I, Praakrit Pradhan, hereby submit this original work as part of the requirements for the degree of Master of Science in Computer Science.

It is entitled:
The Role of Arduino for Increasing Performance and Interest in Programming for First-Year Engineering Students

Student’s name: Praakrit Pradhan

This work and its defense approved by:

Committee chair: Carla Purdy, Ph.D.

Committee member: Gregory Bucks, Ph.D.

Committee member: Karen Davis, Ph.D.
The Role of Arduino for Increasing Performance and Interest in Programming for First-Year Engineering Students

A thesis submitted to the Graduate School of the University of Cincinnati in partial fulfilment of the requirements for the degree of Master of Science in the department of Electrical Engineering and Computer Science of the College of Engineering and Applied Science by

Praakrit Pradhan

B.S. University of Cincinnati

October 2017

Committee Chair: Carla Purdy, Ph.D.
Abstract

Engineers rely on computers for their work. In engineering curricula, programming is typically required. Students tend to question the viability of programming in their prospective field and are reluctant to learning programming for the first time. At the University of Cincinnati (UC), external data acquisition tools, such as National Instruments Data Acquisition (NI myDAQ), have been used to try and help students learn programming, as the theory of programming can be a difficult experience for the students. MATLAB is taught in a required sequence of courses at UC to all first-year engineering students, entitled Engineering Models I and Engineering Models II, and the NI myDAQs are used to in conjunction with MATLAB. One of the major problems using hardware tools like the NI myDAQ is the difficulty in implementing them for teaching purposes. When teaching first-year engineering students, the use of a tool that is easy to implement is very important. Implementation of a tool is the way the tool can be manipulated or programmed to be able to let the students focus on the topic at hand. Arduino is a universally compatible, cheap hardware tool. This study shows its viability as a tool for improving performance in programming for first-year engineering students. Using an Arduino can make learning programming courses more interesting. Arduino as a tool helps with visual stimulation for learning programming. It can be used to make circuits, which can be controlled by the programs made by students. The ability to control simple LEDs gives visual stimulus to the students. Not only is Arduino relatively inexpensive (under thirty dollars), it can be used with all computer operating systems. This study aims to design and implement introductory programming learning modules and evaluate these modules taught to engineering students. This study designs learning modules which enhance the learning experience. The module covers programming of a simple circuit and making a working thermostat. This study uses the Arduino Uno board as the hardware data acquisition tool. The
effectiveness of the board is assessed by comparing two sections of the Engineering Models I course: in one the teacher only uses lecture materials and a coding-focused lab activity with no hardware tool implementation; in the other the labs were enhanced using the Arduino Uno board. The students in both courses were given surveys and short quizzes to answer. Both the quizzes and surveys were given both before and after the lab to allow for an experimental design with a pre- and post-test. This paper evaluates the role the Arduino system plays in increasing the interest and performance of these students in the programming concepts they were taught. This paper offers a conclusion that by making use of Arduino boards, student performance increases by as much as 5 percent.
# Table of Contents

Abstract ii  
List of Tables vi  
List of Figures viii  
Chapter 1 – Introduction and Background 9  
1.1 Introduction 9  
1.2 Related Research 11  
1.3 Computing Course for First-Year Engineering Students 14  
1.3.1 Engineering Models I and II (ENED 1090/1091) 14  
1.3.2 First-Year Programming Concepts 23  
Chapter 2 – Methods and Tests 26  
2.1 The system 26  
2.1.1 Design of the System 27  
2.1.2 Application of the System 30  
2.2 Results and Evaluations 35  
2.2.1 Results Background 35  
2.2.2 Results 36  
Chapter 3 – Discussion, Conclusions and Future Work 44  
3.1 Discussion 44  
3.2 Conclusions 45  
3.3 Future Work 46  
References 48  
Appendix 50  
A – 1: Downloading and Installing Arduino Package for MATLAB 50  
A – 2: Thermostat Lab 52  
A – 3: LabThermoStat_Solved 62  
A – 4: Survey (Before) 64  
A – 5: Survey (After) 66  
A – 6: Raw Data – Complete Data 68  
A – 7: Raw Data – Separated form 70
**List of Tables**

<table>
<thead>
<tr>
<th>Table #</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overall comparison of some researches related to the use of Arduinos as an educational tool</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Actual Values for bar graph in Figure 18</td>
<td>36</td>
</tr>
<tr>
<td>3</td>
<td>Actual Values for bar graph in Figure 19</td>
<td>37</td>
</tr>
<tr>
<td>4</td>
<td>Group 1 Statistics and Change</td>
<td>38</td>
</tr>
<tr>
<td>5</td>
<td>Group 2 Statistics and Change</td>
<td>38</td>
</tr>
<tr>
<td>6</td>
<td>Summary of data with Tests of Differences</td>
<td>39</td>
</tr>
<tr>
<td>7</td>
<td>Students Remarks on the Group 2 Lab (Arduino Lab)</td>
<td>43</td>
</tr>
</tbody>
</table>
## List of Figures

