```
clc
clf
clear all


%***********************************************************************
%
% INPUTS
%
% Click the run button and refer to the command window
% These are the inputs that can be modified by the user
%
% f(x), the function to integrate

  f= @(x) 2000*log(1400/21./x)-9.8*x ;

% a, the lower limit of integration

  a= 8 ;

% b, the upper limit of integration

  b= 12 ;

% n, the maximum number of segments

  n=128 ;

%***********************************************************************

disp(sprintf('\n\nConvergence of Simpson''s 1/3rd Rule'))
disp(sprintf('University of South Florida'))
disp(sprintf('United States of America'))
disp(sprintf('kaw@eng.usf.edu\n'))

disp(sprintf↙
('\n*******************************Introduction*********************************'))

disp('The following simulation illustrates the convergence of Simpson''s 1/3rd Rule')
disp('rule applied to numerically integrate functions. This section is the')
disp('only section where the user interacts with the program.  The user ')
disp('enters a function in the form f(x), the lower and upper limit of integration,')
disp('a and b, and the number of subdivisions to take. By entering this data, the')
disp('program will calculate the exact (Matlab numerical value if it is not exact)')
disp('value of the solution, followed by the results using Simpson''s 1/3rd Rule')
disp('with 2, 4, 8, 16 ... n segments. The program will also display the true error,')
disp('the absolute relative percentage true error, the approximate error, the absolute')
disp('relative aprroximate percentage error, and the least number of significant ')
disp('digits correct all as a function of number of segments.')

disp(sprintf('\n\n*******************************Input↙
Data**********************************\n'))
```

```
disp(sprintf('     f(x), integrand function'))
disp(sprintf('     a = %g, lower limit of integration ',a))
disp(sprintf('     b = %g, upper limit of integration ',b))
disp(sprintf('     n = %g, number of subdivision',n))
format short g

% Determine the exact solution
exact = quad(f,a,b) ;

% Determine number of different segments to consider
nstep = floor(log2(n)) ;

for i=0:nstep-1

  % Determine number of segments and h
  NN(i+1)=2^(i+1) ;
  h=(b-a)/NN(i+1) ;

  integral = f(a) + f(b) ;
  for j=1:2:NN(i+1)-1
    integral = integral + 4*f(a+h*j) ;
  end
  for j=2:2:NN(i+1)-2
    integral = integral + 2*f(a+h*j) ;
  end
  integral = integral * h/3 ;
  YY(i+1)=integral ;

  % Compute Errors
  Et(i+1)=exact-integral ;
  Etabs(i+1)=abs((integral-exact)/exact) ;
  if(i > 0)
    Ea(i+1)=YY(i+1)-YY(i) ;
    Eaabs(i+1)=abs((YY(i+1)-YY(i))/YY(i)) ;
    SD(i+1)=floor((2-log10(Eaabs(i+1)/0.5))) ;
    if(SD(i+1)<0)
      SD(i+1)=0
    end
  else
    Ea(1)=0 ;
    Eaabs(1)=0 ;
    SD(1)=0 ;
  end

end


disp(sprintf('\n\n***********************Table of
Values******************************\n'))

disp('        Approx        True     Relative      Approx   Rel Appr   Sig   ')
disp(' n     Integral       Error   True Error     Error     Error   Digits ')
```

```
disp('-----------------------------------------------------------------------')



for i=1:nstep

if(i > 1)
  if(exact || YY(i) > 0)
    disp(sprintf('%4i  %+1.3e  %+1.3e  %+1.3e  %+1.3e  %+1.3e  %2i',NN(i),YY(i),Et(i),↙
Etabs(i),Ea(i),Eaabs(i),SD(i) ))
  else
    disp(sprintf('%4i  %+1.3e  %+1.3e     n/a      %+1.3e      n/a     n/a',NN(i),YY(i),↙
Etabs(i),Ea(i)))
  end
else
  disp(sprintf('%4i  %+1.3e  %+1.3e  %+1.3e      n/a         n/a     n/a',NN(i),YY(i),Et↙
(i),Etabs(i)))
end

end
disp('-----------------------------------------------------------------------')

% The following generates 3 plots. This function detects information about your
% screensize and tries to then place/size the graphs accordingly.
scnsize = get(0,'ScreenSize');

% Graph 1: The following code sets up the graphical depiction of the method.
x(1)=a ;
y(1)=f(a) ;
hold on
for i=1:n
   x(i+1)=a+i*h ;
   y(i+1)=f(x(i+1)) ;
end

for i=1:2:n
    bg = i ;
    ed = i + 2 ;
    coeffs = polyfit(x(bg:ed), y(bg:ed), 2);
    poly_x1 = [x(bg):(x(ed) - x(bg))/1000:x(ed)];
    poly_y1 = coeffs(1)*poly_x1.^2 + coeffs(2)*poly_x1 + coeffs(3);
    poly_x1 = [poly_x1(1) poly_x1 poly_x1(end)];
    poly_y1 = [0 poly_y1 0];
    fill(poly_x1, poly_y1, 'y')
end
xrange=a:(b-a)/1000:b;
plot(xrange,f(xrange),'k','Linewidth',2)
title('Integrand function and Graphical Depiction of Simpson''s 1/3rd Rule')

% Graph 2: Approximation and True Errors
fig2=figure ;
set(fig2,'Position',[0.2*scnsize(3),0.2*scnsize(3),0.6*scnsize(3),0.2*scnsize(4)]) ;
```

```
subplot(1,3,1); plot(NN,YY,'-O','LineWidth',2,'Color',[1 0 0]);
title('Appr. Integral vs No. of Segments')

subplot(1,3,2); plot(NN,Et,'-O','LineWidth',2,'Color',[0 0 1]);
title('Et vs No. of Segments')

subplot(1,3,3); plot(NN,Etabs,'-O','LineWidth',2,'Color',[0 0 1]);
title('Abs et vs No. of Segments')

% Graph 3: Relative Errors and Significant Digits
fig = figure ;
set(fig,'Position',[0.2*scnsize(3),0,0.6*scnsize(3),0.2*scnsize(4)]) ;
subplot(1,3,1); plot(NN(2:nstep),Ea(2:nstep),'-O','LineWidth',2,'Color',[0 1 0]);
title('Ea vs No. of Segments')

subplot(1,3,2); plot(NN(2:nstep),Eaabs(2:nstep),'-O','LineWidth',2,'Color',[0 1 0]);
title('Abs ea vs No. of Segments')

subplot(1,3,3); plot(NN(2:nstep),SD(2:nstep),'-O','LineWidth',2,'Color',[1 0.5 0.5]);
title('Significant Digits Correct vs No. of Segments')
```