1. **Long Lines:**

   If a command is too long to fit on one line, you can split it over two or more lines using the triple dots command:

```
>> variable_with_a_very_long_name = ...
>> sqrt(9 * 16) - ...
>> sin(pi/6) + cosd(60) /...
>> 2^3^2 + tand(45) + tan(pi/4)
```

   The triple dots tell MATLAB that the expression is continued on the next line. This technique is also useful when evaluating expressions that have too many parentheses and other arithmetic operators.

2. **Operator precedence:**

   Evaluate the following algebraic expression in MATLAB:

$$\frac{\left(2+(\frac{1}{2})^2\right)-\left(1-(\frac{\sqrt{6}}{2\sqrt{2}})^2\right)}{\frac{9}{2}-\left(\frac{\sqrt{5}}{\sqrt{2}}\right)^2+\left((\frac{34}{25-8})^{3\times 2}\right)^{\frac{1}{2}}} \qquad Ans = 0.2$$

   Each of the following expressions gives the **wrong** answer. Your task is to reproduce the correct answer that is given on the RHS.

```
>> 384+551 / 66-711                    Ans: -1.4496
>> 39 * 56 * 72 / 48*19 * 32           Ans: 5.5175e+003
>> 37/97 ^ 1/3                         Ans: 0.7252
>> 58 / 9/7  ^ 0.5                     Ans: 6.7165
```

3. **Script M-Files & Display:**

   Consider the projectile motion covered in class notes. You will now create a MATLAB script m-file that calculates **for different launch angles ($\theta$) and launch speeds ($v$)** the maximum range of a projectile ($x_g$), the time taken ($t_g$), the maximum height reached ($y_m$) and the time taken to reach this height ($t_m$), where

$$t_g = \frac{2\,v\,\sin\theta}{g}, \quad x_g = v\,t_g\,\cos\theta, \quad t_m = \frac{v\,\sin\theta}{g}, \quad y_m = v\,t_m\,\sin\theta - \frac{1}{2}\,g\,t_m^2.$$

   As usual, the gravitational acceleration is $g = 9.8$m/s/s. For each of the two launch velocities $v = 50$m/s and $v = 100$m/s tabulate (using **disp** commands) $\theta$, $t_g$, $x_g$, $t_m$ and $y_m$ for each of the launch angles $0°$, $30°$, $45°$, $60°$, and $90°$. All computations should be carried out in degrees and **not** in radians.

   Each of your tables should look something like the following:

```
theta     t_g          x_g         t_m          y_m

30       5.102       220.9248      2.551       31.8878
```

   Here is how you are expected to proceed:

- Open a new m-file window (click File – New – M-File) or simply click on the blank file icon etc.
- Type-in the following lines verbatim, when you encounter four dots, fill in the relevant information.

```
% MyProjectile.m
% Calculates and tabulates projectile parameters
clc
clear

v = input('Enter the launch velocity (in m/s): ');
theta = input('Enter the launch angle (in degrees): ');
disp(' ') % blank line

g = ....;      % m/s/s

% compute results
t_g = ....;    % time to return to ground, (seconds)
x_g = ....;    % distance traveled, (metres)
t_m = ....;    % time to reach max height, (seconds)
y_m = ....;    % distance traveled, (metres)

% display results as suggested
disp([....])
disp(' ') % blank line
disp([....])
```

- **Saving the m-file:** Save this file (File – Save as) and name it **MyProjectile.m**
- **Running your m-file:** From within the Editor window, click on the solid green arrow. If you are prompted to change the Directory, click Yes. You should then be automatically transferred to the Command Window and prompted to enter the velocity etc.

  Debug your code as needed and if you are satisfied that your program is running as expected, go back to the Editor window and this time press F5. This is another way to run your program.

  Yet a third way to run your program is by simply typing the m-file name in the Command Window:

```
>> MyProjectile
```

- **Comments**

  Note that the symbol % can be used to add comments to the m-file. MATLAB ignores anything to the right of %.

  It is good practice to start your script m-files with several comment lines that explain what the program does.

  In the command window, if you then type **help** followed by the name of the script file, these comment lines will be printed to the screen

```
>> help MyProjectile
```

4. **Beam Bending:**

   A uniform beam is freely hinged at its ends $x = 0$ and $x = L$, so that the ends are at the same level. It carries a uniformly distributed load of weight $W$ per unit length, and there is a tension $T$ along the

$x$-axis. The deflection $y$ of the beam a distance $x$ from one end is given by

$$y = \frac{W\,E\,I}{T^2}\left[\frac{\cosh[a(L/2-x)]}{\cosh(a\,L/2)} - 1\right] + \frac{W\,x\,(L-x)}{2T},$$

where $a^2 = T/E\,I$, with $E$ being the Young's modulus of the beam, and $I$ is the moment of inertia of a cross-section of the beam. The beam is 10 m long, the tension is 1000 N, the load 100 N/m, and $E\,I$ is $10^4$.

Write a script file *Beam.m* that inputs a value of $x$ to compute the deflection $y$, MATLAB has an in-built cosh function. Run your script using several values of $x$, say $x = 0, 1, 2, 3, \ldots, L$.

Can you think of a way to display multiple values of $x$ against the corresponding deflections $y$? In fact the best way to display such results is via a graph. In this case we would need to write $x$ as a vector, then compute the corresponding vector $y$ and thus plot $y$ versus $x$. We will cover these issues in more detail later.