<table>
<thead>
<tr>
<th>Figure #</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Example of a successfully completed Lab problem</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>Example Problem 3</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>Correct Solution to Example Problem 3 (from Figure 2)</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>Part C. of a Pulse Width Modulation (PWM) Lab with Arduino</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>Images in the PWM lab to show the complete setup for beginning the lab</td>
<td>19</td>
</tr>
<tr>
<td>6</td>
<td>Part E. of the Pulse Width Modulation lab</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>Expected plot resulting the correct solution of the PWM lab</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>Parts D and E correctly completed</td>
<td>21</td>
</tr>
<tr>
<td>9</td>
<td>Excerpt from Appendix A-1: Installing Matlab Support Package for Arduino Hardware</td>
<td>26</td>
</tr>
<tr>
<td>10</td>
<td>Path to BareMinimum.c on an Arduino IDE</td>
<td>27</td>
</tr>
<tr>
<td>11</td>
<td>Excerpt from LabThermoStat.m</td>
<td>28</td>
</tr>
<tr>
<td>12</td>
<td>Excerpt from LabThermoStat Final Step of the Circuit</td>
<td>30</td>
</tr>
<tr>
<td>13</td>
<td>Excerpt from LabThermoStat_Solved: Simple initial commands</td>
<td>30</td>
</tr>
<tr>
<td>14</td>
<td>Excerpt from LabThermoStat: Explanation of the questions</td>
<td>31</td>
</tr>
<tr>
<td>15</td>
<td>Excerpt from LabThermoStat: Part E of Thermostat lab</td>
<td>32</td>
</tr>
<tr>
<td>16</td>
<td>Portion of the solution to the Thermostat Lab</td>
<td>33</td>
</tr>
<tr>
<td>17</td>
<td>One variation of the question asked to the student groups</td>
<td>35</td>
</tr>
<tr>
<td>18</td>
<td>Percentage of correct responses received on survey administered to ENED 1090 students using Arduino lab</td>
<td>36</td>
</tr>
<tr>
<td>19</td>
<td>Percentage of correct responses received on survey administered to ENED 1090 students using non-Arduino lab</td>
<td>37</td>
</tr>
<tr>
<td>20</td>
<td>Comfort level questions asked to both groups, before and after the labs</td>
<td>40</td>
</tr>
<tr>
<td>21</td>
<td>Arduino helps increase comfort level for IF..Else and While Loops</td>
<td>41</td>
</tr>
</tbody>
</table>
Chapter 1 - Introduction and Background

1.1 Introduction

Computer programming has become an essential part of our daily lives. The knowledge of rudimentary coding is a must for all engineers. The course Engineering Models I at the University of Cincinnati (UC) attempts to address this issue by teaching MATLAB as a tool for solving engineering problems. This course is a mandatory course for all students in engineering. It uses MATLAB to teach fundamental programming concepts as well as teach data manipulations and presentation of data. Using MATLAB as a tool for both teaching coding and exposing students to engineering challenges, this course has become a struggle for many students. Students with a background of computer science tend to find this course easy but may miss the engineering aspect, and students with no background in coding have difficulties getting through the course. This is because programming is difficult to learn [1].

In the world today, students in engineering need to be able to code as well as see how to make real life problems easier with the use of computing technology. Coding as a topic for non-coders is always a challenge to teach. Students arguably learn better if they enjoy what they are learning. As a lecturer, it is always a challenge to get students to enjoy a course. In this paper, the effects of using an Arduino Uno board to make programming interesting to students are studied. Using a hardware tool helps level the learning field between students with previous coding experience and those without. This raises the level of interest by working with hardware for those with no coding backgrounds and gives a new challenge to those who do have a coding background. Differences
in interest levels between students who didn't use any hardware to learn coding concepts versus those that used the Arduino Uno as a tool to learn are evaluated.

The Engineering Models I course explores different scenarios where the engineering students need to put themselves in the shoes of different engineers. They are given problems that have been written in scenarios to depict practical real world problems. Some examples include generating code for figuring out projectile motion calculations to make simulations with the help of MATLAB and using weather data of the past year to estimate the weather probability for the next day in a given location. After doing the task, they are required to complete a lab. The lab has specific requirements that teach students how to present data professionally. Doing this helps teach students different kinds of tasks engineers need to be prepared to take on.

About 40 percent of students planning on engineering and science majors end up switching to other subjects or failing to get any degree [4]. This course acts as a way for students to experience different engineering careers, to better plan their degree choices.

MATLAB is a great tool for allowing students to see the different engineering problems and work on solving them at a student level. But software and coding alone don’t replicate scenarios best when it comes to engineering. Hardware and software work together to make complete systems. Engineering student, they need to understand that they need to be able to work with both aspects of the engineering world. To make this possible, we will use the Arduino Uno micro-processor board. Arduino can represent the hardware components first-years need to work with. This way different types of engineering concepts can be related to students. This thesis will evaluate the
performance of students who are given labs with Arduino and those who are only given the labs without any hardware component.

The study described in this paper has two objectives: (1) to design and implement an Arduino Uno-based system to work with MATLAB, and (2) to test the system in a learning module for first-year engineering students and get feedback on (a) the performance of the system and (b) the performance and interest of the students in computing.

1.2 Related Research

There are a variety of research efforts that explore the use of Arduinos and similar micro-processor boards in different courses and educational settings. Boards such as the Arduino have become popular due to the ease with which students can quickly develop simple circuits or prototypes and the low cost. In this study, I have evaluated the Arduino for teaching programming to first-year engineering students. Some of these studies are discussed in this section to give a wider view of the use of Arduino as an educational tool.

