MATLAB: A Practical Introduction to Programming and Problem Solving

Fourth Edition

SOLUTION MANUAL

Stormy Attaway

College of Engineering
Boston University

Chapter 1: Introduction to MATLAB

Exercises

1) Create a variable *myage* and store your age in it. Subtract 2 from the value of the variable. Add 1 to the value of the variable. Observe the Workspace Window and Command History Window as you do this.

```
>> myage = 20;
>> myage = myage - 2;
>> myage = myage + 1;
```

2) Explain the difference between these two statements:

```
result = 9*2
result = 9*2;
```

Both will store 18 in the variable result. In the first, MATLAB will display this in the Command Window; in the second, it will not.

3) Use the built-in function **namelengthmax** to find out the maximum number of characters that you can have in an identifier name under your version of MATLAB.

```
>> namelengthmax
ans =
    63
```

4) Create two variables to store a weight in pounds and ounces. Use **who** and **whos** to see the variables. Use **class** to see the types of the variables. Clear one of them and then use **who** and **whos** again.

```
>> pounds = 4;
>> ounces = 3.3;
>> who

Your variables are:

ounces  pounds

>> whos
  Name        Size              Bytes  Class
Attributes

  ounces      1x1                   8  double
  pounds      1x1                   8  double
```

```
>> clear pounds
>> who

Your variables are:

ounces
```

5) Explore the **format** command in more detail. Use **help format** to find options. Experiment with **format bank** to display dollar values.

```
>> format +
>> 12.34
ans =
+
>> -123
ans =
-
>> format bank
>> 33.4
ans =
         33.40
>> 52.435
ans =
         52.44
```

6) Find a **format** option that would result in the following output format:

```
>> 5/16 + 2/7
ans =
     67/112
```

```
>> format rat
>> 5/16 + 2/7
ans =
      67/112
```

7) Think about what the results would be for the following expressions, and then type them in to verify your answers.

```
25 / 5 * 5
4 + 3 ^ 2
(4 + 3) ^ 2
3 \ 12 + 5
4 – 2 * 3
```

```
>> 25/5*5
ans =
    25
>> 4 + 3 ^ 2
ans =
    13
>> (4 + 3) ^ 2
ans =
    49
>> 3 \ 12 + 5
ans =
     9
>> 4 - 2 * 3
ans =
    -2
```

As the world becomes more "flat", it is increasingly important for engineers and scientists to be able to work with colleagues in other parts of the world. Correct conversion of data from one system of units to another (for example, from the metric system to the American system or vice versa) is critically important.

8) Create a variable *pounds* to store a weight in pounds. Convert this to kilograms and assign the result to a variable *kilos*. The conversion factor is 1 kilogram = 2.2 lb.

```
>> pounds = 30;
>> kilos = pounds / 2.2
kilos =
   13.6364
```

9) Create a variable *ftemp* to store a temperature in degrees Fahrenheit (F). Convert this to degrees Celsius (C) and store the result in a variable *ctemp*. The conversion factor is C = (F – 32) * 5/9.

```
>> ftemp = 75;
>> ctemp = (ftemp - 32) * 5/9
ctemp =
   23.8889
```

10) The following assignment statements either contain at least one error, or could be improved in some way. Assume that *radius* is a variable that has been initialized. First, identify the problem, and then fix and/or improve them:

```
33 = number

     The variable is always on the left
     number = 33
```

```
my variable =  11.11;

     Spaces are not allowed in variable names
     my_variable = 11.11;

area = 3.14 * radius^2;

     Using pi is more accurate than 3.14
     area = pi * radius^2;

x = 2 * 3.14 * radius;

     x is not a descriptive variable name
     circumference  =  2 * pi * radius;
```

11) Experiment with the functional form of some operators such as **plus**, **minus**, and **times**.

```
>> plus(4, 8)
ans =
    12
>> plus(3, -2)
ans =
     1
>> minus(5, 7)
ans =
    -2
>> minus(7, 5)
ans =
     2
>> times(2, 8)
ans =
    16
```

12) Generate a random

- real number in the range (0, 20)

```
rand * 20
```

- real number in the range (20, 50)

```
rand*(50-20)+20
```

- integer in the inclusive range from 1 to 10

```
randi(10)
```

- integer in the inclusive range from 0 to 10

```
randi([0, 10])
```

- integer in the inclusive range from 50 to 100

```
randi([50, 100])
```

13) Get into a new Command Window, and type **rand** to get a random real number. Make a note of the number. Then, exit MATLAB and repeat this, again making a note of the random number; it should be the same as before. Finally, exit MATLAB and again get into a new Command Window. This time, change the seed before generating a random number; it should be different.

```
>> rand
ans =
0.8147

>> rng('shuffle')
>> rand
ans =
0.4808
```

14) What is the difference between x and ‘x’?

