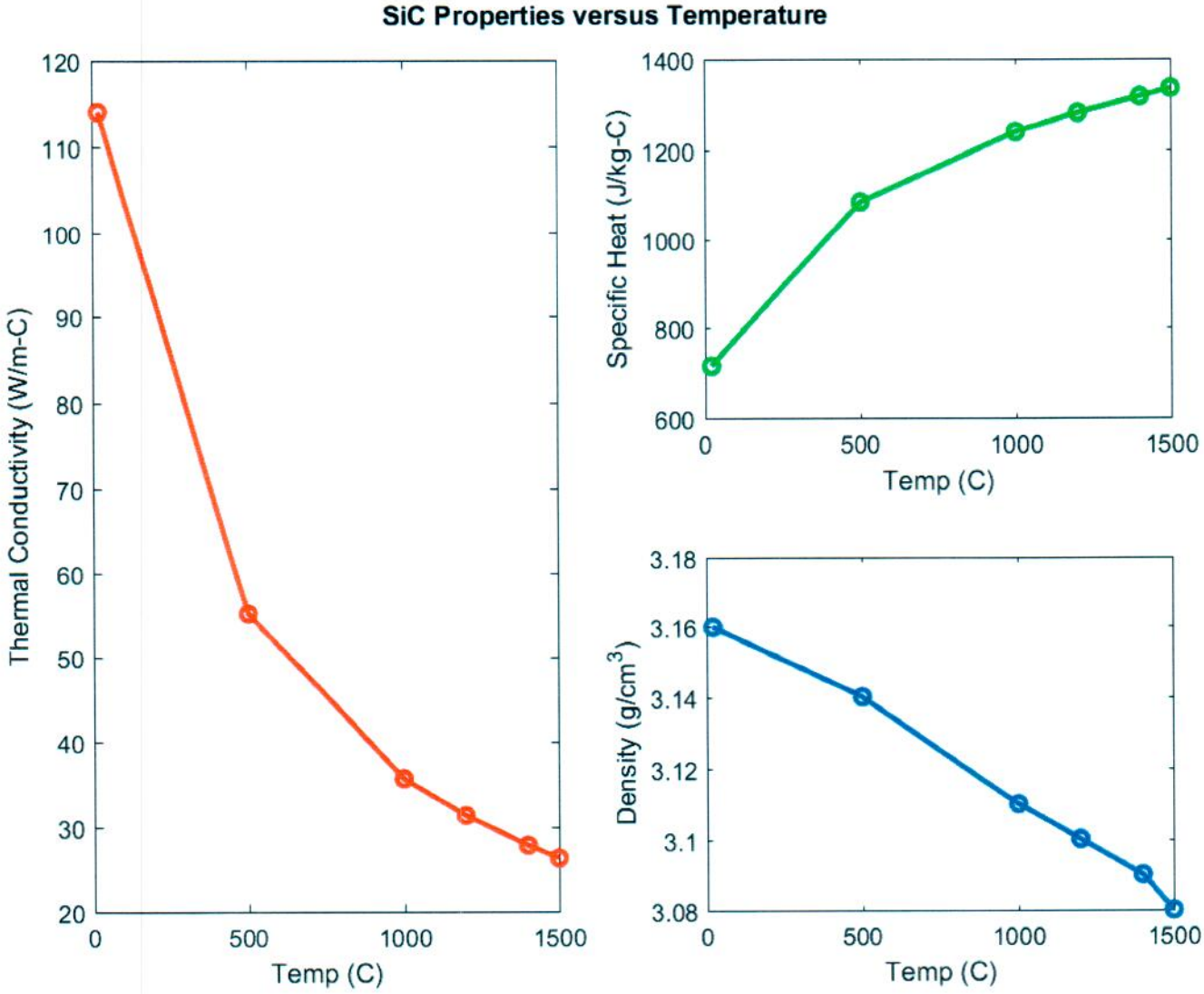
SiC Properties versus Temperature
Thermal Conductivity (W/m-C)
Temp (C)
Specific Heat (J/kg-C)
Temp (C)
Density (g/cm3)
Temp (C)

```
%
%  SiC_PlotData.M   Process some temperature dependent data for SiC
%
%  Two tasks will be performed here given the data file SiC.dat:
%    1. read the data file SiC.dat using the loadColData.m file (file obtained
%       from the NMM toolbox at www.me.pdx.edu/~gerry/nmm/ )
%    2. plot the temperature dependence of the density, specific heat, and
%       thermal conductivity in a specific fashion (3 subplots with one on the left
%       and two on the right)
%
%  File prepared by J. R. White, UMass-Lowell (last update: Sept. 2017)
%

  clear all,   close all,   nfig = 0;
%
%  Task 1:  use loadColData to read the desired data file
%  resulting data:  T = vector containing the temperature (C)
%                   props = matrix containing the bulk modulus (GPa), density (g/cm^3),
%                           specific heat (J/kg-C), thermal conductivity (W/m-C), and
%                           thermal expansion coeff (1/C) within 5 columns
  [T,props] = loadColData('SiC.dat',6,5,1);
%
%  Task 2:  plot the temperature dependence of the properties in a creative way
  nfig = nfig+1;   figure(nfig)
  subplot(1,2,1),plot(T,props(:,4),'r-o','LineWidth',2), grid
  title('SiC Properties versus Temperature')
  xlabel('Temp (C)'),ylabel('Thermal Conductivity (W/m-C)')
  subplot(2,2,2),plot(T,props(:,3),'g-o','LineWidth',2), grid
  xlabel('Temp (C)'),ylabel('Specific Heat (J/kg-C)')
  subplot(2,2,4),plot(T,props(:,2),'b-o','LineWidth',2), grid
  xlabel('Temp (C)'),ylabel('Density (g/cm^3)')
%
%  end of program
```

```
>> geometric_series2

          Output from the Geometric_Series program

    The exact value of the series for a = 1/2 and r = 1/2 is  1.000

 Part 1:
  Input number of terms in the geometric series: 5
   For    5 terms, SN =  9.68750e-01 and this represents an error of  3.125e+00%

 Part 2:
  Input the iteration tolerance (in percent):  1
   For    7 terms, SN =  9.92188e-01 and this represents an error of  7.813e-01%


>> geometric_series2

