Module 03: Selection Statement

Chul Min Yeum

Assistant Professor

Civil and Environmental Engineering

University of Waterloo, Canada

UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Module 3: Intended Learning Outcomes

- Describe a problem that requires if statement
- Construct an if-else statement and explain its operation
- Compare an if-else statement and its variants
- Create a script using a switch statement

If-Statement

- The **if** statement is used to determine whether or not a statement or group of statements is to be executed
- General form:

```
if condition
    action
end
```

- the *condition* is any relational expression (True or False)
- the *action* is any number of valid statements (including, possibly, just one)

If the condition is true, the action is executed – otherwise, it is skipped entirely.

Example: Relational Operator

Q. How to write a code to check if `x` lies in between `5` and `10`. If yes, assign `1` to `lg1` and otherwise `0`.

```
x1 = 6;

lg1 = (5 < x1) && (x1<10)
```

```
x1 = 6;
lg1 = false;

if (5 < x1) && (x1<10)
   lg1 = true
end
```

Q. How to write a code to check if `x` lies in between `5` and `10`. If yes, assign `10` to `val` and otherwise `5`.

```
x1 = 6;

val = 5;

if (5 < x1) && (x1<10)
   val = 10;
end
```

Example: Write a Code for Computing abs(x)

abs(x)	Finds the absolute value of **x**	**abs(-3)**
		abs(2)
			3
			2

```
1   x1 = -3
2
3   if x1>=0
4       xabs = x1;
5   end
6
7   if x1<0
8       xabs = x1*-1;
9   end
10
```

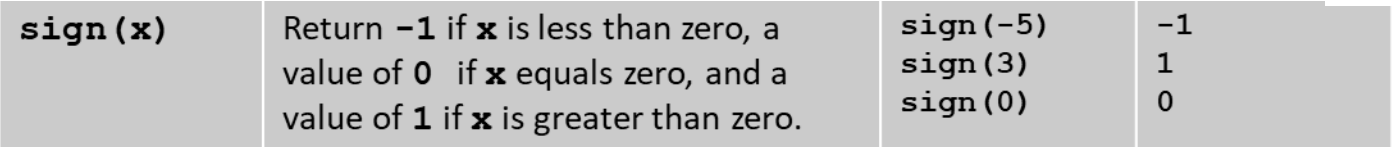
Name	**Value**
x1	**-3**
xabs	**3**

📖: At line 4, x1 is not more than 0, and thus, the if-statement is skipped (move to line 8) . At line 8, since x1 is less than 0, the action (line 9) is executed. x1*-1 is assigned to xabs.

```
x1 = -3
xabs = x1

if x1<0
    xabs = x1*-1;
end
```

This code also works.

Example: Write a Code for Computing `sign(x)`



sign(x)	Return **-1** if **x** is less than zero, a value of **0** if **x** equals zero, and a value of **1** if **x** is greater than zero.	sign(-5)
		sign(3)
		sign(0)
			-1
			1
			0

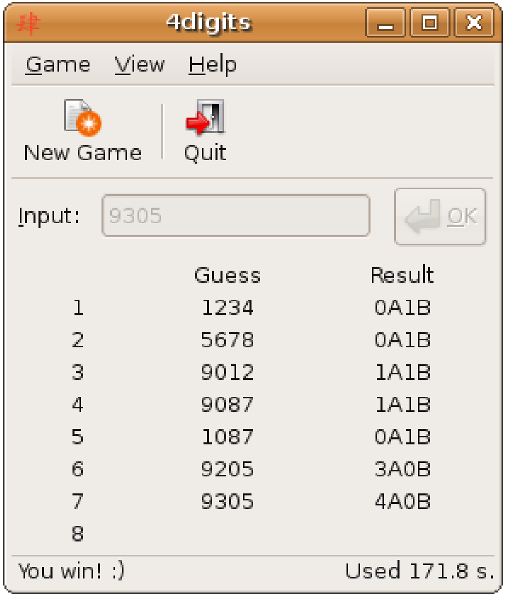
```
x1 = -3

if x1 == 0
   xsign = 0 ;
end

if x1<0
   xsign = -1;
end

if x1>0
   xsign = 1;
end
```

```
x1 = -3
xsign = 0;

if x1<0
   xsign = -1;
end

if x1>0
   xsign = 1;
end
```

Name	**Value**
x1	**-3**
xsign	**-1**

Example: Bulls and Cows

Bulls and Cows is a mind game played by two players. In the game, a random, 4-digit number is chosen and its values are compared to those of another trial number. **All four digits of the number are different.** If any digit in the chosen number is the exact same value and in the exact same position as any digit in the trial number, this is called a bull. If the digit is present in both the trial number and chosen number, but is not in the same location, this is called a cow.



