```matlab
%% Q1
clc;

%Q1.1.

%declaring the veriables and constants
E_0 = 2050; % [V/m]
L = 0.12; % [m]
a = (pi/4)^0.5; % [-]
b = 0.08; % [-]
x1 = L/4; % [m]
x2 = (3*L)/4; % [m]
h = 1*10^(-4); % [m]
n = 2;
N = ceil((x2-x1)/(n*h))

%defining the function E(x)
syms x;
E = E_0*cos(((a*x)/(L))^2)*exp(-b*(x/L)^(3/2));
E = matlabFunction(E);

% We will choose simpson's method of integration.
% If(x) = h/3 + (

% We will now implement the method
to_continue = true;
I = zeros(1,N);
i = 1; % itere tion number
lowerlimit = x1;
upperlimit = lowerlimit + 2*h;
while (to_continue)
    I(i) = (h/3)*(E(lowerlimit)+4*E(lowerlimit+h)+E(upperlimit));
    i = i+1;
    lowerlimit = upperlimit;
    upperlimit = lowerlimit + 2*h;
    if i > N
        to_continue = false;
        i
    end

end
Delta_phi = - sum(I)

%% Defining a function that does the simpson's method for E(x) automatically

% We created a function that does the simpson's method of integration (you
% need to input the lower limit and upper limit of the integral as well as
% the value of h)
%
% The syntax is:  simpsons_for_E(lower limit, upper limit, h)
```

```
%
% We also assigned her an handle.
clc;

%defining constants
L = 0.12; % [m]
x1 = L/4; % [m]
x2 = (3*L)/4; % [m]
h = 1*10^(-4); % [m]

simpsons_for_E = @Integration_by_Simpsons_method_for_E_of_x;
Delta_phi = - simpsons_for_E(x1,x2,h)

%% Q1.2.
clc;
format long

H = [2.5*10^(-3), 1*10^(-3), 5*10^(-4), 2.5*10^(-4), 1*10^(-4), 5*10^(-5), 1*10^(-5)]; % [m]
% H is a vector of h

%defining constants
L = 0.12; % [m]
x1 = L/4; % [m]
x2 = (3*L)/4; % [m]
simpsons_for_E = @Integration_by_Simpsons_method_for_E_of_x;

Delta_phis = zeros(1,length(H));

%claculating the Delta_phi for each h in H
for i = 1:length(H)
    Delta_phis(i) = - simpsons_for_E(x1,x2,H(i))
end

%% ploting delta_phi as a function of h
fig1 = figure ("Name",'Delta-phi as a function of h','Position',[20 50 1500 800]);
semilogx(H,Delta_phis,'-*','LineWidth',2)
title (["Plot of Delta-phi as a function of h", "Almog Dobrescu 214254252 & Ronnel Nawy 325021152"])
xlabel('h [m]')
ylabel('delta-phi [V]')
grid on
grid minor
legend({'delta-phi'},'FontSize',14 ,'Location','southeast')
%exportgraphics(fig1, 'Q1_2-graph.png','Resolution',1200); %export the fig to a png file

%% Q1.3.
clc;
format long
```

```
%defining constants and variabels
E_0 = 2050; % [V/m]
L = 0.12; % [m]
a = (pi/4)^0.5; % [-]
b = 0.08; % [-]
h = 10^(-4);
x1 = 0; % [m]
x2s = 0:h:L;
phi_0 = 300; % [V]
phis = zeros(1,length(x2s));
simpsons_for_E = @Integration_by_Simpsons_method_for_E_of_x;

%phi(i) = phi_0 - simpsons_for_E(x1,x2s(i),h)

for i = 1:length(phis)
    phis(i) = phi_0 - simpsons_for_E(x1,x2s(i),h);
    %i
end

Es = E_0.*cos(((a.*x2s)/(L)).^2).*exp(-b.*(x2s./L).^(3/2));

%% ploting phi and E as a function of x
fig2 = figure ("Name",'phi and E as a function of x','Position',[20 50 1500 800]);
title (["Plot of Phi and E as a Function of x", "Almog Dobrescu 214254252 & Ronnel Nawy 325021152"])
yyaxis left
plot(x2s,phis, 'LineWidth', 2, 'color',[0 0.4470 0.7410])
ylabel('Phi(x) [V]')
xlabel('x [m]')
yyaxis right
plot(x2s,Es, 'LineWidth', 2, 'color',[0.8500 0.3250 0.0980])
ylabel('E(x) [V/m]')
grid on
grid minor
legend({'Phi(x)','E(x)'},'FontSize',14 ,'Location','southwest')
%exportgraphics(fig2, 'Q1_3-graph.png','Resolution',1200); %export the fig to a png file

Delta_phi
phis(end)
```