A study similar to the one presented in this paper is the research done by Rubio et al. [5]. This research focuses on engineering and science students. They explore how teaching concepts with the help of Arduino boards can be effective. They have two major aims: to design and implement introductory programming modules with the use of an Arduino and to evaluate these modules. However, the focus of their work is different than that presented here in that they make these modules for higher level engineering and science students and they are enhancing the teaching of C/C++ with the use of Arduino instead of MATLAB. This research, however, does support the
goals of this study in that they successfully showed that the use of an Arduino to teach programming to students was accepted and improved students learning within their university.

Related research is summarized in Table 1. This table highlights the major aims and conclusions of related studies. The use of Arduino for mechanical and/or control education has been formally validated and many schools utilize Arduino’s in their curriculum. In this study, however, Arduinos are used to teach programming to first-year engineering students. By going through the aims and conclusions in the table below, we do not see the use of Arduino to increase performance in programming for first-year students.

This study intends to address the lack of research into the use of Arduinos with first-year engineering students. Specifically, this research explores the role Arduino plays in helping first-year engineering students increase performance and interest towards programming.
### Research Major Aims Conclusions Remarks

**Rubio et al. [5]**  
(i) Design and implement several introductory programming modules  
(ii) Evaluate these modules when taught to science students  
Applying use of Arduino in teaching students could increase learning and satisfaction overall.  
This research focuses on the teaching of different subjects with the use of an Arduino.

**Jamieson et al. [6]**  
(i) How there are a range of boards, and how they are being used in education  
(ii) Challenge adopting these boards  
This research exposed educators to the benefits and challenges of using modern prototyping micro boards.  
This study explores the boards and how they can be used, and what the challenges are in using them. It includes some useful tips

**Sobota et al. [7]**  
(i) Use of boards over virtual circuit simulations  
(ii) REX Core and the uses and advantages  
The mentioned boards can be used in control education. These boards have features that allow for the use of virtual simulations as well as allow physical proof of concepts  
This research focuses on control education and how making use of physical boards is more beneficial than just doing simulations of closed loop experiments

**Bjedov et al. [8]**  
(i) Programming teaches students to think logically  
(ii) Programming allows students to solve problems  
(iii) They’ll need to program on the job  
Programming languages and tools like Matlab can be beneficial to freshmen engineering students as traditional programming languages  
This research challenges the traditional programming languages (C/C++, FORTRAN). MATLAB has greater ease of use, it helps students emphasize problem solving and causes fewer errors

**Grover et al. [9]**  
(i) To show how Arduino platform is helpful and can be used  
Effective low cost platform for a mechatronics curriculum with the use of Arduino.  
This research is slightly less connected to my research, as it is focused on the use of Arduino in a specific course, i.e. Mechatronics.

| **Table 1. Overall comparison of some researches related to the use of Arduinos as an educational tool** |  |
|---|---|---|---|

12
1.3 Computing Course for First-Year Engineering Students

1.3.1 Engineering Models I and II (ENED 1090/1091)

At the University of Cincinnati, Engineering Models I (ENED 1090) and Engineering Models II (ENED 1091) is a required for all first-year engineering students. These courses meet twice a week. One meeting is a lecture session and the other is a lab/recitation session. The students that are in these classes are primarily working on learning to use MATLAB as a tool for solving engineering problems. While teaching MATLAB, these courses also work on making the students more accustomed to engineering practices. They are given work that makes them use knowledge from their other first-year courses, such as calculus and physics. While trying to get the students to work through engineering problems, these courses also helps them learn to work with data and display data in a professional manner. These courses also have Homework assignments that follow up on what is done during the lecture and lab to further help the students understand the concepts covered in MATLAB.

ENED 1090 taught in the fall semester and ENED 1091 is typically taught in the spring semester. The first course helps the students understand the basics of MATLAB and programming in general. The second course takes them further into applications and problem solving, data collection/presentation, and group work. The courses teach programming, data manipulation and teamwork to students entering the engineering programs at the University of Cincinnati.

The lectures during these courses teach the fundamental concepts of programming. MATLAB syntax is presented so the students to be able to work on their labs. The lecture includes scenarios
such as calculating sum of numbers or cumulative interest, where the students would need to make use of the programming concepts taught. This helps students broaden their views of where programming can be applied.

The courses are set up so that the lectures happen before the labs. The students are given lecture supplemental videos to watch before the lecture. These videos show them the different styles/techniques that they can use to apply the syntax or code they learnt during the lecture. When they come to the lecture, they are expected to do a quick quiz on the lecture topic taught in the video.

The lab consists initially of simple questions or results for running certain lines of code. After the initial set of questions, the lab will go on further to make the students work on larger problems that require them to make use of their previous knowledge of MATLAB (from previous lectures) and what they learned the given week.