Input: 9305

	Guess	Result
1	1234	0A1B
2	5678	0A1B
3	9012	1A1B
4	9087	1A1B
5	1087	0A1B
6	9205	3A0B
7	9305	4A0B
8		

You win! :)

Used 171.8 s.

In this figure, A: Bulls, B: Cows

Example: Bulls and Cows (Continue)

```
x_true = [1 2 3 4]; % true
x_test = [3 2 5 6]; % test

numb = 0; % number of Bull

if x_true(1) == x_test(1)
   numb = numb + 1;
end

if x_true(2) == x_test(2)
   numb = numb + 1;
end

if x_true(3) == x_test(3)
   numb = numb + 1;
end

if x_true(4) == x_test(4)
   numb = numb + 1;
end
```

Q: Write a script to compute “Bull” and assign its value to `numb`. The true and test sequence are in `x_true` and `x_test`, respectively.

Name	**Value**
x1	**[1 2 3 4]**
x2	**[3 2 5 6]**
numb	**1**

☺: We will keep revisiting these examples later!!

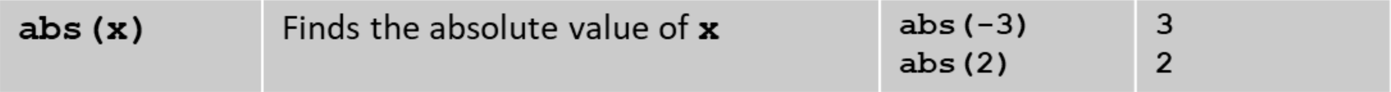
If-else Statement

- The **if-else** statement chooses between two actions
- General form:

```
if condition
    action1
else
    action2
end
```

- ***Only one action is executed***; which one depends on the value of the condition. In the above statement, execute `action1` if `condition` is `true` or `action2` if `condition` is `not true` (`false`).

Example: If-else Statement



abs(x)	Finds the absolute value of **x**	abs(-3)
		abs(2)
			3
			2

```
x1 = -3

if x1>=0
   xabs = x1;
end

if x1<0
   xabs = x1*-1;
end
```

Name	**Value**
x1	**-3**
xabs	**3**

```
x1 = -3
xabs = 0

if x1>0
   xabs = x1;
else
   xabs = x1*-1;
end
```

Name	**Value**
x1	**-3**
xabs	**3**

☺: In this example, there is no difference in the final result. However, I recommend if-else statement because `abs(x)` is an obvious logical (binary) operation, meaning that if `action1` is performed, `action2` will not be executed.

Nested if-else Statements

- To choose from more than two actions, ***nested*** **if-else** statements can be used (an **if** or **if-else** statement as the action of another)

- General form:

```
if condition1
    action1
else
    if condition2
       action2
    else
       action3
    end
end
```

📖: In this statement, only one action statement is executed!!

Recall the if-else statement:

```
if condition
    action1
else
    action2
end
```

Example: Nested if-else Statements

Q: If ‘scalar1’ is larger than 0 and less than 50, assign 10 to ‘out1’. Otherwise, assign 5 to ‘out1’.

```
scalar1 = 20;

if scalar1 > 0
   if scalar1<50
      out1 = 10;
   else
      out1 = 5;
   end
else
   out1 = 5;
end
```

```
scalar1 = 20;

if (scalar1 > 0) && (scalar1 < 50)
   out1 = 10;
else
   out1 = 5;
end
```

☺**:** In general, I try to avoid multiple levels of nested if-else statement. The script on the right is more readable and avoids potential mistakes.

if-else and elseif Statements

- MATLAB also has an `elseif` clause which shortens the code (and cuts down on the number of ends)
- General form:

```
1   if condition1
2         action1
3   elseif condition2
4         action2
5   elseif condition3
6         action3
7   else
8         action4
9   end
```

📖: Again, in this statement, only one action statement must be executed!!

If `condition1` is `true`, `action1` is executed and go to `end` at line9. If `condition1` is `false`, go to `condition2` and execute `action2` if `condition2` is `true`. The same operation is performed for `condition3` and `action3`. If none of `condition1, 2, 3` is `true`, `action4` is executed.

Example: if-else and elseif Statements

Q: Write a script to determine a grade based on a score named `score`: A: score >=90, B: 80 <= score < 90, C:70 <= score < 80, D: score < 70. A character grade should be assigned to a variable named `grade`.

```
score = 81;

if score>= 90
    grade = 'A';
end
if score>=80 && score < 90
    grade = 'B';
end
if score>=70 && score < 80
    grade = 'C';
end
if score < 70
    grade = 'D';
end
```

Option 1

```
score = 81;

if score>= 90
    grade = 'A';
else
    if score >=80
        grade = 'B';
    else
        if score >=70
            grade = 'C';
        else
            grade = 'D';
        end
    end
end
```

Option 1

Example: if-else and elseif Statements (Continue)

Q: Write a script that a grade based on a score named `score`: A: score >=90, B: 80 <= score < 90, C:70 <= score < 80, D: score < 70. A character grade should be assigned to a variable named `grade`.

```
score = 81;

if score>= 90
    grade = 'A';
end
if score>=80 && score < 90
    grade = 'B';
end
if score>=70 && score < 80
    grade = 'C';
end
if score < 70
    grade = 'D';
end
```

Option 1

```
score =81;

if score>= 90
    grade = 'A';
elseif score >=80
    grade = 'B';
elseif score >=70
    grade = 'C';
else
    grade = 'D';
end
```

Option 3

☺: This is simpler and more readable.

