solar cell

%internal surf prop due to normally black coating on the inside

def\_ans = {'200','100','5','0.8','0.8','0.8','0.8','2700','0.91','167','285'};

ans2 = str2num(char(inputdlg(plate2\_prompt,dlg\_title,num\_lines,def\_ans,options)));

fprintf('\nPlate 2 done..');

%PLATE 3

plate3\_prompt ={'LENGTH(mm)::','BREADTH(mm)','THICKNESS(mm)',...

'ABSORPTIVITY(internal)','EMISSIVITY(internal)','ABSORPTIVITY(external)','EMISSIVITY(external)'...

'DENSITY(kg/m3)','SPECIFIIC HEAT (Cp)(kJ/kg K)','THERMAL CONDUCTIVITY(W/mK)','INITIAL TEMP(K)'};

dlg\_title = '::PLATE 3::';

num\_lines = 1;

%aluminium surface property is given 0.8 because of prescence of solar cell

%internal surf prop due to normally black coating on the inside

def\_ans = {'200','100','5','0.8','0.8','0.8','0.8','2700','0.91','167','285'};

ans3 = str2num(char(inputdlg(plate3\_prompt,dlg\_title,num\_lines,def\_ans,options)));

fprintf('\nPlate 3 done..');

%PLATE 4

plate4\_prompt ={'LENGTH(mm)::','BREADTH(mm)','THICKNESS(mm)',...

'ABSORPTIVITY(internal)','EMISSIVITY(internal)','ABSORPTIVITY(external)','EMISSIVITY(external)'...

'DENSITY(kg/m3)','SPECIFIIC HEAT (Cp)(kJ/kg K)','THERMAL CONDUCTIVITY(W/mK)','INITIAL TEMP(K)'};

dlg\_title = '::PLATE 4::';

num\_lines = 1;

%aluminium surface property is given 0.8 because of prescence of solar cell

%internal surf prop due to normally black coating on the inside

def\_ans = {'200','100','5','0.8','0.8','0.8','0.8','2700','0.91','167','285'};

ans4 = str2num(char(inputdlg(plate4\_prompt,dlg\_title,num\_lines,def\_ans,options)));

fprintf('\nPlate 4 done..');

%PLATE 5

plate5\_prompt ={'LENGTH(mm)::','BREADTH(mm)','THICKNESS(mm)',...

'ABSORPTIVITY(internal)','EMISSIVITY(internal)','ABSORPTIVITY(external)','EMISSIVITY(external)'...

'DENSITY(kg/m3)','SPECIFIIC HEAT (Cp)(kJ/kg K)','THERMAL CONDUCTIVITY(W/mK)','INITIAL TEMP(K)'};

dlg\_title = '::PLATE 5::';

num\_lines = 1;

%aluminium surface property is given 0.8 because of prescence of solar cell

%internal surf prop due to normally black coating on the inside

def\_ans = {'200','100','5','0.8','0.8','0.8','0.8','2700','0.91','167','285'};

ans5 = str2num(char(inputdlg(plate5\_prompt,dlg\_title,num\_lines,def\_ans,options)));

fprintf('\nPlate 5 done..');

%PLATE 6

plate6\_prompt ={'LENGTH(mm)::','BREADTH(mm)','THICKNESS(mm)',...

'ABSORPTIVITY(internal)','EMISSIVITY(internal)','ABSORPTIVITY(external)','EMISSIVITY(external)'...

'DENSITY(kg/m3)','SPECIFIIC HEAT (Cp)(kJ/kg K)','THERMAL CONDUCTIVITY(W/mK)','INITIAL TEMP(K)'};

dlg\_title = '::PLATE 6::';

num\_lines = 1;

%aluminium surface property is given 0.8 because of prescence of solar cell

%internal surf prop due to normally black coating on the inside

def\_ans = {'100','100','6','0.8','0.8','0.8','0.8','2700','0.91','167','285'};

ans6 = str2num(char(inputdlg(plate6\_prompt,dlg\_title,num\_lines,def\_ans,options)));

fprintf('\nPlate 6 done..');

%ASSIGN ANSWER 1 - 6 TO THE RESPECTIVE VARIABLES

l1 = ans1(1); b1 = ans1(2); t1 = ans1(3); s2a1 = ans1(4); s2e1 = ans1(5); s1a1 = ans1(6); s1e1 = ans1(7); density1 = ans1(8)\*1e-9; cp1 = ans1(9)\*1e3; k1 = ans1(10)\*1e-3; s1t1 = ans1(11); s2t1 = ans1(11);

l2 = ans2(1); b2 = ans2(2); t2 = ans2(3); s2a2 = ans2(4); s2e2 = ans2(5); s1a2 = ans2(6); s1e2 = ans2(7); density2 = ans2(8)\*1e-9; cp2 = ans2(9)\*1e3; k2 = ans2(10)\*1e-3; s1t2 = ans2(11); s2t2 = ans2(11);

l3 = ans3(1); b3 = ans3(2); t3 = ans3(3); s2a3 = ans3(4); s2e3 = ans3(5); s1a3 = ans3(6); s1e3 = ans3(7); density3 = ans3(8)\*1e-9; cp3 = ans3(9)\*1e3; k3 = ans3(10)\*1e-3; s1t3 = ans3(11); s2t3 = ans3(11);

l4 = ans4(1); b4 = ans4(2); t4 = ans4(3); s2a4 = ans4(4); s2e4 = ans4(5); s1a4 = ans4(6); s1e4 = ans4(7); density4 = ans4(8)\*1e-9; cp4 = ans4(9)\*1e3; k4 = ans4(10)\*1e-3; s1t4 = ans4(11); s2t4 = ans4(11);

l5 = ans5(1); b5 = ans5(2); t5 = ans5(3); s2a5 = ans5(4); s2e5 = ans5(5); s1a5 = ans5(6); s1e5 = ans5(7); density5 = ans5(8)\*1e-9; cp5 = ans5(9)\*1e3; k5 = ans5(10)\*1e-3; s1t5 = ans5(11); s2t5 = ans5(11);

l6 = ans6(1); b6 = ans6(2); t6 = ans6(3); s2a6 = ans6(4); s2e6 = ans6(5); s1a6 = ans6(6); s1e6 = ans6(7); density6 = ans6(8)\*1e-9; cp6 = ans6(9)\*1e3; k6 = ans6(10)\*1e-3; s1t6 = ans6(11); s2t6 = ans6(11);

a1 = l1\*b1; a2 = l2\*b2; a3 = l3\*b3; a4 = l4\*b4; a5 = l5\*b5; a6 = l6\*b6;

stf\_blt = 5.67e-2;

fprintf('\n Calculate view factors between parallel plates...');

vf16=vf\_1pll6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f35=vf\_3pll5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f24=vf\_2pll4(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

fprintf('\n Loaded parallel faces view factors...');

fprintf('\n Calculating view factors between perpendicular plates...');

vf12=vf\_1pa2(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

vf13=vf\_1pa3(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

vf14=vf\_1pa4(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

vf15=vf\_1pa5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f23=vf\_2pa3(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f25=vf\_2pa5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f26=vf\_2pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f34=vf\_3pa4(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f36=vf\_3pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f45=vf\_4pa5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f46=vf\_4pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

f56=vf\_5pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6);