1.3.1.1 Labs without the use of external hardware devices

The labs in ENED 1090 run through simple initial questions. These questions are made to give a quick review of the use of the coding concept or syntax taught that week.
Problem 1: Consider the code show below. Complete the table showing the values for the variables as the loop progresses. Do this without MATLAB first then check your results using MATLAB.

```matlab
x = 3;
y = 5;
for k = 1:4
    Sum = x + y;
    x = y;
    y = Sum;
end
```

<table>
<thead>
<tr>
<th>Sum</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before Loop</td>
<td>Not Defined Before Loop</td>
<td>3</td>
</tr>
<tr>
<td>k = 1</td>
<td>3 + 5 = 8</td>
<td>5</td>
</tr>
<tr>
<td>k = 2</td>
<td>5 + 8 = 13</td>
<td>8</td>
</tr>
<tr>
<td>k = 3</td>
<td>8 + 13 = 21</td>
<td>13</td>
</tr>
<tr>
<td>k = 4</td>
<td>13 + 21 = 34</td>
<td>21</td>
</tr>
</tbody>
</table>

Figure 1. Example of a successfully completed Lab problem

Figure 1 shows an example problem from the lab. This is one example of a simple initial question. It requires students to do the problem without the use of MATLAB first, and later use MATLAB to correct their own work. This problem helps students understand “For loops”.

As the students go through the lab, the problems get more difficult and require more work. Students are required to use more than just the appropriate syntax to get the result. They are asked questions that expect them to further analyze their results and to make smart use of the concepts they have been working on.
Problem 3: For this problem (and the next one), you will need to download HighTemp.mat from the metaite (posted with Homework #5) and save it in your current MATLAB folder. This file contains the daily high temperature measured at the CVG airport from January 1, 1970 through December 31, 2014 which amounts to 45 years of data or 16,436 daily temperatures. One really nice application of for loops is to count stuff in large sets of data which is what you will do in this problem and in next problem.

Write a script file that uses a for loop to count how many days in this set of data, the high temperature was greater than or equal to 80°F. You will need to do the following in your script:

- clear; close all; clc;
- load HighTemp;
  *This command will import the 1-d array to your Workspace. Put this line in your script right below the clear; close all; clc line. This command will only work if HighTemp.mat really is in your current MATLAB folder.*
- Determine how many data points are in HighTemp and save it as variable (the length function should prove useful here!)
- Create a variable called Count and set it initially to 0.
- Create a for loop that counts how many days the high temperature was 80 or above. Your for loop should do exactly what you would do if you had to do this problem by hand (which would be painful!). You would likely first look the first temperature in the list, HighTemp(1). If it is at least 80, you would increase Count by 1. Then you would look at the 2nd temperature, HighTemp(2). Again, if it is at least 80, Count would increase by 1. This pattern would continue until you finally arrived at the last value and completed the counting process. **Note: you must implement a for loop to get full credit for this problem – there are ways to do this without a loop that we will discuss later but we are practicing for loops here.**
- After the loops ends, include an fprintf statement that outputs the number of days that the high temperature was 80°F or above (integer value) and another fprintf statement that outputs the percentage of days in the dataset that the high temperature was at least 80°F (using one place behind the decimal point).

Figure 2. Example Problem 3

Figure 2 is an example problem that requires students to work with more data. To complete the problem successfully, students need to understand conditional statements and how to use them within loops. On top of the different syntax, they are also required to understand how arrays work within MATLAB. Figure 3 below shows the correct solution to the problem. The students have had to understand the use of loops and how to setup conditionals inside of loops. This is more than just using the syntax and applying to the problem. This shows that the have had to do more than just apply the syntax to get a solution to the problem.
% Number of High Temps >= 80 at CVG from Jan 1,1970 thru Dec 31,2014
clear; close all; clc
load HighTemp;

N = length(HighTemp);
Count = 0;
for k = 1:N
    if HighTemp(k) >= 80
        Count = Count + 1;
    end
end

fprintf('The number of days with High Temp >= 80 is: %i \n',Count);
fprintf('The percentage of days with High Temp >= 80 is: %0.1f \n',Count/N*100);

Figure 3. Correct Solution to Example Problem 3 (from Figure 2)

The above figures are only one example of labs that students perform every week. They work on these labs together in class with the assistance of teaching assistants and the professor.

1.3.1.2 Labs with Arduino Uno

At the University of Cincinnati, a hardware tool was used to assist the students in learning programming. National Instruments myDAQ was used as a tool. Arduino has advantages over the myDAQ. The major problem with the myDAQ is that the students do not have easy access to the tool. This tool was exclusive and expensive, buying it as a student cost $450.00 dollars [10]. This tool also was incompatible to use in a MacBook which caused some groups to have more than 4 members. Even though the function of myDAQ is like that of an Arduino for the purposes of ENED 1090, the fact that an Arduino Uno is cheaper, and works with all operating systems makes it a better choice. There is an added benefit to using an Arduino, it is open source. This makes it easy to use on all computers, and allows students to look at other projects done by enthusiasts on YouTube. The Arduino is an affordable tool. The Arduino Uno board only costs $25.00 [2]. During this research, the students are not given myDAQ, they are only given access to an Arduino Uno.
Labs with Arduino Uno are used for enhancing the students’ experience to better understand the programming concepts. The use of an Arduino helps students design systems during lab. Systems like a Refrigerator Alarm, Pulse Width Modulation to control LED brightness levels, and Thermostats are example systems that students can run during a lab with the help of an Arduino. Learning using systems helps students not only use the syntax but also to visualize the problems. Only those labs utilizing the Arduino were evaluated for this thesis. The labs were made like previous labs to keep the structure that the ENED 1090 course implemented.

