Learning Objectives

1. Explain how conditional statements (e.g., if-end, if-else-end, switch-case) are used to make decisions

Topics

Read Chapter 6.2 – 6.3 of the MATLAB book before coming to class. This preparation material is provided to supplement this reading.

Students will learn a basic understanding of conditional statements. This knowledge will be used to create if statements and proper if-end logic in MATLAB. This material contains the following:

1. Key Definitions
2. Conditional Statements
3. Forms of the ‘if’ Statement
4. Using the ‘if-elseif-else-end’ Statement
5. Summary of ‘if-end’ Statements
6. Switch-Case

1. Key Definitions

**Relational Operator** – compares two numbers by determining whether a comparison statement is true or false. Operators include: ‘less than <’, ‘greater than >’, ‘less than or equal to <=’, ‘greater than or equal to >=’, ‘equal to ==’, ‘not equal to ~=’.

**Logical Operator** – examines true/false statements and produces a result that is true (1) or false (0) according to the specific operator. Operators include ‘AND’ (&), ‘OR’ (|), ‘NOT’ (~).

**Order of Precedence** – computers and calculators follow a hierarchy for the order in which operations are executed. Higher-precedence operations are executed before lower-precedence operations; when they are equal, they are executed from left to right.

**Conditional Statements** – a command that allows a program to make a decision of whether to execute a set of commands or skip those commands.
2. Conditional Statements

- Conditional statements enable MATLAB to make decisions.
- The process is similar to the way we (humans) make decisions.
- A condition is stated. If the condition is met, one set of actions is taken. If the condition is not met, either nothing is done, or a second set of actions is taken.

Real-life example:

If I win the Lottery,

- I will quit college, buy a new car, and go fishing.

If I do not win the Lottery,

- I will study harder so that I can get a better job.

The first step in programming is to identify what the condition is that is being tested (for TRUE or FALSE).

Examples of ‘if’ statements:

```matlab
if a < b
if c >= 5
if a == b
if a ~= 0
if (d < h) & (x > 7)
if (x ~= 13) | (y < 0)
```

All variables must already have assigned values.
3. Forms of the ‘if’ Statement

The most basic conditional statement is an ‘if’ statement with the conditional test, what command to execute if it is true, and an ‘end’. This format will only execute commands if that conditional statement is TRUE; otherwise, there is no further action.

```
if  conditional statement
    command group
end
```

**Example:**
```
if the temperature is below 65 degrees
    turn on the furnace
end
```

The next layer could be adding an ‘else’ statement and a 2\textsuperscript{nd} command group. This still includes what to execute if the first condition is met, but now also includes what to execute if that condition is NOT met. There is now a command group for TRUE and a command group for all other outcomes that make the condition FALSE.

```
if  conditional statement
    command group 1
else
    command group 2
end
```

**Example:**
```
if the temperature is below 65 degrees
    turn on the furnace
else
    turn off the furnace
end
```

**NOTE:** Whether furnace is already ON or OFF, it will take the proper action based on the condition.

**Remember that including an ‘else’ as a catch-all is an optional item. It is up to the programmer to have no action listed if FALSE, or execute something else.**
The final layer(s) comes from adding one or more 'elseif' statements into the program. This allows multiple tests and conditions to be included. Again, the final 'else' statement is optional.

```matlab
if  conditional statement 1
   command group 1
elseif  conditional statement 2
   command group 2
...
(Optional)
elseif  conditional statement #
   command group #
...
(Optional)
else    %no conditional statement
   command group n+1
...
end
```

**Example:**
```
if the temperature is below 65 degrees
   turn on the furnace
elseif the temperature is above 78 degrees
   turn on the A/C
else
   open the windows
end
```
4. Using the ‘if-elseif-else-end’ Statement

Recall from Computer Problem Solving, that we can use a flow chart to outline decisions and actions:

As such, the flowchart can then be used as a roadmap for condition expressions:

```
......
......
    if    conditional expression
          .......
          Command Group 1
          .......
    elseif   conditional expression
          .......
          Command Group 2
          .......
    else
          .......
          Command Group 3
          .......
    end
......
```
For example, let’s say we were asked to write a program that calculates the tip based on amount of bills, using the following rules and the variable ‘bill’:

- bill is less than $10
  - Tip is $1.80
- bill is between $10 and $60
  - Tip is %18
- bill is above $60
  - Tip is %20

We could make a flowchart like the following:
5. Summary of ‘if-end’ Statements
A quick recap of ‘if-end’ statements include:

- Every ‘if’ command must have an ‘end’ command.
- A program can have many ‘if ... end’ statements following each other, or nested within each other.
- A computer program can perform the same task using different combinations of ‘if – end’, ‘if – else – end’, and ‘if– elseif – else – end’ statements.
- Multiple ‘elseif’ conditions are allowed within an ‘if– elseif – else – end’ statement.
- An ‘else’ condition is not required.
  - When ‘else’ is used, a conditional statement is NOT added.
6. Switch-Case

Switch-Case is a form of a conditional statement and “if-end” loop with syntax differences:

- switch-case is similar to ‘if-elseif-end’, but conditions must be an exact match.
- You cannot use switch-case with a range, such as <0.
- Upper-case vs. lower-case matters, as done extra spaces
- switch on either scalar variable or a string (identifies test variable)
  - e.g. switch x
- ‘case’ is similar to ‘if’ statement and requires a variable condition that is then followed on
  the next line by the outcome (command) if the condition is TRUE
- ‘otherwise’ is similar to ‘else’, and is also optional

```matlab
x = input (’Resistance: ’);
switch x
    case 100
        out= x;
    case 200
        out= 2*x;
    case 300
        out= 3*x;
    case 400
        out= 4*x;
    otherwise
        out = 0;
end
```
x = input ('Purchase Class: ','s');
x = upper(x) % change to upper case
switch x
    case 'A'
        rate = 2.00;
    case 'B'
        rate = 3.00;
    case 'C'
        rate = 3.50;
    case 'D'
        rate = 4.00;
    otherwise
        rate = 0;
end