%internal areas

ia1=(l1-t3-t5)\*(b1-t2-t4);

ia2=l2\*(b2-t5-t3);

ia3=l3\*(b3-t2-t4);

ia4=l4\*(b4-t5-t3);

ia5=l5\*(b5-t2-t4);

ia6=(l6-t3-t5)\*(b6-t2-t4);

%reciprocity relations

vf11=0;f21=(ia1/ia2)\*vf12; f22=0; f33=0; f44=0; f55=0; f66=0;f31=ia1/ia3\*vf13; f32=ia2/ia3\*f23;f43=ia3/ia4\*f34;f54=ia4/ia5\*f45;

f41=ia1/ia4\*vf14; f42=ia2/ia4\*f24;f53=ia3/ia5\*f35;f65=ia5/ia6\*f56;f51=ia1/ia5\*vf15; f52=ia2/ia5\*f25;f63=ia3/ia6\*f36;f61=ia1/ia6\*vf16; f62=ia2/ia6\*f26; f64 = ia4/ia6\*f46;

fprintf('\n Loaded perpendicular faces view factors...');

notifyme = true;

% imshow('sate\_image.jpg');

fprintf('\n Will clear command window in 5 seconds');

pause(2);

clc; label;

while notifyme

fprintf('\n ::General Notifications:: \n 1. The radiation loads that you will provide next will be directly used with suitable cosine components. \n consider any conversions etc (if required)');

fprintf('\n 2. The program will deduct 23.5d from satellite lattitude pos you provide to compensate earth"s axis tilt');

fprintf('\n 3. Provide reasonable time step to ensure quick but accurate computations');

fprintf('\n 4. When asked for, provide a crude value for lowest temperature that you will say satellite may reach (default is 100K)');

fprintf('\n 5. Axis of rotation::\n A. plate 2 through plate 4 (default-ONLY OPTION)\n B. plate 1 through 6 <NEXT VERSION>\n C. plate 5 through 3 <NEXT VERSION>\n');

notifyme = false;

end

fprintf('\n Load Mission Parameters...Launching input dialog');

mission\_prompt = {'Ambient Temperature(K)','Launch Day(jan1=1,jan31=31,feb1=32...)','Earth Radiation Load (IR)','Albedo Factor','Orbit Height(km)','Orbit Orientation Angle(d)','Satellite Pos (Lat(d))','Time required for one rotation(s)'};

def\_ans = {'5','45','237','0.33','350','22.5','23.5','15'};

dlg\_title = 'Mission Parameters';

ans\_msn = str2num(char(inputdlg(mission\_prompt,dlg\_title,num\_lines,def\_ans,options)));

amb = ans\_msn(1); day1 = ans\_msn(2); erad = ans\_msn(3)\*1e-6; alb\_f = ans\_msn(4); o\_ht = ans\_msn(5); Be = ans\_msn(6); sat\_pos\_orbit\_earth = ans\_msn(7);time\_1\_rot= ans\_msn(8);

time\_elapsed = 0;

sat\_pos\_axis = 0;

%SATELLITE POSITION WRTO EARTH

sat\_pos\_orbit = sat\_pos\_orbit\_earth -23.5;

if sat\_pos\_orbit < 0

sat\_pos\_orbit = sat\_pos\_orbit + 360; %find the actual relevant satellite position

end

ref\_pos = sat\_pos\_orbit;

clc; label;

fprintf('\n Loading other necessary data...');

cr\_prompt = {'Contact Resistance(m2.K/W) btn 1 & 2(= 1 to 4):','between 1 and 3 (= 1 to 5):','between 2 and 3 (= 2 to 5): '};

def\_ans = {'5','5','5'};

dlg\_title = '::Contact Resistances::';

ans\_cr = str2num(char(inputdlg(cr\_prompt,dlg\_title,num\_lines,def\_ans,options)));

cr\_p1top2 = ans\_cr(1)\*1e10;

cr\_p1top3 = ans\_cr(2)\*1e10;

cr\_p2top3 = ans\_cr(3)\*1e10;

cr\_p2top5 = cr\_p2top3;

cr\_p4top5 = cr\_p2top3;

cr\_p6top3 = cr\_p1top3;

cr\_p6top2 = cr\_p1top2;

cr\_p1top4 = cr\_p1top3;

cr\_p3top4 = cr\_p1top4;

cr\_p1top5 = cr\_p1top3;

cr\_p6top4 = cr\_p1top4;

cr\_p6top5 = cr\_p1top5;

cr\_p1top5 = cr\_p1top3;

calc\_prompt = {'Unit Time Step:','Analysis required for (seconds)','% of radiation @ poles wrto equator','(crude)Worst Cold Case Temp(K)'};

def\_ans = {'1','100','30','100'};

dlg\_title = 'Calculation Parameters';

ans\_cal = str2num(char(inputdlg(calc\_prompt,dlg\_title,num\_lines,def\_ans,options)));

utime = ans\_cal(1); t\_req = ans\_cal(2);y1 = ans\_cal(3); x0 = ans\_cal(4);

x0 = [x0 x0 x0 x0 x0 x0];

o\_tym=sqrt(4\*(pi\*pi)\*(((6378.1+o\_ht)\*1000)^3)/(3.986\*(10^14))); %orbit time calculated is in seconds

b = 2\*pi\*day1/365;

sat\_pos\_axis\_u = (utime \*360)/(time\_1\_rot); %step in which angles change wrt step time

sat\_pos\_orbit\_u = (utime\*360)/(o\_tym);

m1 = l1\*b1\*t1\*density1;m2 = l2\*b2\*t2\*density2;m3 = l3\*b3\*t3\*density3;m4 = l4\*b4\*t4\*density4;m5 = l5\*b5\*t5\*density5;m6 = l6\*b6\*t6\*density6;