Figure 4. Part C. of a Pulse Width Modulation (PWM) Lab with Arduino
Figure 4 shows a lab with Arduino. This is the Pulse Width Modulation (PWM) Lab. In this lab, students are exposed to frequency of a wave, the period, and duty cycles. This lab, with the help of Arduino and LEDs, shows the students wave patterns. The above figure shows the beginnings of the lab after the setup process is complete. Below in Figure 5, is the initial complete setup for the PWM lab.

![Image 1](image1.jpg)
![Image 2](image2.jpg)

**Figure 5. Images in the PWM lab to show the complete setup for beginning the lab**

The students run through the beginning of the lab, which includes setup instructions to be able to get to the point shown in Figure 5. After reaching this point, students run through simple instructions to confirm their setup is correct and get a brief explanation of some of the commands necessary to operate the Arduino, as shown in Figure 4.
E. Measuring the Voltage Across the Resistor and Sending Data Back to MATLAB

So far, you've only sent voltage out to the circuit using ~6" and GND (red and black wires). Now, we will use ~A0~ to measure the voltage across the LED and send the data back to MATLAB (yellow and black wires).

1. Go to Line 84 in the LED_PartD_E.m-file. Read the comments and fill in the required code to plot the measured and applied voltages.

2. Run the file using the following inputs then answer the following questions using the plot:
   - Frequency, \( f \): 0.5
   - Duty Cycle, \( t_{\text{d}} \): 50
   - How Long? 5

Using the data cursor tool, what is the voltage across the resistor when the Applied Voltage is 5 V?

Using Ohm's Law (\( V = IR \)) and the resistor value of 1 kOhm, what is the current flow through the circuit when the Applied Voltage is 10 V?

What is the voltage across the resistor when the Applied Voltage is 0 V?

What is the current flow through the circuit when the Applied Voltage is 0 V?

3. Paste your plot and your script file commands in the spaces indicated below

PASTE PLOT HERE:

PASTE SCRIPT HERE:

Figure 6. Part E. of the Pulse Width Modulation lab

Figure 7. Expected plot resulting the correct solution of the Pulse Width Modulation lab
In making this lab, it was structured like the previous labs done in the ENED 1090 course. After completing the initial setup and simple questions of the lab, the students then move on to the harder problem of the lab. Figure 6 shows the main part of the lab. This is the section where students need to apply what they know from their previous lectures and labs to solve the given problem.

```plaintext
%Student Code Starts Here For Part D
%Complete the line of code below to ask the user to enter in the desired frequency of the square wave (input statement)
f = input('Please enter the frequency of the square wave (Hz): '); %Complete the line of code below to calculate the period, T, based on f:
T = 1/f;
%Write an fprintf statement to display the computed period to 2 decimal places
fprintf('The period of the square wave is: %0.2f sec\n',T);
%Complete the line of code below to prompt the user for a duty cycle
tau = input('Please enter the duty cycle (%): '); %Complete the line of code below to prompt the user for how long the square wave should last
Tf = input('Please enter the duration of the square wave signal (sec): '); %Complete the line of code below to create a vector of times, t.
t should start at 0, increment by 1/arduino_rate, and end close to or at Tf
T = 0:1/arduino_rate:Tf;
%Complete the line of code below to create a square wave called AppliedVoltage %using your vector, t, the desired frequency, f, and desired duty cycle, tau. Square wave should vary from 0 to 1 (0V to 5V).
AppliedVoltage = 2.5*square(2*pi*f*t,tau)+2.5;
```

Figure 8. Parts D and E correctly completed

Figure 8 shows code written by a student to successfully complete the lab. The students are given script files in this lab with instructional comments that guide them to be able to complete the lab (Green font in Figure 8 indicates instructional comments in the code for students to follow).
1.3.2 First-Year Programming Concepts

This paper studies learning modules used to cover introductory programming concepts for first-year engineering students. Engineering Models at the University of Cincinnati is a course where simple coding concepts are tied to real world engineering systems.

The tools that are used to help at the university are MATLAB and Arduino. MATLAB is a high level dynamic programming language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include data analysis, exploration, and visualization. Arduino is an open source computer hardware and software company. It has a user community that designs and manufactures single-board micro-controllers and micro-controller kits for building digital devices and interactive objects that can sense and control objects in the physical world [3].

This thesis compares these two tools for first-year engineering at the University of Cincinnati. MATLAB is required for the students in first-year engineering to purchase in the university. It is used as a basic programming language. It is also a tool that makes programming concepts easier to teach and data easy to manipulate. The open source Arduino allows flexibility and easy use. With the use of Arduino, it is easy to be able to do simple hardware labs at a lower cost as well as to get the students familiar with simple micro controllers, Arduinos are arguable among the popular DIY (do it yourself) kit available [11].
To increase interest for engineering students, requires more than just programming simple scripts. Using Arduino with MATLAB allows the students to understand coding can be converted to physical systems. Arduino allows for physical circuits and systems and MATLAB acts as the gateway to controlling the system.

In this project an Arduino Uno is being used for the understanding of “If..Else” and “While” loops. Core programming concepts such as loops and conditionals are important for students in engineering to learn. The aim is to use the traditional teaching methodology, and further enhance it with the use of an Arduino. The Arduino provides several advantages when used as a teaching tool. For the university, making use of open source hardware comes at a low cost [12]. For students, being able to use something they can easily acquire for themselves if they want to practice for themselves or design their own systems makes Arduino a very comfortable tool.