Correct Use of if-else-elseif Statement

```
score =81;

if score>= 90
    grade = 'A';
elseif score >=70
    grade = ‘C';
elseif score >=80
    grade = ‘B';
else
    grade = 'D';
end
```

Option 3

⚠: If the order of the condition statements is changed, a total wrong result will be obtained. Here, `grade` become `‘C’`, not `‘B’`. Again, only one action is executed!

```
score =81;

if score>= 90
    grade = 'A';
elseif and(score < 90, score>=80)
    grade = 'B';
elseif and(score < 80, score>=70)
    grade = 'C';
else
    grade = 'D';
end
```

Option 4

☺: You can also define a clear condition statement to avoid confusion.

Summary: if, if-else, if-elseif, if-elseif-else

```
if condition1
    action1
end
```

```
if condition1
    action1
else
    action2
end
```

```
if condition1
    action1
elseif condition2
    action2
end
```

```
if condition1
    action1
elseif condition2
    action2
else
    action3
end
```

How to Write a Formatted Script

- Use indentation to show the structure of a script or function for readability. Four empty spaces are indented in an action statement.
- The best way to format your script is to apply smart indenting while writing your script. Select your script and press ***Ctrl + I***.

```
x1 = -3
xabs = 0

if x1>0
   xabs = x1;
else
   xabs = x1*-1;
end
```

```
x1 = -3
xabs = 0
if x1>0
xabs = x1;
else
xabs = x1*-1;
end
```

📖: Both are the same and working scripts. Which do you prefer?

Shorten Your Scripts

Optional, Challenge

```
x1  = -3
xabs = 0

if x1>=0
   xabs = x1;
end

if x1<0
   xabs = x1*-1;
end
```

Option 1

```
x1  = -3;
xabs = 0;

if x1>=0
    xabs = x1;
else
    xabs = x1*-1;
end
```

Option 2

abs(x)	Finds the absolute value of **x**

```
x1  = -3
xabs = (x1>=0)*x1 + -1*(x1<0)*x1
```

Option 3

🕮: Here, `x1` is negative. Thus, `(x1>=0)` becomes `logical 0`. Due to type casting, `0*x1` becomes `0` (type: `double`). In the next term, `(x1<0)` becomes `logical 1`. Then, `-1*(x1<0)*x1` becomes `-1*x1`. Finally, `-1*x1` is assigned to `xabs`, which is the same operation using if-statement. Please think about when x1 is positive.

Switch-case Statement

- The **switch** statement can frequently be used in place of a nested **if-else** statement
- This can be used when comparing the switch_expression to see if it is equal to the values on the case labels (the **otherwise** clause handles all other possible values)

General form:

```
switch switch_expression
  case caseexp1
    action1
  case caseexp2
    action2
  case caseexp3
    action3
  % etc: there can be many of these n actions
  otherwise
    actionn
end
```

☺: In practice, `switch-case` statement is used more with character (or character vector than with numbers.

Example: switch-case Statements

```
grade = 'A';

if grade == 'A'
    disp('Your score is in the range of 90-100.');
elseif grade == 'B'
    disp('Your score is in the range of 80-90.');
elseif grade == 'C'
    disp('Your score is in the range of 70-80.');
elseif grade == 'D'
    disp('Your score is below 70.');
else
    disp('We do not have such grade.')
end
```

Q: Write a script to tell you the score range when your input is a grade. `disp()` is a function to print out text in a command window.

```
grade = 'A';

switch grade
    case 'A'
        disp('Your score is in the range of 90-100.');
    case 'B'
        disp('Your score is in the range of 80-90.');
    case 'C'
        disp('Your score is in the range of 70-80.');
    case 'D'
        disp('Your score is below 70.');
    otherwise
        disp('We do not have such grade.');
end
```

Throwing an Error

Optional

- MATLAB has an `error` function that can be used to display an error message in red, similar to the error messages generated by MATLAB or generated by MATLAB grader.
- When an error is thrown in a script, the script stops executing

```
if your_var ~= model_var
    error('Sorry: your value is not correct');
else
    disp('Pass the test');
end
```

- `assert` function is to throw error if condition is `false`.
- `assert(cond, msg) throws an error if cond is false.`

```
assert(your_var == model_var, 'Sorry: your value is not correct');
```