%defining zones

%sangle = shadow angle

sangle = acosd(6378.1/(6378.1+o\_ht));

cntr = 1;

temp\_mat(1,cntr) = s1t1;temp\_mat(2,cntr) = s1t2;temp\_mat(3,cntr) = s1t3;temp\_mat(4,cntr) = s1t4;temp\_mat(5,cntr) = s1t5;temp\_mat(6,cntr) = s1t6;

% fprintf('\nthis is x0 created');

% disp(x0);

save('rtime.mat','temp\_mat','sat\_pos\_orbit','sat\_pos\_axis','cntr')

save('f\_run\_var.mat') %save first run variables to be used later if reqd

**[MAIN MODULE: MAIN\_MOD] THIS IS WHERE CALCULATIONS ARE DONE**

function F=main\_mod(x)

if ischar(x)

clear x

syms s1t s2t s3t s4t s5t s6t %kick start the program

else

s1t = x(1);

s2t = x(2);

s3t = x(3);

s4t = x(4);

s5t = x(5);

s6t = x(6);

end

%THIS IS THE COMPUTATION CORE OF THE PROGRAM

%BEGIN COMPUTATIONS NOW

load('f\_run\_var.mat'); %loads all variables

%rtime.mat must be created inside the for loop which maintains iterations.

load('rtime.mat'); %rtime: real time:: day, orbital pos, temperatures!!THIS WILL REWRITE SOME OLD DATA FROM F\_RUN\_VAR

c1 = temp\_mat(1,cntr);

c2 = temp\_mat(2,cntr);

c3 = temp\_mat(3,cntr);

c4 = temp\_mat(4,cntr);

c5 = temp\_mat(5,cntr);

c6 = temp\_mat(6,cntr);

%load computers!

%solar loads

if sat\_pos\_orbit >= 0 && sat\_pos\_orbit <= 90

s = (1367e-6)\*(1.00011+0.034221\*cosd(2\*b))\*(((y1-100)/90)\*sat\_pos\_orbit+100)/100; %y1 is generally a number 0-100

elseif sat\_pos\_orbit > 90 && sat\_pos\_orbit <= 270

s = (1367e-6)\*(1.00011+0.034221\*cosd(2\*b))\* y1/100;

elseif sat\_pos\_orbit >270 && sat\_pos\_orbit < 360

s = (1367e-6)\*(1.00011+0.034221\*cosd(2\*b))\*(-((y1-100)/90)\*(sat\_pos\_orbit-180)+100)/100; %y1 is generally a number 0-100

else

s = 0;

end

solr = s\*cosd(sat\_pos\_axis)\*cos(Be); %solar load has been taken care in all zones

if sat\_pos\_axis > 90 && sat\_pos\_axis <270

solr = 0;

end

%albedo direct load

alb\_dir = alb\_f\*s\*cosd(sat\_pos\_axis);

if sat\_pos\_axis >0 && sat\_pos\_axis <90 || sat\_pos\_axis >270 && sat\_pos\_axis < 360

alb\_dir = 0;

end

%albedo diffuse load

alb\_diff = (30/100)\*(alb\_f\*s)\*sind(sat\_pos\_axis);

%earth direct load

earth\_dir = erad\*cosd(sat\_pos\_axis);

if sat\_pos\_axis >0 && sat\_pos\_axis <90 || sat\_pos\_axis >270 && sat\_pos\_axis < 360

earth\_dir = 0;

end

%earth diffuse load

earth\_diff = (30/100)\*erad\*sind(sat\_pos\_axis);

%total loads

tot\_l=solr+alb\_diff+alb\_dir+earth\_diff+earth\_dir; %just multiply this thing for all plates

%

q\_1\_in\_tot = a1\*s1a1\*tot\_l;

q\_3\_in\_tot = a3\*s1a3\*tot\_l;

q\_5\_in\_tot = a5\*s1a5\*tot\_l;

q\_6\_in\_tot = a6\*s1a6\*tot\_l;

%because 2 and 4 are the axis, they need special treatment

solr\_axs = sind(Be)\*s;

tot\_axs = solr\_axs+alb\_diff+alb\_dir+earth\_diff+earth\_dir; %just multiply this thing for all plates

q\_2\_in\_tot = a2\*s1a2\*tot\_axs;

tot\_axs = solr\_axs+alb\_diff+alb\_dir+earth\_diff+earth\_dir; %just multiply this thing for all plates

q\_4\_in\_tot = a4\*s1a4\*tot\_axs;

if Be >= 0 && Be < 90 || Be >180 && Be < 270 %this will reset the value according to position of orbit

solr\_axs = 0;

tot\_axs = solr\_axs+alb\_diff+alb\_dir+earth\_diff+earth\_dir; %just multiply this thing for all plates

q\_2\_in\_tot = a2\*s1a2\*tot\_axs;

else

solr\_axs = 0;

tot\_axs = solr\_axs+alb\_diff+alb\_dir+earth\_diff+earth\_dir; %just multiply this thing for all plates

q\_4\_in\_tot = a4\*s1a4\*tot\_axs;

end

q\_1\_out\_rad = s1e1\*stf\_blt\*a1\*((s1t-m1\*cp1\*(s1t-c1))^4 - (amb).^4);

%special nomenclature one plate to any other plates by conduction or radiation

%then q\_1to2\_c = heat 1 to 2 by conduction replace c by r for radiation

q\_1to2\_c = (s1t-s2t)/(cr\_p1top2\*b2\*t2);

q\_1to3\_c = (s1t-s3t)/(cr\_p1top3\*b3\*t3);

q\_1to4\_c = (s1t-s4t)/(cr\_p1top4\*b4\*t4);

q\_1to5\_c = (s1t-s5t)/(cr\_p1top5\*b5\*t5);

q\_1toall\_c= q\_1to2\_c+q\_1to3\_c+q\_1to4\_c+q\_1to5\_c;

