LESSON 13: Program control questions

FOCUS QUESTION: How can I execute different code depending on the data?

Contents

EXAMPLE 1: Simulate tossing a coin (selection using `if-else`)

```
toss = rand(1, 1);

if toss <= 0.5

    fprintf('Tossed heads\n');

else

    fprintf('Tossed tails\n');

end;
```

Questions	Answers
Why does the output change when I execute this cell multiple times?	The `if` statement gives alternative paths of execution, and only one of the two `fprintf` statements executes each time. The branch is determined by the "random" value generated by `rand`.
Does `rand` produce truly "random" values?	The `rand` function generates "pseudorandom" values that are distributed uniformly in (0, 1). The values are not correlated and so they can simulate picking numbers at random or tossing a coin. However, they appear in a well-defined, repeatable sequence so they are not truly random. The

Questions	Answers
	fact that the sequence is repeatable is very useful for repeating experiments.
Does `rand` ever generate a value that is exactly 0 or exactly 1?	No, the output of `rand` is strictly between 0 and 1, exclusive.
What do the parameters of the `rand` function do?	The `rand` function generates an array of random numbers and the parameters specify the size of the array. In this example, `rand` generates a single value (i.e., a 1 x 1 array). The call `rand(2, 3)` generates an array with 2 rows and 3 columns.
Can each branch of the `if` contain more than one statement?	Yes, you can put any number of statements in each section.
What happens after the selected branch of the `if` executes?	MATLAB continues execution with the statement after the `end`.

```
Tossed tails
```

EXAMPLE 2: Output the square roots of first 3 integers (simple `for` loop)

```
for k = 1:3

  fprintf('sqrt(%g) = %g\n', k, sqrt(k));

end;
```

Questions	Answers
How many times is this loop executed?	MATLAB executes statements between the `for` and the `end` 3 times. Each time, the loop variable `k` takes on one of the values in the list: 1, 2, 3.
Does a loop body always contain just one statement?	No, you can put as many statements as you want between the `for` and the `end`.
Do I always have to use `k` for the loop variable?	No, you can use any loop variable name that you want.
Does the loop variable have to take on consecutive integral values?	No, you can specify any list of values (e.g., `for x = [1.3, 2.2, 4.2, 5.1]`).
Can I modify the loop variable inside the loop?	Technically you can. However, it is highly recommended that you do not.
What is the value of the loop variable after the loop completes?</strong>?	Assuming that you following good programming practice and do not modify the loop variable inside the loop, the loop variable will have the last value in the list. For this example, the loop variable `k` will have the value 3.

Questions	Answers
What happens after the loop finishes?</strong>?	MATLAB continues execution with the statement after the `end`.

```
sqrt(1) = 1

sqrt(2) = 1.41421

sqrt(3) = 1.73205
```

EXAMPLE 3: Sum the square roots of the first 10 integers (accumulation using a `for` loop)

```
    sumSqrts = 0;

    for k = 1:10

      sumSqrts = sumSqrts + sqrt(k);

    end;

    fprintf('Sum of square roots is %g\n', sumSqrts);
```

Questions	Answers
What does the statement `sumSqrts = sumSqrts + sqrt(k)` mean?	Recall that the equals (=) is the assignment operator, not test for equality. This statement evaluates the right hand side by adding the current value of `sumSqrts` and the square root of `k`. MATLAB assigns the result to the variable on the left of the `=`. In this case `sumSqrts` receives the result.
What does the statement `sumSqrts = sumSqrts + sqrt(k)` actually do?	This statement accumulates a sum (of the square roots of the first 10 integers).
What happens if I remove the `sumSqrts = 0` statement?	MATLAB will terminate with an error the first time through the loop because the assignment statement requires that `sumSqrts` have a value.
What happens if I move the `sumSqrts = 0` statement inside the body of the `for` loop?	MATLAB sets `sumSqrts` to 0 each time and so instead of accumulating a sum, `sumSqrts` only has the square root of 10 on loop completion.

```
Sum of square roots is 22.4683
```

EXAMPLE 4: Simulate tossing coin 50 times (`for` loop with selection and accumulation)

```
    numTosses = 50;

    numHeads = 0;

    for k = 1:numTosses

       if rand(1, 1) <= 0.5

           numHeads = numHeads + 1;

        end;

    end;

    fprintf('%g heads in %g tosses\n', numHeads, numTosses);
```

Questions	Answers
How many times is this loop executed?	MATLAB executes this loop 50 times, one for each value of `k` in the list 1, 2, ..., 50.
How many times is the statement `numHeads = numHeads + 1` executed?	MATLAB only executes this statement when the result of `rand` is less than or equal to 0.5. On average, this happens half the time. However, results will vary each time you execute the cell, just as the results of tossing a coin 50 times will vary.
Which `end` statement closes the loop?	The last `end` statement closes the `for` loop. The other `end` statement finishes the `if`.

```
27 heads in 50 tosses
```