`In an expression, the first would be interpreted as the name of a variable, whereas ‘x’ is the character x.`

15) What is the difference between 5 and ‘5’?

`The first is the number 5, the second is the character 5. (Note: int32(5) is 53. So, 5+1 would be 6. ‘5’+1 would be 54.)`

16) The combined resistance R_T of three resistors R_1, R_2, and R_3 in parallel is given by

$$R_T = \frac{1}{\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_3}}$$

Create variables for the three resistors and store values in each, and then calculate the combined resistance.

```
>> r1 = 3;
>> r2 = 2.2;
>> r3 = 1.5;
>> rt = 1/(1/r1 + 1/r2 + 1/r3)
rt =
    0.6875
```

17) Explain the difference between constants and variables.

Constants store values that are known and do not change. Variables are used when the value will change, or when the value is not known to begin with (e.g., the user will provide the value).

18) What would be the result of the following expressions?

```
'b' >= 'c' - 1           1

3 == 2 + 1               1

(3 == 2) + 1             1

xor(5 < 6, 8 > 4)        0

10 > 5 > 2

  0  Evaluated from left to right: 10>5 is 1,
     then 1 > 2 is 0

result = 3^2 - 20;
0 <= result <= 10

     1    Evaluated left to right: 0 <= result is 0,
     then 0 <= 10 is 1
```

19) Create two variables *x* and *y* and store numbers in them. Write an expression that would be **true** if the value of *x* is greater than five or if the value of *y* is less than ten, but not if both of those are **true**.

```
>> x = 3;
>> y = 12;
>> xor(x > 5, y < 10)
ans =
     0
```

20) Use the equality operator to verify that 3*10^5 is equal to 3e5.

```
>> 3*10^5 == 3e5
```

```
ans =
     1
```

21) In the ASCII character encoding, the letters of the alphabet are in order: ‘a’ comes before ‘b’ and also ‘A’ comes before ‘B’. However, which comes first - lower or uppercase letters?

```
>> int32('a')
ans =
          97
>> int32('A')
ans =
          65

The upper case letters
```

22) Are there equivalents to **intmin** and **intmax** for real number types? Use **help** to find out.

```
>> realmin
ans =
  2.2251e-308
>> realmin('double')
ans =
  2.2251e-308
>> realmin('single')
ans =
  1.1755e-38
>> realmax
ans =
  1.7977e+308
```

23) Use **intmin** and **intmax** to determine the range of values that can be stored in the types **uint32** and **uint64**.

```
>> intmin('uint32')
ans =
           0
>> intmax('uint32')
ans =
  4294967295
>> intmin('uint64')
ans =
                    0
>> intmax('uint64')
ans =
 18446744073709551615
```

24) Use the **cast** function to cast a variable to be the same type as another variable.

```
>> vara = uint16(3 + 5)
vara =
      8
>> varb = 4*5;
>> class(varb)
ans =
double
>> varb = cast(varb, 'like', vara)
varb =
     20
>> class(varb)
ans =
uint16
```

25) Use **help elfun** or experiment to answer the following questions:

- Is **fix(3.5)** the same as **floor(3.5)**?

```
>> fix(3.5)
ans =
     3
>> floor(3.5)
ans =
     3
```

- Is **fix(3.4)** the same as **fix(-3.4)**?

```
>> fix(3.4)
ans =
     3
>> fix(-3.4)
ans =
    -3
```

- Is **fix(3.2)** the same as **floor(3.2)**?

```
>> fix(3.2)
ans =
     3
>> floor(3.2)
ans =
     3
```

- Is **fix(-3.2)** the same as **floor(-3.2)**?

```
>> fix(-3.2)
ans =
    -3
>> floor(-3.2)
ans =
    -4
```

- Is **fix(-3.2)** the same as **ceil(-3.2)**?

```
>> fix(-3.2)
ans =
    -3
>> ceil(-3.2)
ans =
    -3
```

26) For what range of values is the function **round** equivalent to the function **floor**?
For positive numbers: when the decimal part is less than .5
For negative numbers: when the decimal part is greater than or equal to .5

For what range of values is the function **round** equivalent to the function **ceil**?
For positive numbers: when the decimal part is greater than or equal to .5
For negative numbers: when the decimal part is less than .5

27) Use **help** to determine the difference between the **rem** and **mod** functions.

```
>> help rem
 rem    Remainder after division.
    rem(x,y) is x - n.*y where n = fix(x./y) if y ~= 0.
    By convention:
       rem(x,0) is NaN.
       rem(x,x), for x~=0, is 0.
       rem(x,y), for x~=y and y~=0, has the same sign as x.

rem(x,y) and MOD(x,y) are equal if x and y have the same
sign, but differ by y if x and y have different signs.

>> help mod
 mod    Modulus after division.
```