%radiation loads from plate 1 to other plates interior surfaces

%s2t1 = s1t + m\*cp\*(s1t-c1)\*l/(2\*k\*a1);

q\_1to2\_r = vf12\*stf\_blt\*s2e1\*((s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4-(s2t + m2\*cp6\*(s2t-c2)\*t2/(2\*k2\*a2))^4);

q\_1to3\_r = vf13\*stf\_blt\*s2e1\*((s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4-(s3t + m3\*cp6\*(s3t-c3)\*t3/(2\*k3\*a3))^4);

q\_1to4\_r = vf14\*stf\_blt\*s2e1\*((s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4-(s4t + m4\*cp6\*(s4t-c4)\*t4/(2\*k4\*a4))^4);

q\_1to5\_r = vf15\*stf\_blt\*s2e1\*((s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4-(s5t + m5\*cp6\*(s5t-c5)\*t5/(2\*k5\*a5))^4);

q\_1to6\_r = vf15\*stf\_blt\*s2e1\*((s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4-(s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4);

q\_1toall\_r=q\_1to6\_r+q\_1to2\_r+q\_1to3\_r+q\_1to4\_r+q\_1to5\_r;

% q\_net\_1= q\_1\_in\_tot-q\_1\_out\_rad-q\_1toall\_c-q\_1toall\_r;

%heat transfer from plate 2 to other plates:

%fprintf('\nEquation formed for plate 1...');

%fprintf('\nBuilding equation for plate 2...');

%heat transfer by conduction

q\_2\_out\_rad = s1e2\*stf\_blt\*a2\*((s2t-m2\*cp2\*(s2t-c2))^4 - (amb).^4);

%special nomenclature one plate to any other plates by conduction or radiation

%then q\_2to2\_c = heat 2 to 2 by conduction replace c by r for radiation

q\_2to1\_c = (s2t-s1t)/(cr\_p1top2\*b2\*t2);

q\_2to3\_c = (s2t-s3t)/(cr\_p2top3\*l2\*t2);

q\_2to5\_c = (s2t-s5t)/(cr\_p2top3\*l5\*t5);

q\_2to6\_c = (s2t-s6t)/(cr\_p6top2\*b6\*t6);

q\_2toall\_c= q\_2to1\_c+q\_2to3\_c+q\_2to5\_c+q\_2to6\_c;

%radiation loads from plate 2 to other plates interior surfaces

q\_2to1\_r = f21\*stf\_blt\*s2e2\*((s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4-(s2t + m2\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4);

q\_2to3\_r = f23\*stf\_blt\*s2e2\*((s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4-(s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4);

q\_2to4\_r = f24\*stf\_blt\*s2e2\*((s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4-(s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4);

q\_2to5\_r = f25\*stf\_blt\*s2e2\*((s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4-(s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4);

q\_2to6\_r = f25\*stf\_blt\*s2e2\*((s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4-(s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4);

q\_2toall\_r=q\_2to6\_r+q\_2to1\_r+q\_2to3\_r+q\_2to4\_r+q\_2to5\_r;

% q\_net\_2= q\_2\_in\_tot-q\_2\_out\_rad-q\_2toall\_c-q\_2toall\_r;

%fprintf('\nEquation formed for plate 2...');

%fprintf('\nBuilding equation for plate 3...');

q\_3\_out\_rad = s1e3\*stf\_blt\*a3\*((s3t-m3\*cp3\*(s3t-c3))^4 - (amb).^4);

%special nomenclature one plate to any other plates by conduction or radiation

%then q\_3to2\_c = heat 3 to 2 by conduction replace c by r for radiation

q\_3to2\_c = (s3t-s2t)/(cr\_p2top3\*l2\*t2);

q\_3to1\_c = (s3t-s3t)/(cr\_p1top3\*b3\*t3);

q\_3to4\_c = (s3t-s4t)/(cr\_p3top4\*l4\*t4);

q\_3to6\_c = (s3t-s5t)/(cr\_p6top3\*b3\*t3);

q\_3toall\_c= q\_3to2\_c+q\_3to1\_c+q\_3to4\_c+q\_3to6\_c;

%radiation loads from plate 3 to other plates interior surfaces

%s2t3 = s3t + m\*cp\*(s3t-c3)\*l/(2\*k\*a3);

q\_3to2\_r = f32\*stf\_blt\*s2e3\*((s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4-(s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4);

q\_3to1\_r = f31\*stf\_blt\*s2e3\*((s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4-(s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4);

q\_3to4\_r = f34\*stf\_blt\*s2e3\*((s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4-(s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4);

q\_3to5\_r = f35\*stf\_blt\*s2e3\*((s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4-(s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4);

q\_3to6\_r = f35\*stf\_blt\*s2e3\*((s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4-(s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4);

q\_3toall\_r=q\_3to6\_r+q\_3to2\_r+q\_3to1\_r+q\_3to4\_r+q\_3to5\_r;

% q\_net\_3= q\_3\_in\_tot-q\_3\_out\_rad-q\_3toall\_c-q\_3toall\_r;

%fprintf('\nEquation formed for plate 3...');

%fprintf('\nBuilding equation for plate 4...');

q\_4\_out\_rad = s1e4\*stf\_blt\*a4\*((s4t-m4\*cp4\*(s4t-c4))^4 - (amb).^4);

%special nomenclature one plate to any other plates by conduction or radiation

%then q\_4to2\_c = heat 4 to 2 by conduction replace c by r for radiation

q\_4to1\_c = (s4t-s2t)/(cr\_p1top4\*b4\*t4);

q\_4to3\_c = (s4t-s3t)/(cr\_p3top4\*l4\*t4);

q\_4to6\_c = (s4t-s4t)/(cr\_p6top4\*b4\*t4);

q\_4to5\_c = (s4t-s5t)/(cr\_p4top5\*l4\*t4);

q\_4toall\_c= q\_4to1\_c+q\_4to3\_c+q\_4to6\_c+q\_4to5\_c;

%radiation loads from plate 4 to other plates interior surfaces

q\_4to2\_r = f42\*stf\_blt\*s2e4\*((s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4-(s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4);

q\_4to3\_r = f43\*stf\_blt\*s2e4\*((s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4-(s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4);

q\_4to1\_r = f41\*stf\_blt\*s2e4\*((s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4-(s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4);

q\_4to5\_r = f45\*stf\_blt\*s2e4\*((s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4-(s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4);

q\_4to6\_r = f45\*stf\_blt\*s2e4\*((s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4-(s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4);

q\_4toall\_r=q\_4to6\_r+q\_4to2\_r+q\_4to3\_r+q\_4to1\_r+q\_4to5\_r;

% q\_net\_4= q\_4\_in\_tot-q\_4\_out\_rad-q\_4toall\_c-q\_4toall\_r;

%fprintf('\nEquation formed for plate 4...');

%fprintf('\nBuilding equation for plate 5...');

q\_5\_out\_rad = s1e5\*stf\_blt\*a5\*((s5t-m5\*cp5\*(s5t-c5))^4 - (amb).^4);

