```
clc
close all
clear
%Milestone 1
%Code written by: Yaman Pandey
%Submitted to: Dr. Pawan Karra

%This code takes the input for initial speed and angle of projection from
%the user. The projectile launches with that input speed and angle. There
%are three set target. If the projectile hits red target, user gets 3
%points. If the projectile hits blue of black target, user gets 5 points.


%As per requirement, air drag has been considerd.

%Terms and conditions
disp('Terms and Conditions')
disp('1. The game is on the development phase, you might not be able to enjoy
to fullest')
disp('2. You need to be a human to play this game')
disp('3. You may require some physics and mathematics knowledge to play this
game')

disp(' Type Y to continue')
disp('You will not be able to proceed if you dont type Y')

%Asking user if they want to accept terms and conditions.
Terms = input('Do you accept the terms and conditions?','s')

%Asking user their name to display it on the screen.
name = input('What is your name?','s')
Diff = input('Select level of difficulty,you may chhose from 1 to 3');


v(1) = input('What velocity you want to project?'); %Having user input
initial velocity in m/s
th(1) = input('What angle you want to project?'); % Having user input
projection angle in degrees

magv(1) = v(1);
magth(1) = th(1);
g = 9.8; %acceleration due to gravity in m/s^2
m =1; %mass of projectile in Kg
rho = 1.12; %Density of air
D = .2;  %diameter of projectile in m
A = (pi/4*D^2); %Area of projectile
CD = 0.45;  %Drag coefficient

x1(1) = 0; %m, initial position


y1(1) = 0;    %initial position, m
x3(1) = 0; %initial acceleration in x-direction, m/s^2
y3(1) = -g; %initial acceleration in y-direction, m/s^2
```

```
magx1(1) = 0;
magy1(1) = 0;
magx3(1) = 0;
magy3(1) = -g;

tfin =5;   %s
dt = 0.01; %s
t = 0:dt:tfin;

K = 9.99*10^-8; %constant, u/4pi

B1 = 5000; %Magnetic flux, Tesla
B2 = 5000; %Magnetic flux, Tesla
if Diff == 1;
B1 = 20000;
B2 = 20000;
elseif Diff == 2;
    B1 = 50000;
    B2 == 50000;

elseif Diff == 3;
    B1 = 100000;
    B2 = 100000;
end


x2(1) = v(1)*cosd(th(1));  %vx = vcos(th)
y2(1) = v(1)*sind(th(1)); %vy = vsin(th)
magx2(1) = v(1)*cosd(magth(1));  %vx = vcos(th)
magy2(1) = v(1)*sind(magth(1)); %vy = vsin(th)


q = 0:0.1:2*pi; %Used for drawing circle
mw = 6 +0.2*cos(q);
    mo =  4+0.2*sin(q);
    plot(mw,mo)

for n = 1:length(t)-1

    %If the user doesn't accept terms and conditions, the loop stops.
    if Terms ~= 'Y';
        disp('You need to accept terms and conditions first')
   end

    FD = 0.5*rho*CD*A*v(n)^2;  %drag force
    x3(n+1) = (FD*cosd(180+th(n)))/m;  %acceleration in x, m/s^2
    y3(n+1) = (FD*sind(180+th(n))/m)-g; %acceleration in y, m/s^2
    x1(n+1) = x1(n) +x2(n)*dt; %position in x, m
    x2(n+1) = x2(n) + dt*x3(n);  %velocity in x, m/s^2

    y1(n+1) = y1(n) + y2(n)*dt; %position in y, m
    y2(n+1) = y2(n) + dt*y3(n); %velocity in y, m/s^2
    v(n+1) = sqrt(((x2(n+1))^2)+((y2(n+1))^2)); %resultant velocity, m/s
    th(n+1) = atand(((y2(n+1)))/((x2(n+1)))); %angle of projection, degrees
```

```
    distance = sqrt(((x1(n)-6)^2)+(y1(n)-4)^2); %Calculating the distance
between magnet and projectile

    magnet = K*(B1*B2)/(distance)^2; %magnetic force
    magx3(n+1) = ((FD*cosd(180+th(n)))+magnet*sign(th(n)))/m;  %acceleration
in x, m/s^2
