Lab Objectives

By the end of this lab you should be able to:

1. Create a program that properly uses if/else commands.
2. Use a while-loop to perform multi-step calculations.

if/else Statement

One of the main tools used by programmers is the if/else statement. The if/else statement is a simple way to direct the behavior of your program based on the truth of some condition. The basic syntax of the if/else statement is as follows:

```matlab
if (condition)
    some action
elseif (condition)
    some action
else
    some action
end
```

You can read the syntax of the if/else statement in the same way you would tell someone of some action you might perform. For example, “If the it is sunny out, then I will go to the beach.” If the condition “it is sunny out” is true, then I will perform the action of going to the beach. The elseif and else parts of the statement are optional and are only executed if the initial condition is false. Multiple elseif parts can be included in an if/else statement as needed.

The condition(s) for an if/else statement can be any relationship or a series of relationships with an answer of true or false. In the context of our robots, the conditions used are typically based on comparing information from one of the sensors to a known value. For programming in general, relational operators are used to create conditions. The relational operators in MATLAB are:

<table>
<thead>
<tr>
<th>Relational Operator Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>Equal</td>
</tr>
<tr>
<td>~=</td>
<td>Not Equal</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater Than</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less Than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater Than or Equal To</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less Than or Equal To</td>
</tr>
</tbody>
</table>
Below is an example of a working *if/else* statement. The command ‘display’ is used to display text on the screen. You can learn how the ‘display’ command works by using the ‘help’ command.

```plaintext
grade = 0.8;
if (grade >= 0.6)
    display('You passed.‘)
else
    display('You failed.‘)
end
```

What message will this *if/else* statement display on the screen?

**while-loop**

A *while-loop* is a hybrid between a *for-loop* and an *if/else* statement. A *while-loop* is like a *for-loop* in that it performs a given action multiple times. Unlike a *for-loop*, a *while-loop* does not use a loop counter to determine the number of times to perform the action. Instead, a *while-loop* uses the truth of some condition, much like an *if/else* statement, to determine how long the action should be performed. The basic syntax of a *while-loop* is given below:

```plaintext
while(condition)
    some action
end
```

Just like an *if/else* statement, you can read the syntax the same way you would tell someone about an action you might perform. For example, “While it is sunny outside, I will stay at the beach.” While the condition “it is sunny outside” is true, I will perform the action of staying at the beach. The while statement implies that when the condition becomes false, i.e. when it is not sunny outside, then I will leave the beach. Again, just like an *if/else* statement, a *while-loop* uses relational operators to determine if the condition is true. Unlike an *if/else* statement, the condition is not just evaluated once but it is evaluated after each iteration of the loop.

**Exercises**

1. Using an *if/else* statement like the one above, display one of three different messages depending on the value of the variable *grade*. If *grade* is greater than or equal to 0.9, display the message “You aced the course.” If *grade* is between 0.8 and 0.9, display the message “You almost aced the course.” If *grade* is less than or equal to 0.8, display the message “You didn’t ace the course...nice try.”

2. Using a *while-loop*, divide 1000 by 2 until the result becomes less than 1. Count and display the total number of iterations it takes.

3. Make a very simple program that does the following:
For any number between -1 and 1, the program calculates and displays the arccosine and arcsine of the number in both degrees and radians. For example: if the number is 0, the arccosine is either 90° or π/2 radians, and 270° or 3π/2 radians. To make things easier, only need to be given for angles between 0° and 180° (π).

**Note:** It’s very IMPORTANT to comment your code. Please make sure to include a brief explanation of the code used to complete each of the exercises.