%special nomenclature one plate to any other plates by conduction or radiation

%then q\_5to2\_c = heat 5 to 2 by conduction replace c by r for radiation

q\_5to2\_c = (s5t-s2t)/(cr\_p2top5\*b2\*t2);

q\_5to1\_c = (s5t-s3t)/(cr\_p1top5\*b5\*t5);

q\_5to4\_c = (s5t-s4t)/(cr\_p4top5\*l4\*t4);

q\_5to6\_c = (s5t-s5t)/(cr\_p6top5\*b5\*t5);

q\_5toall\_c= q\_5to2\_c+q\_5to1\_c+q\_5to4\_c+q\_5to6\_c;

%radiation loads from plate 5 to other plates interior surfaces

q\_5to2\_r = f52\*stf\_blt\*s2e5\*((s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4-(s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4);

q\_5to3\_r = f53\*stf\_blt\*s2e5\*((s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4-(s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4);

q\_5to4\_r = f54\*stf\_blt\*s2e5\*((s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4-(s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4);

q\_5to1\_r = f55\*stf\_blt\*s2e5\*((s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4-(s1t + m1\*cp6\*(s5t-c5)\*t5/(2\*k1\*a1))^4);

q\_5to6\_r = f55\*stf\_blt\*s2e5\*((s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4-(s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4);

q\_5toall\_r=q\_5to6\_r+q\_5to2\_r+q\_5to3\_r+q\_5to4\_r+q\_5to1\_r;

% q\_net\_5= q\_5\_in\_tot-q\_5\_out\_rad-q\_5toall\_c-q\_5toall\_r;

%fprintf('\nEquation formed for plate 5...');

%fprintf('\nBuilding equation for plate 6...');

q\_6\_out\_rad = s1e6\*stf\_blt\*a6\*((s6t-m6\*cp6\*(s6t-c6))^4 - (amb).^4);

%special nomenclature one plate to any other plates by conduction or radiation

%then q\_6to2\_c = heat 6 to 2 by conduction replace c by r for radiation

q\_6to2\_c = (s6t-s2t)/(cr\_p6top2\*b2\*t2);

q\_6to3\_c = (s6t-s3t)/(cr\_p6top3\*b3\*t3);

q\_6to4\_c = (s6t-s4t)/(cr\_p6top4\*b4\*t4);

q\_6to5\_c = (s6t-s5t)/(cr\_p6top5\*b5\*t5);

q\_6toall\_c= q\_6to2\_c+q\_6to3\_c+q\_6to4\_c+q\_6to5\_c;

%radiation loads from plate 6 to other plates interior surfaces

%s2t6 = s6t + m\*cp\*(s6t-c6)\*l/(2\*k\*a6);

q\_6to2\_r = f62\*stf\_blt\*s2e6\*((s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4-(s2t + m2\*cp2\*(s2t-c2)\*t2/(2\*k2\*a2))^4);

q\_6to3\_r = f63\*stf\_blt\*s2e6\*((s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4-(s3t + m3\*cp3\*(s3t-c3)\*t3/(2\*k3\*a3))^4);

q\_6to4\_r = f64\*stf\_blt\*s2e6\*((s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4-(s4t + m4\*cp4\*(s4t-c4)\*t4/(2\*k4\*a4))^4);

q\_6to5\_r = f65\*stf\_blt\*s2e6\*((s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4-(s5t + m5\*cp5\*(s5t-c5)\*t5/(2\*k5\*a5))^4);

q\_6to1\_r = f61\*stf\_blt\*s2e6\*((s6t + m6\*cp6\*(s6t-c6)\*t6/(2\*k6\*a6))^4-(s1t + m1\*cp1\*(s1t-c1)\*t1/(2\*k1\*a1))^4);

q\_6toall\_r=q\_6to1\_r+q\_6to2\_r+q\_6to3\_r+q\_6to4\_r+q\_6to5\_r;

% q\_net\_6= q\_6\_in\_tot-q\_6\_out\_rad-q\_6toall\_c-q\_6toall\_r;

F = [q\_1\_in\_tot-q\_1\_out\_rad-q\_1toall\_c-q\_1toall\_r;...

q\_2\_in\_tot-q\_2\_out\_rad-q\_2toall\_c-q\_2toall\_r;...

q\_3\_in\_tot-q\_3\_out\_rad-q\_3toall\_c-q\_3toall\_r;...

q\_4\_in\_tot-q\_4\_out\_rad-q\_4toall\_c-q\_4toall\_r;...

q\_5\_in\_tot-q\_5\_out\_rad-q\_5toall\_c-q\_5toall\_r;...

q\_6\_in\_tot-q\_6\_out\_rad-q\_6toall\_c-q\_6toall\_r];

**SUPPORTING CODE FILES:**

**VIEW FACTOR CALCULATORS:**

**View factor between plate 5 and 6:**

function f56=vf\_5pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 5 and 6

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=l5;

L=b5-t2-t4;

W=b6-t3-t5;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f56=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f56 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f56');

f56 = 0 - f56;

end

**View factor between 4 and 6:**

function f46=vf\_4pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 4 and 5

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=l4;

L=b4;

W=b6-t2-t4;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f46=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f46 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f46');

f46 = 0 - f46;

end

**View factor between 4 and 5:**

function f45=vf\_4pa5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 4 and 5

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=l1-t2-t5;

L=b5;

W=b5-t2-t4;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f45=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f45 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f45');

f45 = 0 - f45;

end

**View factors between 3 and 5:**

function f35=vf\_3pll5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%INTAKE VARIABLES ARE DECLARED GLOBAL THEY MUST BE AVALABLE EVERYWHERE

%WHEREEVER THE CONTROL FLOWS

%view factor for parallel rectangular identical plates

%nomenclature: vf\_3pll5 read as view factor for surface 3 parallel to

%surface 5

%the view factor is in 3 cases 1||6 and 3||5 and 2||4

w1=(l3-t1-t6);

w2=(b2-t2-t4);

H=(b2-t3-t5);

x=w1/H;

y=w2/H;

x1=sqrt(1+x.^2);

y1=sqrt(1+y.^2);

f35=(1/(pi\*x\*y))\*(log((x1.^2)/(x1.^2+y1.^2-1))+2\*x\*(y1\*atan(x/y1)-atan(x))+2\*y\*(x1\*atan(y/x1)-atan(y)));

if f35 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f35');

f35 = 0 - f35;

end

**View Factor between 3 and 6:**

function f36=vf\_3pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 3 and 6

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=l3;