A conditional statement, symbolized by $p \rightarrow q$, is an if-then statement in which $p$ is a hypothesis and $q$ is a conclusion. The logical connector in a conditional statement is denoted by the symbol $\rightarrow$. The conditional is defined to be true unless a true hypothesis leads to a false conclusion. All coding environments, from Excel, to VB script, to C/C++, to assembly language have forms of conditional structures. Being able to understand conditional statements and how to use them is often the first step in any programming language after the basic understanding of variables and calculations.

A loop repeats the same instructions or processes the same information over and over until receiving the order to stop [14]. A loop is fundamental in all programming languages as well.
Having to do things multiple times, under a certain condition is a typical example of useful programming. This is another initial step in learning programming languages.

The first-year programming for the scope of this research is limited to these two-major programming constructs. These two constructs are core for students to be able to understand the use of programming and hardware in real world applications. In this research, the students will use these constructs to simulate a thermostat. Students’ ability to understand how while loops and if-else structures are applied towards real-world systems will be evaluated. In the process of doing the lab, students are also exposed to a microprocessor’s ability to emulate different real world systems.
Chapter 2 - Method and Tests

An Arduino Uno typically requires its own integrated development environment (IDE) for compiling. To make it easier on the students for this research and lab, they do not have to learn another language or environment to work with Arduino. Labs are setup with certain instructions embedded into files. This allowed the students to access the Arduino through the USB interface and MATLAB. This chapter explores what was required for the system to be setup. It also discusses the labs that the students ran, the surveys and questions they filled out and the evaluations and results.

2.1 The System

The Arduino IDE requires compiling code written in C/C++. First-year engineering students at UC are not taught to use this language and have not worked with systems that require compiling and pre-allocating of data. This required a bridge to make use of Arduino in first-year engineering. The bridge is the connection between MATLAB and the Arduino Uno. There are two tasks: the design of the system and the application of the system.

The system has constraints due to its requirement of being easy to use for students in first-year engineering courses. Testing or application of the system is also required. This is when the system was applied in a classroom setting. Here, difficulties arose in explaining and helping with the initial setup of the system on all the students’ computers.
2.1.1 Design of the System

The use of Arduino Uno for this research had some initial problems. The first step to working with an Arduino Uno on MATLAB included downloading and installing the Arduino support package add-on from Mathworks into MATLAB. This package allowed the computer to talk to the Arduino through MATLAB. An excerpt from the required support page documentation is shown in Figure 9.

![Add-On Manager](image)

**Figure 9. Excerpt from Appendix A-1: Installing Matlab Support Package for Arduino Hardware**

This setup process was required to be done on all computers that were going to use the system. The first step of designing the system consisted of creating the download installation guide for the students and having them install it at least a day before the lab. This allowed the students’ computers to be able to communicate with the Arduino.

The second part to setup the system required the Arduino to be cleared of all previous codes running on the board. The students were all given brand new Arduino Uno’s which allowed this process to be skipped. However, to work on the system one would have to plug in the Arduino Uno, open the Arduino IDE and compile the “BareMinimum” into their Arduino. Figure 10 shows how to get to the BareMinimum file.
If unable to find the BareMinimum file, it is also possible to simply compile the following lines of code into the Arduino Uno board:

```
void setup() {}
void loop() {}
```

This clears the Arduino of all potential code still running inside.

The next step was to set up the Arduino to be prepped for the lab given to the students. The solution to applying code to all Arduino Uno’s was to provide the students with files they downloaded to work in the labs. As shown in Figure 11, students were given the setup script as a section in the file they could download. They then worked on their lab with the setup file already incorporated.
Figure 11. Excerpt from LabThermoStat.m

Arduino has several built-in functions: Setup, Read/Write, I2C, SPI, Shift Registers, Quadrature Encoders, Servo Motors and more [15]. In this project only the Setup functions and the Read/Write functions were used. These were limited functions that allowed for configuring pins to either read voltage values or apply voltage values.

The configurations of the pins shown in Figure 10 are specific to the thermostat lab. This was a small restriction that was in place as the students were not expected to write the commands on their own that set up communication between MATLAB and the Arduino.

The system includes the initial setup of the Arduino Hardware add on, the clearing of the physical Arduino Uno board and having the File LabThermoStat.m (Appendix A2.).
2.1.2 Application of the System

After setting up the system, it was ready to be implemented as a lab. There were labs created to test out the system. The first lab was Pulse Width Modulation (PWM) lab with LEDs. PWM was a test lab to confirm the system was working. Before testing the system, the students were told to download and install the Arduino package in MATLAB (Appendix A1.). They were then told to make groups of two with at least one computer between them that had the package installed.

The PWM lab requires the students to complete a provided skeleton script which configures the Arduino and performs the PWM. They then run through simple exercises as shown in Figure 6. The completion of the PWM lab showed that Arduino could talk to MATLAB and the students were able to write code and complete the lab.

The second lab was for the focus of investigation for the hypothesis of this work. For this lab, the students were asked to sit with the same group as the PWM lab and work together. The previous lab’s completion meant that no extra time was spent for setup of the system. The second lab was where the students were learning about if-else conditions and working with while loops. The lab was set up for the mockup of a thermostat with the help of the Arduino Uno.