    magy3(n+1) = ((FD*sind(180+th(n))/m+magnet*sign(th(n))))-g; %acceleration
in y, m/s^2
    magx1(n+1) = magx1(n) +magx2(n)*dt; %position in x, m
    magx2(n+1) = magx2(n) + dt*magx3(n);  %velocity in x, m/s^2

    magy1(n+1) = magy1(n) + magy2(n)*dt; %position in y, m
    magy2(n+1) = magy2(n) + dt*magy3(n); %velocity in y, m/s^2
    magv(n+1) = sqrt(((magx2(n+1))^2)+((magy2(n+1))^2)); %resultant velocity,
m/s
    magth(n+1) = atand(((magy2(n+1)))/((magx2(n+1)))); %angle of projection,
degrees

    w = magx1(n) +0.1*cos(q);   %making a cicle and moving it on projectile
    o =  magy1(n)+0.1*sin(q); %Considering magnet

    hold on



    rectangle('Position',[6 1.5 0.3 0.4],'edgecolor','r') ;  %Drawing
rectangle at x=6 and y=1.5
    %with length of 0.3 and width of 0.4


    rectangle('Position',[5 2 0.3 0.4],'edgecolor','b')  %Drawing rectangle
at x=5 and y=2
    %with length of 0.3 and width of 0.4


    rectangle('Position',[5 1 0.3 0.4],'edgecolor','black') %Drawing
rectangle at x=5 and y=1
    %with length of 0.3 and width of 0.4
    axis([0 10 0 10])



    if y1(n) > 1 && y1(n) < 1.4 && x1(n)>5 && x1(n) < 5.3
        text(3,6,'Game Over','fontsize',40,'color','r')
        text(1,4,'Your score is 3','fontsize',20,'color','r')
        break

    elseif y1(n) > 2 && y1(n) < 2.4 && x1(n)>5 && x1(n) < 5.3
        text(3,6,'Game Over','fontsize',40,'color','r')
        text(1,4,'Your score is 3','fontsize',20,'color','r')
        break
```

```
    elseif y1(n) > 1.5 && y1(n) < 1.9 && x1(n)>6 && x1(n) < 6.3
        text(3,6,'Game Over','fontsize',40,'color','r')
        text(1,4,'Your score is 5','fontsize',20,'color','r')
        break

    elseif y1(n+1)<0
        %this command end the loop when projectile lands on ground.
        text(3,6,'Game Over','fontsize',40,'color','r')
        text(1,4,'Loser!! Your score is 0','fontsize',20,'color','r')
        break


    end
    if x1(n) > 10
       text(3,6,'Game Over','fontsize',40,'color','r')
        text(1,4,'Loser!! Your score is 0','fontsize',20,'color','r')
        break
    end
    if y1(n) > 10
       text(3,6,'Game Over','fontsize',40,'color','r')
        text(1,4,'Loser!! Your score is 0','fontsize',20,'color','r')
        break
    end
    plot(w ,o,'color','r') %Plotting projectile motion on red color
    pause(0.001)
    plot(w,o,'color','w')  %Plotting projectile motion on while color which
gives
    %illusion of animation
    plot(magx1,magy1,'color','k','linestyle','--')
    plot(x1,y1,'color','yellow','linestyle','--')
    text(1,9,'H i t  r e d = 5 p o i n t s','fontsize',14,'color','red')
    text(1,8.5,'H i t  b l u e = 3 p o i n t s','fontsize',14,'color','b')
    text(1,8,'H i t  b l a c k = 3 p o i n t
s','fontsize',14,'color','black')
    text(1,7,'P l a y e r  n a m e:','fontsize',14,'color','g')
   % text(4,7,name,'fontsize',20,'color','g')
    text(7,0.4,'Â© Y a m a n','fontsize',14,'color','black')

end


vv(1) = v(n);
tth(1) = (180+th(n));
xx1(1) = x1(n);
yy1(1) = y1(n);
xx3(1) = 0;
xx2(1) = vv(1)*cosd(tth(1));
yy2(1) = vv(1)*sind(tth(1));
yy3(1) = -g;



if y1(n) > 1 && y1(n) < 1.4 && x1(n)>5 && x1(n) < 5.3
```

```
    for j = 1:length(t)-1
        FDD = 0.5*rho*CD*A*v(j)^2;


        xx3(j+1) = (FDD*cosd(180+th(j)))/m;  %acceleration in x, m/s^2
        yy3(j+1) = (FDD*sind(180+th(j))/m)-g; %acceleration in y, m/s^2
