```
>> vdwaal_1
Input the desired temperature (C): 20
Input the desired molar volume (L/mole): 10

Pressure of helium at T = 20 C and v = 10 L/mole is 2.40973 atm.


>> vdwaal_1
Input the desired temperature (C): 20
Input the desired molar volume (L/mole): 10

Pressure of hydrogen at T = 20 C and v = 10 L/mole is 2.40833 atm.


>> vdwaal_1
Input the desired temperature (C): 20
Input the desired molar volume (L/mole): 10

Pressure of oxygen at T = 20 C and v = 10 L/mole is 2.39843 atm.


>> vdwaal_1
Input the desired temperature (C): 20
Input the desired molar volume (L/mole): 10

Pressure of chlorine at T = 20 C and v = 10 L/mole is 2.35305 atm.


>> vdwaal_1
Input the desired temperature (C): 20
Input the desired molar volume (L/mole): 10

Pressure of carbon dioxide at T = 20 C and v = 10 L/mole is 2.37877 atm.

>>
```

```
%
%   VDWAAL_1.M   Main program to test the VDWAAL_1A function
%
%   This program simply calls the VDWAAL_1A function with a given set of data that is
%   selected/input by the user.  The gas pressure is output for the conditions given.
%
%   The goal here is to get some experience with a few relatively simple programming
%   tasks -- that is, to become familiar with the use of functions and the input
%   and menu commands in Matlab.
%
%   File prepared by J. R. White, UMass-Lowell (last update: Oct. 2017)
%

     clear all,  close all,   nfig = 0;
%
%   select gas of interest
     gases = {'helium';'hydrogen';'oxygen';'chlorine';'carbon dioxide'};
     ng = menu('Select the gas of interest:','helium', ...
               '          hydrogen          ', ...
               'oxygen','chlorine','carbon dioxide');
     gas = char(gases(ng));  % gas of interest (this is now a character string)
%
%   input the desired conditions
     TC = input('Input the desired temperature (C): ');   TK = TC+273;
     v = input('Input the desired molar volume (L/mole): ');
%
%   call function to determine the pressure of the gas under the input conditions
     P = vdwaal_1a(TK,v,ng);
%
%   now edit the numerical results
     fprintf(1,'\n');
     fprintf(1,'Pressure of %s at T = %g C and v = %g L/mole is %g atm. \n', ...
            gas,TC,v,P);
%
%   end of program
```

```
%
%   VDWAAL_1A.M   Function file to evaluate van der Waal's eqn
%                        for several gases (given T and v)
%
%   Inputs:
%     T  --  temperature of gas (K)
%     v  --  molar volume of gas (L/mole)
%     ng --  gas of interest (position number in table of available gases)
%
%   Outputs:
%     P  -- pressure (atm) of gas selected at conditions given
%
%   NOTE:  All inputs and outputs are scalars  (will not work for vector inputs)
%
%   File prepared by J. R. White, UMass-Lowell (last update: Oct. 2017)
%

      function [P] = vdwaal_1a(T,v,ng)
%
%   test input data
      N = length(T) + length(v);
      if N > 2
        disp(' Warning  --  This function was not designed for vector inputs!!!');
      end
%
%   data for various gases (order is important)
%   order -> helium, hydrogen, oxygen, chlorine, carbon dioxide
      a = [0.0341 0.244 1.36 6.49 3.59];          % a coeff (L^2-atm/mole^2)
      b = [0.0237 0.0266 0.0318 0.0562 0.0427];  % b coeff (L/mole)
      R = 0.08206;            % universal gas constant (L-atm/mole-K)
%
%   now compute pressure for the given input data
      P = R*T/(v-b(ng)) - a(ng)/v^2;
%
%   end of function
```

```
%
%   LOOPS_2.M   Demo to evaluate finite and infinite series in Matlab
%
%   Written by J. R. White, UMass-Lowell (last update: Oct. 2017)
%

%
%   Part a  --  evaluate series with finite number of terms
      N = 25;    S = 0;
      for k = 1:N
        Sk = (-1)^(k+1)/(2*k-1);
        S = S + Sk;
      end
      fprintf(' Part a \n');
      fprintf(' Value of the series with N = %3i is: %8.5f \n\n', N,S);

%
%   Part b  --  evaluate infinite series until relative error < 0.001
      S = 0;  k = 0;
      rerr = 1.0;    tol = 1e-3;
      while rerr > tol
        k = k+1;
        Sk = (-1)^(k+1)/(2*k-1);
        S = S + Sk;
        rerr = abs(Sk/S);
      end
      fprintf(' Part b \n');
      fprintf(' Final relative error is = %10.5e \n',rerr);
      fprintf(' Value of the series with %3i terms is: %8.5f \n', k,S);
%
%   end of program
```

```
>> loops_2
 Part a
 Value of the series with N =  25 is:  0.79539

 Part b
 Final relative error is = 9.99689e-04
 Value of the series with 637 terms is:  0.78579
```

```
%
%   DINT_MAIN.M   Main program to test the DINT function
%
%   This program simply calls the DINT function using y(x) = exp(-x) to generate
%   the integral over the range a = 0 to b = 2.  This represents a simple test
%   of the function.  The exact result is I = 0.8647, so the integral estimate
%   should approach this result as the number of intervals, N, is increased.
%   Let's see...
%
%   File prepared by J. R. White, UMass-Lowell (last update: Oct. 2017)
%

     clear all,  close all,   nfig = 0;
%
%   set the range, number of intervals, the function parameters, and the desired I
     a = 0;  b = 2;
     N = input('Enter the number of intervals to use:  ');
     x = linspace(a,b,N+1);    y = exp(-x);
     I = dint(x,y);
     disp('Desired Integral = '), disp(I)
%
%   end of program
```

```
%
%   DINT.M   Function file to integrate discrete data
%
%   Inputs:
%     x  --  discrete independent variable (vector)
%     y  --  discrete dependent variable (vector)
%
%   Outputs:
%     I  --  scalar value of estimate of integral of y(x) over x range
%
%   File prepared by J. R. White, UMass-Lowell (last update: Oct. 2017)
%

     function I = dint(x,y)
%
     N = length(x);
     I = 0.0;
     for i = 1:(N-1)
       ym = (y(i+1)+y(i))/2;
       dx = x(i+1)-x(i);
       I = I + ym*dx;
     end
%
%   end of function
```