The lab is similar in structure to the PWM lab in that the students initially set up a circuit to work with utilizing a step-by-step process. The final step is shown in Figure 12.
Students are then expected to run through simple commands and record the results. This is just to get the students to be able to understand the use of the Arduino Package commands to be able to control the LEDs connected, as shown in Figure 13.

4. Run the following commands and make sure your LED’s light up:

```matlab
>> configurePin(a,'D6','DigitalOutput');
>> configurePin(a,'D5','DigitalOutput');
>> writePWMVoltage(a,'D5',5)
>> writePWMVoltage(a,'D6',5)
```

2. Execute the following line of code at the command prompt:

```matlab
>> writePWMVoltage(a,'D5',5);
```

Record what happens to the LEDs – check your circuit:

LED turns on:

3. Execute the following line of code at the command prompt:

```matlab
>> writePWMVoltage(a,'D6',5);
```

Record what happens to the LEDs:

Second LED also turns on
Figure 14. Excerpt from LabThermoStat: Explanation of the questions

Figure 14 shows an excerpt from the lab that explains how the commands used in the first part of the lab operate. This serves as a setup process to allow the students to work towards making the thermostat system.

In the lab the students set up a bias with a simple thermocouple that gives a voltage value based on the temperature of the room. They input that reading via Arduino to MATLAB. Checking that voltage and converting that into a temperature value is a simple calculation:

\[ V_{\text{sensor}} = 0.01 \times (T - T_o) + V_o \]

which they can to find the temperature. They need to use this equation within a loop to be able to repeatedly measure the temperature of the room.

Figure 15 shows Part E of the LabThermoStat (Appendix A-5.), which is used to operate the circuit and make the thermostat. Students are given a script file which they download. This file includes the configuration setup of the Arduino.
E. Creating the Thermostat System

Download the script Thermostat_Template.m from Blackboard and complete the required sections using the information from parts C and D and the descriptions below.

1. In the section marked PART E.1, write commands that will ask the user (1) for the desired temperature to maintain and (2) for the acceptable range around the desired temperature.

2. In the section marked PART E.2, you will write the majority of the commands to implement your thermostat system. The code will need to perform the following operations:
   a. Continue to perform the following steps for the thermostat system until the A5 channel drops below 2, indicating that the thermostat has been turned off.
   b. Measure the voltage on the temperature sensor and convert it to a temperature in Fahrenheit (remember that the temperature in the equation you created before is in Celsius!).
   c. Display the current temperature to the command window (the `c1e` command may be useful here).
   d. Based on your temperature, do the following:
      i. If the temperature rises above your acceptable range, turn on the AC by illuminating the LED on line ~D5 and displaying a message to the user that the AC is on.
      ii. If the temperature falls below your acceptable range, turn on the Furnace by illuminating the LED on line ~D6 and displaying a message to the user that the furnace is on.
      iii. Once the AC or the Furnace is on, you should only turn them off once the temperature gets back to desired temperature. This means that to turn off the AC, the temperature must be at or below the desired temperature. Similarly, for the furnace, the temperature must be at or above the desired temperature.
   e. Include the following command to pause the program for 1 second between voltage measurements:

```
    pause(1);
```

Figure 15. Excerpt from LabThermoStat: Part E of Thermostat lab
```matlab
while status >= 4
    status = readVoltage(a,'A5')
data = readVoltage(a,'A0')
temp_C = (data-2.9912)/0.01 + 22.222;
temp_F = (9/5)*temp_C + 32
clc;
    fprintf('Current Temperature: %0.2f *F\n',temp_F);
    % Turning on AC
    if (temp_F > (desired_temp + range_temp))
        writePWMVoltage(a,'D5',0);
        writePWMVoltage(a,'D6',5);
        AC = 1;
        Furnace = 0;
        fprintf('AC is on.\n');
    % Turning on Heat
    elseif (temp_F < (desired_temp - range_temp))
        writePWMVoltage(a,'D5',5);
        writePWMVoltage(a,'D6',0);
        AC = 0;
        Furnace = 1;
        fprintf('Furnace is on.\n');
    % Turning off both AC and furnace
    else
        writePWMVoltage(a,'D5',0);
        writePWMVoltage(a,'D6',0);
        AC = 0;
        Furnace = 0;
        disp('here')
    end
pause(1);
end
```

**Figure 16. Portion of the solution to the Thermostat Lab**

This lab requires the students to use the topic they have been working on for the week during lecture. The students were learning how to use conditional statements and while loops. Figure 16 shows the use of conditionals and while loops required for the solution of the lab. The thermostat lab not only requires them to know these two programming concepts but also gives a practical demonstration of how those programming concepts can be used.
2.2 Results and Evaluations

The research was done with two groups. The first group was the group without an Arduino in their labs. The second group ran the system as explained above, each group had 70 students. They were both given the same lecture and lecture materials. The two groups were only differentiated during the lab portion.

2.2.1 Results Background

The first group was given a lab like that explained in section 1.3.1.1 of this document. These labs were simple, without any hardware component. They were required to know the same two main concepts: if-else and while loops.

The first group of students used a scenario to monitor coffee temperature. This lab required students to get an initial required temperature for the coffee. They were given equations for calculation of temperature over time. They could increase the temperature, also given to them as an equation. They then needed to keep the temperate at a desired value by applying these equations based on conditions within a loop.