EXAMPLE 5: Alternative implementation of coin toss simulation (vector indexing)

```
    timesToTosses = 50;

    randTosses = rand(timesToTosses,1);

    numHeads = sum(randTosses <= 0.5);

    fprintf('%g heads in %g tosses\n', numHeads, numTosses);
```

Questions	Answers

Questions	Answers
How big is `randTosses`?	The `randTosses` variable has 50 rows and 1 column.
How big is `randTosses <= 0.5`?	The result is a logical array with 50 rows and 1 column. The entries are 1's when the corresponding entries of `randTosses` are less than or equal to 0.5.
Why does `sum(randTosses <= 0.5)` give the number of heads?	The result counts the number of values in `randTosses` that are less than or equal to 0.5. We have assigned the values of `rand` that are less than or equal to 0.5 to mean heads.
Could I choose heads to be sum(randTosses > 0.5) to mean heads?	Yes, the assignment is arbitrary as long as you designate half of the values in (0, 1).

```
20 heads in 50 tosses
```

EXAMPLE 6: Load the sleep diary data

```
load diaries.mat;
```

EXAMPLE 7: Output a message if any subjects awoke after 3:30 pm

```
wakeHours = (wakeTimes - floor(wakeTimes))*24;

lateWakeup = sum(sum(wakeHours > 15.5));

if  lateWakeup > 0

    fprintf('%g wake-ups after 3:30 pm\n', lateWakeup);

end;
```

Questions	Answers
What is the size of `wakeHours`?	The `wakeHours` variable has 21 rows and 144 columns.
What is the size of `wakeHours > 15.5`?	The variable holds an array that is the same size as `wakeHours`. It has 1's where the corresponding entries of `wakeHours` are greater than 15.5 and 0's elsewhere.
Why do I need two `sum` functions to find the total?	Since `wakeHours > 15.5` is a 21 x 144 array, `sum(wakeHours > 15.5)` is a row vector size 1 x 144 containing the column sums. To find the overall total, you need to add up these column sums. Hence, the nested `sum` functions.

```
30 wake-ups after 3:30 pm
```

EXAMPLE 8: Output subject number and gender for subjects with at least 1 wake-up after 3:30 pm

Create a new cell in which you type and execute:

```
timesLate = sum(wakeHours > 15.5);

fprintf('Subjects who had a least one wake-up after 3:30 pm:\n');

for k = 1:length(timesLate)

    if timesLate(k) > 0

      fprintf('Subject %g: a %s with %g late wake-ups\n', ...

        k, gender{k}, timesLate(k));

    end;

end;
```

```
Subjects who had a least one wake-up after 3:30 pm:

Subject 2: a female with 1 late wake-ups

Subject 7: a male with 1 late wake-ups

Subject 8: a female with 1 late wake-ups

Subject 40: a female with 1 late wake-ups

Subject 46: a male with 3 late wake-ups

Subject 66: a male with 1 late wake-ups

Subject 70: a female with 1 late wake-ups

Subject 71: a female with 1 late wake-ups

Subject 73: a female with 1 late wake-ups

Subject 86: a female with 5 late wake-ups
Subject 101: a male with 4 late wake-ups

Subject 118: a female with 3 late wake-ups

Subject 125: a male with 2 late wake-ups

Subject 134: a female with 3 late wake-ups

Subject 142: a female with 2 late wake-ups
```

EXAMPLE 9: Output the subject number and gender of the first student in section 3 (break)

```
    sect3 = section == 3;

    for k = 1:length(sect3)

       if sect3(k)

            fprintf('First in section 3 is a %s with subject number %g\n', ...

                gender{k}, k);

            break;

       end;

    end;
```

Questions	Answers
What does the `break` do?	The `break` provides a controlled way to get out of a loop without going through all of the iterations. The `break` causes the statement after the innermost enclosing loop to be the next one executed.
Does `break` transfer control out of all loops?	No, `break` only "breaks" out of the innermost enclosing loop.
Does it matter that the `break` appears in an `if-else`?	No, the `break` is not affected by `if-else`. However, you will often see `break` as part of an `if-else` because you will usually only want to "break" out of the loop under certain circumstances.

```
First in section 3 is a female with subject number 2
```

EXAMPLE 10: Output a table of early wake-ups using a loop

```
averWake = mean(wakeHours);

earlyWake = 6;

fprintf('\n\n\tEarly wake-ups\n');

fprintf('Subj\tSect\tGender\tAver Wakeup\n');

for k = 1:length(averWake)

   if averWake(k) >= earlyWake

       continue;

   end;

   fprintf(' %g\t  %g\t%s\t  %5.2f\n', k, section(k), gender{k}, averWake(k));

end;
```

```

        Early wake-ups

Subj     Sect     Gender   Aver Wakeup

 32        1      male        5.87

 91        2      female      5.50

 140       3      female      5.68
```

This lesson was written by Kay A. Robbins of the University of Texas at San Antonio and last modified on 05-Nov-2012. Please contact krobbins@cs.utsa.edu with comments or suggestions.

Published with MATLAB® 7.14