L=b3-t2-t4;

W=l6-t5-t3;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f36=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f36 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f36');

f36 = 0 - f36;

end

**View factor between 3 and 4:**

function f34=vf\_3pa4(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 3 and 4

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=b3-t4-t2;

L=l3;

W=l1-t4-t5;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f34=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f34 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f15');

f34 = 0 - f15;

end

**View factor between 2 and 4:**

function f24=vf\_2pll4(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%INTAKE VARIABLES ARE DECLARED GLOBAL THEY MUST BE AVALABLE EVERYWHERE

%WHEREEVER THE CONTROL FLOWS

%view factor for parallel rectangular identical plates

%nomenclature: vf\_2pll4 read as view factor for surface 2 parallel to

%surface 4

%the view factor is in 3 cases 1||6 and 3||5 and 2||4

w1=(l2-t1-t6);

w2=(b2-t3-t5);

H=(b3-t2-t4);

x=w1/H;

y=w2/H;

x1=sqrt(1+x.^2);

y1=sqrt(1+y.^2);

f24=(1/(pi\*x\*y))\*(log((x1.^2)/(x1.^2+y1.^2-1))+2\*x\*(y1\*atan(x/y1)-atan(x))+2\*y\*(x1\*atan(y/x1)-atan(y)));

if f24 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f24');

f24 = 0 - f24;

end

**View factor between 2 and 6:**

function f26=vf\_2pa6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 2 and 6

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=b6-t2-t4;

L=l6-t4-t5;

W=l2;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f26=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f26 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f26');

f26 = 0 - f26;

end

**View factor between 2 and 5:**

function f25=vf\_2pa5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 2 and 5

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=b2;

L=l2;

W=b5-t2-t4;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f25=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f25 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f25');

f25 = 0 - f25;

end

**View factor between 2 and 3:**

function f23=vf\_2pa3(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 2 and 3

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=b2;

L=l2;

W=b3-t2-t4;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f23=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f23 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f23');

f23 = 0 - f23;

end

**View factor between 1 and 6:**

function f16=vf\_1pll6(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%INTAKE VARIABLES ARE DECLARED GLOBAL THEY MUST BE AVALABLE EVERYWHERE

%WHEREEVER THE CONTROL FLOWS

%view factor for parallel rectangular identical plates

%nomenclature: vf\_1pll6 read as view factor for surface 1 parallel to

%surface 6

%the view factor is in 3 cases 1||6 and 3||5 and 2||4

w1=(l1-t3-t5);

w2=(b1-t2-t4);

H=(l2);

x=w1/H;

y=w2/H;

x1=sqrt(1+x.^2);

y1=sqrt(1+y.^2);

f16 = (1/(pi\*x\*y))\*((log((x1\*x1\*y1\*y1)/(x1\*x1+y1\*y1-1)))+2\*x\*(y1\*atan(x/y1)-atan(x))+2\*y\*(x1\*atan(y/x1)-atan(y)));

if f16 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f16');

f16 = 0 - f16;

end

**View factor between 1 and 5:**

function f15=vf\_1pa5(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 1 and 5

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=l1-t5-t2;

L=b1-t2-t4; %this should be actual to b2

W=l5;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f15=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f15 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f15');

f15 = 0 - f15;

end

**View factor between 1 and 4:**

function f14=vf\_1pa4(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 1 and 4

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=b1-t2-t4;

L=l1-t4-t5;

W=l4;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f14=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f14 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f14');

f14 = 0 - f14;

end

**View factor between 1 and 3:**

function f13=vf\_1pa3(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 1 and 3

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=l1-t3-t5;

L=b1-t2-t4;

W=l3;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f13=1/(pi\*w)\*(h\*atan(1/h)+w\*atan(1/w)-sqrt(h\*h+w\*w)\*atan(1/sqrt(h\*h+w\*w))+0.25\*log(a\*(b.^(w\*w)).\*(c.^(h\*h))));

if f13 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f13');

f13 = 0 - f13;

end

**View factor between 1 and 2:**

function f12=vf\_1pa2(l1, l2, l3, l4, l5, l6, b1, b2, b3, b4, b5, b6, t1, t2, t3, t4, t5, t6)

%function calculates fiew factors between perp plates 1 and 2

%nomenclature view factor\_ plate to plate pa refers to perpendicular

H=b1-t2-t4;

L=l1-t4-t5; %this should be actual to b2

W=l2;

h=H/L;

w=W/L;

a=(1+h\*h).\*(1+w.\*w)/(1+h.\*h+w.\*w);

b=(w\*w).\*(1+h\*h+w\*w)/((1+w.\*w).\*(h.\*h+w.\*w));

c=(h\*h).\*(1+h\*h+w\*w)/((1+h.\*h).\*(h.\*h+w.\*w));

f12=1/(pi\*w).\*(h.\*atan(1/h)+w.\*atan(1/w)-sqrt(h.\*h+w.\*w)\*atan(1/sqrt(h.\*h+w.\*w))+0.25.\*log(a.\*(b.^(w.\*w)).\*(c.^(h.\*h))));

if f12 < 0

fprintf('\n Negative view factor value detected. Force it to positive value');

fprintf('\n f12');

f12 = 0 - f12;

end

**And finally the label generator: (creates a design in the command window)**

function label

%creates a label of the name of the program (suited for Satellite)

clc;

fprintf('\n');

fprintf('\n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ');

fprintf('\n / /| ');

fprintf('\n \_\_\_\_\_\_\_\_\_\_ =============== | \_\_\_\_\_\_\_\_\_\_ ');

fprintf('\n |\_\_\_\_\_\_\_\_\_\_\\ # # //\_\_\_\_\_\_\_\_\_\_| ');

fprintf('\n |\_\_\_\_\_\_\_\_\_\_ \\# PI-SAT #//\_\_\_\_\_\_\_\_\_\_\_| ');

fprintf('\n |\_\_\_\_\_\_\_\_\_\_ //# #\\ \_\_\_\_\_\_\_\_\_\_| ');

fprintf('\n |\_\_\_\_\_\_\_\_\_\_// # #/\\\_\_\_\_\_\_\_\_\_\_| ');

fprintf('\n =============== ');

