Chapter 2 Tutorial 1

The purpose of this tutorial is to teach you about variables in MATLAB.

```
% Always clear workspace variables before a new tutorial or program.
clear
```

Edit the code below and update the variable named **name** with your name for this tutorial in the code below.

```
name="";
fprintf("Output for Tutorial_02_1 run by %s.\n", name)
```

```
Output for Tutorial_02_1 written by .
```

Variables

Naming conventions

MATLAB variables and filenames must begin with a letter, and contain only letters, numbers, and underscores. The format of the name is generally in one of the following formats.

- **Camel case** - Multi-word variable names with with the first word in all lowercase, each subsequent word with the first letter capitalized. For example, a variable holding the value for top speed might be `topSpeed`.
- **Snake case** - Multi-word variable names with all words lowercase, each word separated by an underscore. For example, top speed would appear as `top_speed`.
- **Constants** - Values that are only used for reference and don't change within your code are generally in the form of snake case but all capitalized instead of lowercase. This is the general rule in programming however, MATLAB does not follow this convention though you may see it in custom programs.

Variable names should be descriptive and give general clues as to the data they store. For example, a good name for radius might be `radius` rather than just simply `r`.

MATLAB variables are case sensitive. Generally it is bad practice to name two variables the same and only change the case. However, in MATLAB it is common practice to declare scalars as a single lowercase letter and matrices/arrays, or vectors as a single uppercase letter.

```
A=[1 2 3]   % A 1x3 row vector using an uppercase letter
```

```
A = 1×3
     1     2     3
```

```
a=10        % A scalar using a lowercase letter
```

```
a = 10
```

Data Types

Variables can store numbers or characters. Unlike some other languages, MATLAB does not require you to declare the type of data being stored in a variable.

The following are common data types you may see

- Integer
- Floating point (decimal numbers)
- Character (a single letter)
- String (a vector or letters)
- Dates

```
payDay='Thursday'   % A vector of characters
```

```
payDay =
'Thursday'
```

```
GRAVITY=9.81        % A floating point value representing gravity in m/s^2
```

```
GRAVITY = 9.8100
```

Reserved Names

There are a number of "reserved" variable names provided to users for simplicity, these are variables that already exist in all workspaces and are provided by MATLAB libraries. Generally it is bad practice to create a variable with the name of a reserved word. The following are a few, more common, reserved variable names you'll encounter.

- inf - Infinity
- eps - The smallest number MATLAB can represent
- pi

```
fprintf("pi is already a MATLAB variable, the value of pi is %8.6f\n", pi);
```

```
pi is already a MATLAB variable, the value of pi is 3.141593
```

```
pi=10; % Overriding the reserved variable pi
fprintf("We can still override the value of pi, now the value of pi is %8.6f\n", pi);
```

```
We can still override the value of pi, now the value of pi is 10.000000
```

```
clear pi % Clear only the value of the variable pi
fprintf("Much better, pi is back to %8.6f\n", pi)
```

```
Much better, pi is back to 3.141593
```

Viewing Variable Data

If you would like to display information on the variables in your workspace, there are a few commands to do just that.

```
who     % Displays all workspace variables
```

```
Your variables are:

A        GRAVITY  a        name     payDay
```

```
whos    % Displays all workspace variables, their data types, sizes, and values
```

```
  Name         Size            Bytes  Class     Attributes

  A            1x3                24  double
  GRAVITY      1x1                 8  double
  a            1x1                 8  double
  name         1x1               148  string
  payDay       1x8                16  char
```

You can also view only certain variables using the `whos` command.

```
whos GRAVITY          % Show the whos information on GRAVITY
```

```
  Name         Size            Bytes  Class     Attributes

  GRAVITY      1x1                 8  double
```

```
whos GRAVITY payDay  % Show the whos information on GRAVITY, payDay, etc
```

```
  Name         Size            Bytes  Class     Attributes

  GRAVITY      1x1                 8  double
  payDay       1x8                16  char
```

Additional Notes:

- Variables should have comments explaining the values they hold