        xx1(j+1) = xx1(j) +xx2(j)*dt; %position in x, m
        xx2(j+1) = xx2(j) + dt*xx3(j);  %velocity in x, m/s^2


        yy1(j+1) = yy1(j) + yy2(j)*dt; %position in y, m
        yy2(j+1) = yy2(j) + dt*yy3(j); %velocity in y, m/s^2
        vv(j+1) = sqrt(((xx2(j+1))^2)+((yy2(j+1))^2)); %resultant velocity,
m/s
        tth(j+1) = atand(((yy2(j+1)))/((xx2(j+1)))); %angle of projection,
degrees




        ww = xx1(j) +0.1*cos(q);
        oo = yy1(j) + 0.1*sin(q);

        plot(ww ,oo,'color','r') %Plotting projectile motion on red color
        pause(0.001)
        plot(ww,oo,'color','w')  %Plotting projectile motion on while color
which gives

        if yy1(j+1) < 0
            break
        end
    end
end

vvv(1) = v(n);
tthh(1) = (180+th(n));
xxx1(1) = x1(n);
yyy1(1) = y1(n);
xxx3(1) = 0;
xxx2(1) = vvv(1)*cosd(tthh(1));
yyy2(1) = vvv(1)*sind(tthh(1));
yyy3(1) = -g;


if y1(n) > 2 && y1(n) < 2.4 && x1(n)>5 && x1(n) < 5.3
    for k = 1:length(t)-1

        FFDD = 0.5*rho*CD*A*v(k)^2;


        xxx3(k+1) = (FFDD*cosd(180+th(k)))/m;  %acceleration in x, m/s^2
        yyy3(k+1) = (FFDD*sind(180+th(k))/m)-g; %acceleration in y, m/s^2
        xxx1(k+1) = xxx1(k) +xxx2(k)*dt; %position in x, m
        xxx2(k+1) = xxx2(k) + dt*xxx3(k);  %velocity in x, m/s^2
```

```
        yyy1(k+1) = yyy1(k) + yyy2(k)*dt; %position in y, m
        yyy2(k+1) = yyy2(k) + dt*yyy3(k); %velocity in y, m/s^2
        vvv(k+1) = sqrt(((xxx2(k+1))^2)+((yyy2(k+1))^2)); %resultant
velocity, m/s
        tthh(k+1) = atand(((yyy2(k+1)))/((xxx2(k+1)))); %angle of projection,
degrees



        www = xxx1(k) +0.1*cos(q);
        ooo = yyy1(k) + 0.1*sin(q);

        plot(www ,ooo,'color','r') %Plotting projectile motion on red color
        pause(0.001)
        plot(www,ooo,'color','w')  %Plotting projectile motion on while color
which gives
        %illusion of animation
        %disp(vv(j))
        %disp(tth(j))
        %disp(xx1(j))
        if yyy1(k+1) < 0
            break
        end
    end
end

vvvv(1) = v(n);
tthhh(1) = (180+th(n));
xxxx1(1) = x1(n);
yyyy1(1) = y1(n);
xxxx3(1) = 0;
xxxx2(1) = vvvv(1)*cosd(tthhh(1));
yyyy2(1) = vvv(1)*sind(tthhh(1));
yyyy3(1) = -g;


if y1(n) > 1.5 && y1(n) < 1.9 && x1(n)>6 && x1(n) < 6.3

    for nn = 1:length(t)-1

        FFDDD = 0.5*rho*CD*A*v(nn)^2;


        xxxx3(nn+1) = (FFDDD*cosd(180+th(nn)))/m;  %acceleration in x, m/s^2
        yyyy3(nn+1) = (FFDDD*sind(180+th(nn))/m)-g; %acceleration in y, m/s^2
        xxxx1(nn+1) = xxxx1(nn) +xxxx2(nn)*dt; %position in x, m
        xxxx2(nn+1) = xxxx2(nn) + dt*xxxx3(nn);  %velocity in x, m/s^2


        yyyy1(nn+1) = yyyy1(nn) + yyyy2(nn)*dt; %position in y, m
        yyyy2(nn+1) = yyyy2(nn) + dt*yyyy3(nn); %velocity in y, m/s^2
        vvvv(nn+1) = sqrt(((xxxx2(nn+1))^2)+((yyyy2(nn+1))^2)); %resultant
velocity, m/s
```

```
        tthhh(nn+1) = atand(((yyyy2(nn+1)))/((xxxx2(nn+1)))); %angle of
projection, degrees



        wwww = xxxx1(nn) +0.1*cos(q);
        oooo = yyyy1(nn) + 0.1*sin(q);

        plot(wwww ,oooo,'color','r') %Plotting projectile motion on red color
        pause(0.001)
        plot(wwww,oooo,'color','w')  %Plotting projectile motion on while
color which gives
        %illusion of animation
        %disp(vv(j))
        %disp(tth(j))
        %disp(xx1(j))
        if yyyy1(nn+1) < 0
            break
        end
    end
end
```