```
clc; clear;

%% Allows user to set # of runs
runs=input('How many times do you want to run this code: ');
for bigloop=1:runs %Runs the code as many number of times user wants to.

%% Imports data from data file
fid=fopen('Data.txt'); %Opens The data file
File=textscan(fid, '%f %f %f','HeaderLines', 5); %puts the numbers in cell array while ignoring↙
the headers
A_raw = File{1,1};  %Sorts the 1x3 cell arrays into individual arrays
v_raw = File{1,2};  %Sorts the 1x3 cell arrays into individual arrays
hf_raw = File{1,3}; %Sorts the 1x3 cell arrays into individual arrays

%% Allows user to input which set of data to run
row_selected=input('Please enter the row number to graph: '); %user selects which row to run
if row_selected>19 %prevents user from selcting invalid rows
    error('Please enter a row number less than 20') %Displays error and stops the run
elseif row_selected<0 %prevents user from selcting invalid rows
    error('Please enter a row number greater than 0') %Displayes error and stops the run
else
end
%% Assigns the Selected values to the Variables
A=A_raw(row_selected,1);   %Launch angle
v=v_raw(row_selected,1);   %Launch Velocity
hf=hf_raw(row_selected,1); %Landing Height
g=9.81;                    %Gravitational Constant

%% Calculations for setting Limits on graphs
tf=(-2*v*sind(A)-sqrt((2*v*sind(A))^2-4*(-g)*(-2*hf)))/(2*(-g)); %time it takes reach final↙
height (Final Time)
tmax=tf/2; %time it takes to reach max height (to calculate max height)
hmax=v*tmax*sind(A)-((g)*tmax^2)/2; %Max height
dismax=(v*cosd(A))*tf; %max horizontal Distance

%% Graphs Vertical Height over Time
if A<=90 && v>0 && hf<hmax && 0<A %Sets input Condition inorder to prevent invalid data
    t1=0:.01:tf; %Sets time interval, 0 to Final Time
    h1=v*t1.*sind(A)-((g)*t1.^2)/2;% Height Equation
    xlim([0 tf]) %Sets x-axis limit from 0 to Final Time
    if hf<0 %if landing height is less than 0
        ylim([hf hmax+1]) %Sets y-axis limit from landing Height to max height+1
    else %if landing height is not less than 0
        ylim([0 hmax+1]) %Sets y-axis limit from 0 to max height+1
    end
    subplot(3,1,1) %Makes a figure with 3 rows and 1 column of graph
    plot(t1,h1,'.') %Plots Verticle Height vs Time
    title('Vertical Height over Time') %adds titile to the graph
    xlabel('Time (s)'); %lables x-axis
    ylabel('Vertical Height(m)'); %lables y-axis
else %prevents invalid input values from running
    error(['Angle has to be between 0 and 90, Velocity cant be Negative, ' ...
        'Landing Height cant be greater than Max Height']) %Displayes error and stops the run
end

%% Graphs Horizontal Distance over Time
if A<=90 && v>0 && hf<hmax && 0<A %Sets input Condition inorder to prevent invalid data
    t2=0:.01:tf; %Sets time interval, 0 to Final Time
    distance2=v*t2.*cosd(A); %horizontal Distance formula
    xlim([0 tf]); %Sets x-axis limit from 0 to Final Time
    subplot(3,1,2); %Makes a figure with 3 rows and 1 column of graph
    plot(t2,distance2,'.') %Plots Horizontal Distance vs Time
    title('Horizontal Distance over Time') %adds titile to the graph
```

```
    xlabel('Time (s)'); %lables x-axis
    ylabel('Horizontal Distance (m)') %lables y-axis
else %prevents invalid input values from running
    error(['Angle has to be between 0 and 90, Velocity cant be Negative, ' ...
        'Landing Height cant be greater than Max Height']) %Displays error and stops the run
end

%% Graphs Instantaneous Velocity over Time
if A<=90 && v>0 && hf<hmax && 0<A %Sets input Condition inorder to prevent invalid data
    t3=0:.01:tf; %Sets time interval, 0 to Final Time
    xlim([0 tf]); %Sets x-axis limit from 0 to Final Time
    h3=v*t1.*sind(A)-((g)*t1.^2)/2; %Height Equation for velocity input
    Velo3=sqrt(v^2-2*g*h3); %velocity Equation
    subplot(3,1,3); %Makes a figure with 3 rows and 1 column of graph
    plot(t3,Velo3,'.') %Plots Horizontal Distance vs Time
    hold on %prevents these graphs from being overridden
    title('Instantaneous Velocity over Time') %adds titile to the graph
    xlabel('Time (s)'); %lables x-axis
    ylabel('Instantaneous Velocity (m/s)'); %lables y-axis
else %prevents invalid input values from running
    error(['Angle has to be between 0 and 90, Velocity cant be Negative, ' ...
        'Landing Height cant be greater than Max Height']) %Displayes error and stops the run
end

%% pause allows user to move the figure 1 graph so its not coverd by figure 2
pause(2) %pauses the run for 2 seconds

%% Function graphs the animation of trajectory
Trajectory(v,A,tf,dismax,hf,hmax);

%% Allows graphs to be override for next run
hold off

end
```