fprintf('\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

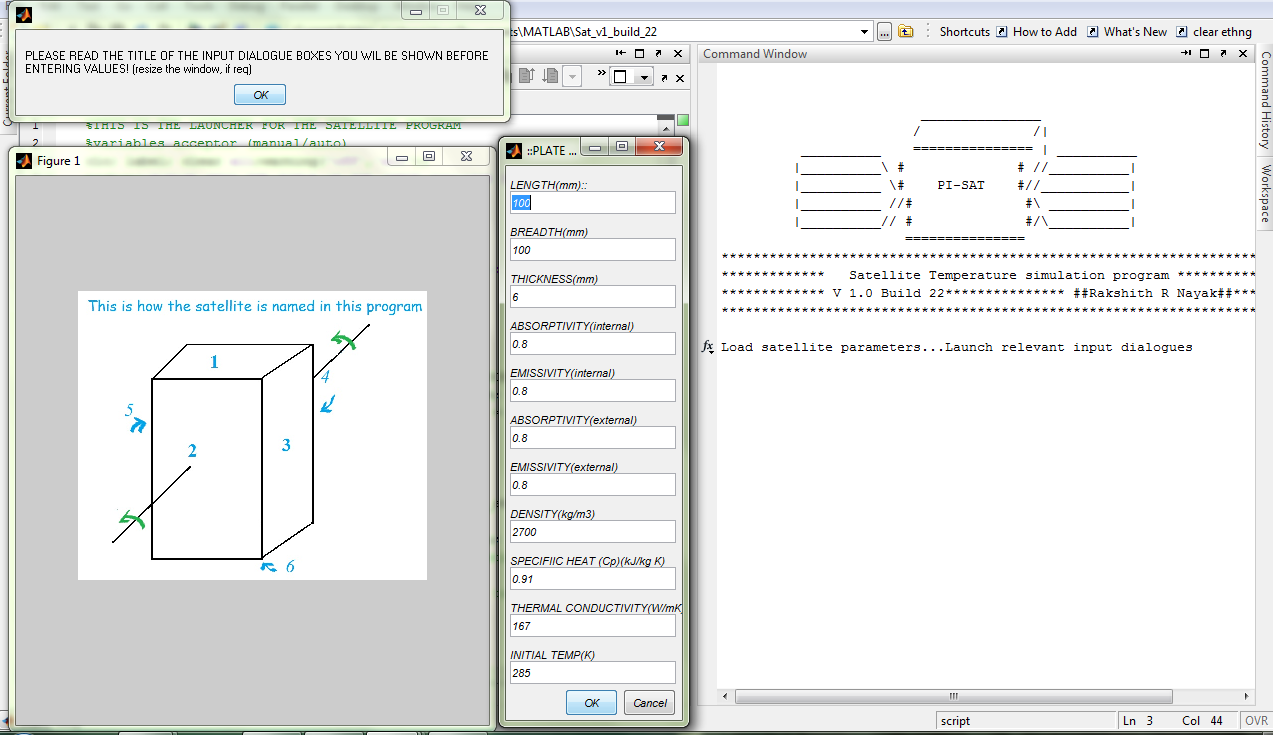
fprintf('\n\*\*\*\*\*\*\*\*\*\*\*\*\*Satellite Temperature simulation program\*\*\*\*\*\*\*\*\*\*');

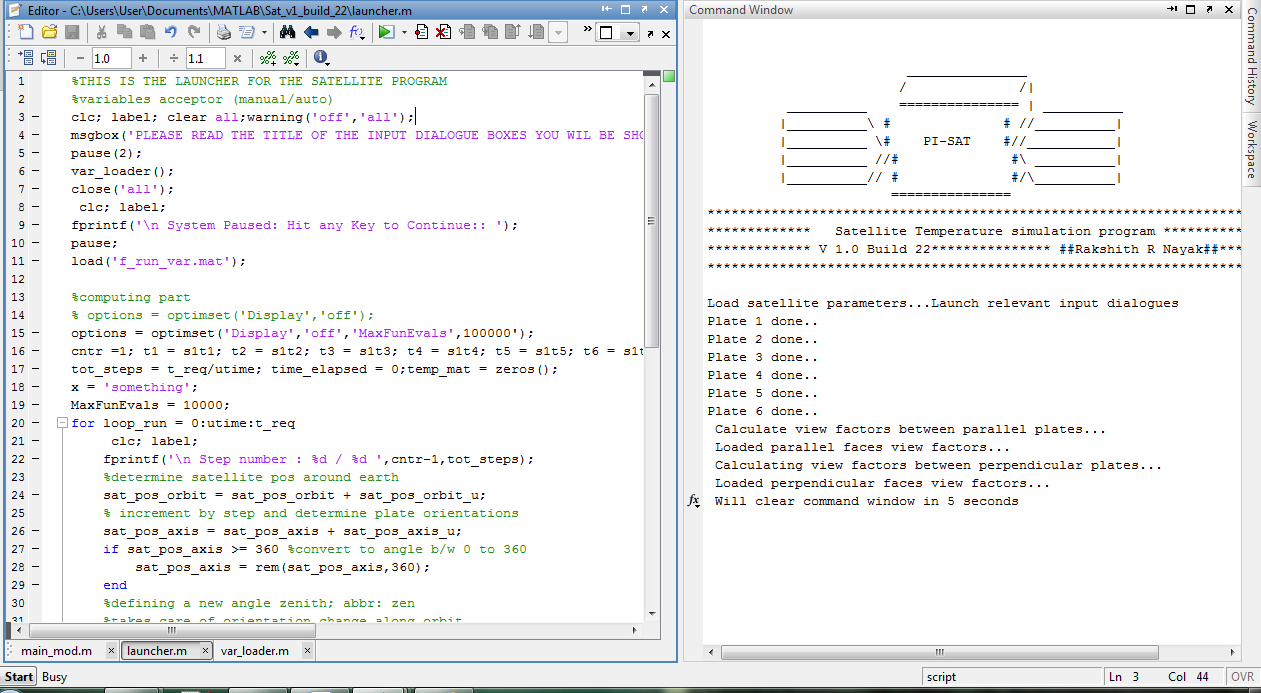
fprintf('\n\*\*\*\*\*\*\*\*\*\*\*\*\* V 1.0 Build 22\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