          Output from the Geometric_Series program

    The exact value of the series for a = 1/2 and r = 1/2 is  1.000

 Part 1:
  Input number of terms in the geometric series: 10
   For   10 terms, SN =  9.99023e-01 and this represents an error of  9.766e-02%

 Part 2:
  Input the iteration tolerance (in percent):  0.01
   For   14 terms, SN =  9.99939e-01 and this represents an error of  6.104e-03%
>>
```

```
%
%   GEOMETRIC_SERIES2.M  Program to illustrate various looping structures in Matlab
%
%   The program uses the geometric series to illustrate for .. end and while ... end
%   structures in Matlab.  It uses two different approaches to evaluate the truncated
%   infinite series:
%     1. evaluate sum for fixed number of terms  (uses for ... end loop)
%     2. evaluate sum until some error criterion is met (uses while ... end loop)
%
%   File prepared by J. R. White, UMass-Lowell  (last update:  Oct. 2017)
%
%
     clear all,  close all
%
     fprintf(1,'\n          Output from the Geometric_Series program \n\n');
     a = 1/2;   r = 1/2;   S = a/(1-r);
     fprintf(1,'    The exact value of the series for a = 1/2 and r = 1/2 is %6.3f ↙
\n\n',S);
%
%   Part 1   Evaluate sum for fixed number of terms          ← printer error
     fprintf(1,' Part 1: \n');
     N1 = input('  Input number of terms in the geometric series: ');
     S1 = 0;
     for k = 1:N1
       S1 = S1 + a*r^(k-1);
     end
     E1 = abs(S1 - S)/S*100;   % percent error
     fprintf(1,'   For %4i terms, SN = %12.5e and this represents an error of %10.3e%% ↙
\n\n', ...
              N1,S1,E1);
%
%   Part 2   Evaluate sum based on conditional test
     fprintf(1,' Part 2: \n');
     tolp = input('  Input the iteration tolerance (in percent):  ');
     tol = tolp/100;     % relative tolerance (best not to work with percent values)
     S2 = 0;  k = 0;  rerr = tol+1;
     while rerr > tol
       k = k+1;
       Tk = a*r^(k-1);
       S2 = S2 + Tk;
       rerr = abs(Tk)/S2;
     end
     N2 = k;
     E2 = abs(S2 - S)/S*100;   % percent error
     fprintf(1,'   For %4i terms, SN = %12.5e and this represents an error of %10.3e%% ↙
\n', ...
              N2,S2,E2);
%
%   end of function
```