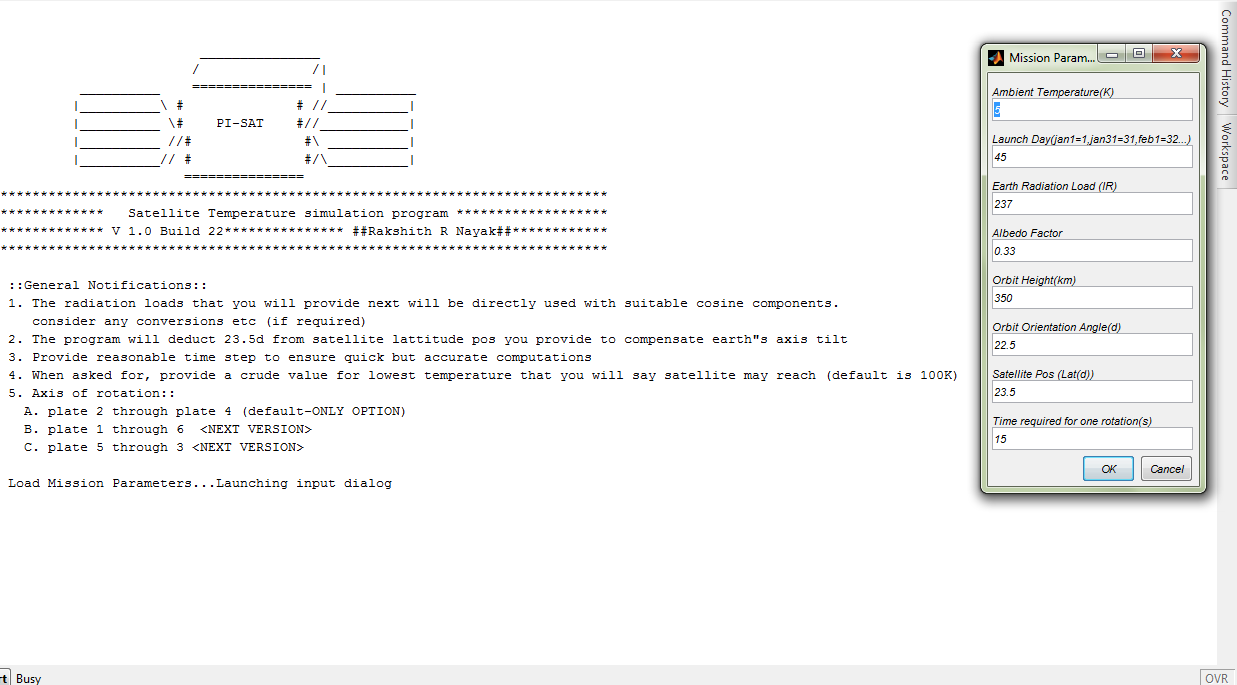
fprintf('\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

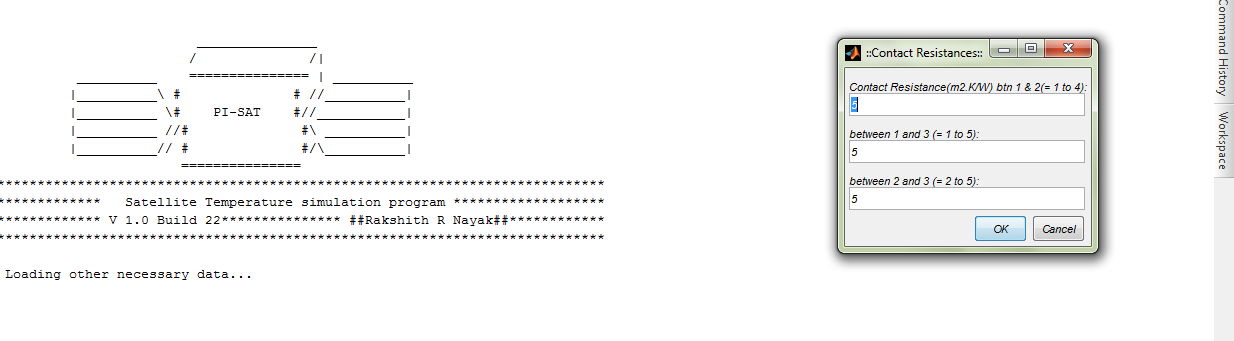
fprintf('\n');

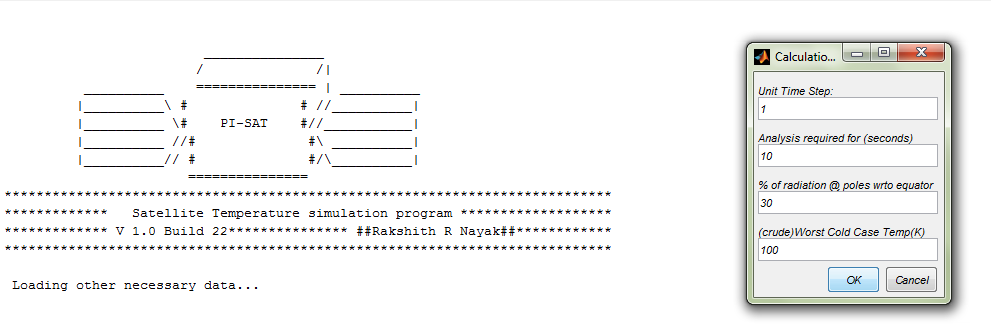
**A few screen shots of the system, when the program is running:**

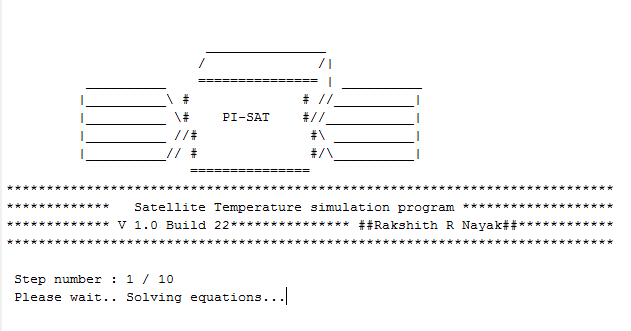
****

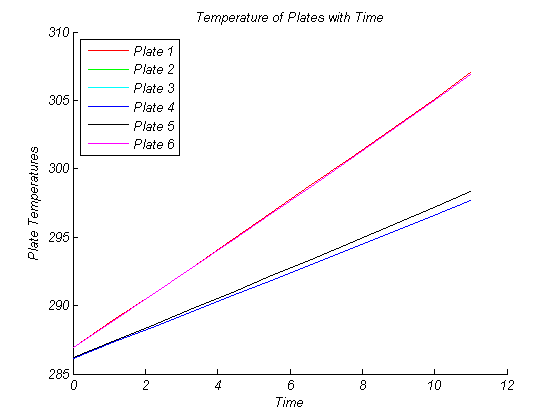
****

****

****

****

****

****

**And finally thank the user for his time and patience:**

****

**[This image is from the internet and is assumed to be on the public domain]**