Similarly, the students in the second group were expected to keep track of the temperature of the room and keep the temperature at the desired level. They could turn on the furnace (turning on a Red LED), or turn on the AC (turning on a Green LED). Their system needed to know when to turn on the furnace and when to turn on the AC depending on whether the thermocouple was being heated or cooled. The second group were given the Arduino Uno to create and use a thermostat.
To evaluate student understanding of the concepts, the students from both groups were asked to solve a set of questions shown in Figure 17. The survey (to gauge student interest and comfort with the Arduino) and questions were given both before beginning the lab and after the end of the lab. Both groups were required to complete these questions. This setup allowed for an experimental design with a pre- and post-test.

### 2.2.2 Results

The bar graphs in the following pages will show two colors. The blue color representing the results of the data collected from the “before” survey and the green the data results collected from the “after” survey.
Students were asked questions like the one shown in Figure 16 regarding results of certain code snippets. The bar graph shown in Figure 18 shows the percentage of correct responses by the students who worked on the lab with Arduino. The bar on the left shows the responses for the if-else and the one on the right shows the responses to the while loop questions. The bar on the left shows over 80% students got the if-else questions correct, both before and after.

<table>
<thead>
<tr>
<th></th>
<th>IF..ELSE</th>
<th>WHILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before</td>
<td>0.825</td>
<td>0.3333</td>
</tr>
<tr>
<td>After</td>
<td>0.8571</td>
<td>0.5122</td>
</tr>
<tr>
<td>Change</td>
<td>0.0321</td>
<td>0.1789</td>
</tr>
</tbody>
</table>

Table 2. Actual Values for bar graph in Figure 18

Table 2 shows actual values depicted in the bar graph in Figure 18. It shows that the students got 82.5% correct for questions with if-else and got only 33.33% correct for questions with while loop before the lab. However, the results show that more students got questions regarding both topics correct after the lab.
correct after the lab. We can see that about 20% more students were able to understand the concept of while loops after doing the Arduino lab.

![Bar graph showing percentage of correct responses received on survey administered to ENED 1090 students using non-Arduino lab.]

**Figure 19. Percentage of correct responses received on survey administered to ENED 1090 students using non-Arduino Lab**

Figure 19 shows a bar graph similarly structured to the one shown in Figure 18. It shows the students in the group who did not work with Arduino.

<table>
<thead>
<tr>
<th></th>
<th>IF..ELSE</th>
<th>WHILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before</td>
<td>0.5965</td>
<td>0.3214</td>
</tr>
<tr>
<td>After</td>
<td>0.5714</td>
<td>0.4286</td>
</tr>
<tr>
<td>Change</td>
<td>-0.0251</td>
<td>0.1072</td>
</tr>
</tbody>
</table>

**Table 3. Actual values for bar graph in Figure 19**

Table 3 shows the values shown in the Figure 19 bar graph. This shows us the change in correctness before and after for this lab was a negative for if-else and only 10% increase for the while loop.

Students were also asked questions regarding their ability to work with both types of syntax and concepts. They were asked how comfortable they were in using the codes in scenarios where it...
was necessary. This question was asked to both groups, both before and after the lab. Table 4 and 5 show the resultant values from the survey.

<table>
<thead>
<tr>
<th>Group 1</th>
<th>Before (N=51)</th>
<th>After (N=35)</th>
<th>Both (N=34)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF</td>
<td>Mean</td>
<td>Mean</td>
<td>Change</td>
</tr>
<tr>
<td>Comfort</td>
<td>6.9</td>
<td>7.3</td>
<td>+0.5</td>
</tr>
<tr>
<td>Correct</td>
<td>84.3%</td>
<td>71.4%</td>
<td>-1 0 1 20.6% 70.6% 8.8%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>While</th>
<th>Mean</th>
<th>Mean</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comfort</td>
<td>5.6</td>
<td>6.6</td>
<td>+0.8</td>
</tr>
<tr>
<td>Correct</td>
<td>37.3%</td>
<td>42.9%</td>
<td>-1 0 1 20.6% 64.7% 14.7%</td>
</tr>
</tbody>
</table>

Table 4. Group 1 Statistics and Change

<table>
<thead>
<tr>
<th>Group 2</th>
<th>Before (N=39)</th>
<th>After (N=41)</th>
<th>Both (N=36)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF</td>
<td>Mean</td>
<td>Mean</td>
<td>Change</td>
</tr>
<tr>
<td>Comfort</td>
<td>7.7</td>
<td>8.1</td>
<td>+0.3</td>
</tr>
<tr>
<td>Correct</td>
<td>89.7%</td>
<td>87.8%</td>
<td>-1 0 1 11.1% 77.8% 11.1%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>While</th>
<th>Mean</th>
<th>Mean</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comfort</td>
<td>6.3</td>
<td>7.2</td>
<td>+0.9</td>
</tr>
<tr>
<td>Correct</td>
<td>35.9%</td>
<td>63.4%</td>
<td>-1 0 1 0.0% 72.2% 27.8%</td>
</tr>
</tbody>
</table>

Table 5. Group 2 Statistics and Change

The change columns in these graphs show the change in comfort and the change in correctness from the before and after survey responses. The comfort level change is the change in the mean

38
comfort level. The change for the correctness is represented in three sections. The first is a ‘-1’ (negative one) which shows the percentage of students who got the answers correct in the first survey and got it incorrectly for the second survey. The second is a ‘0’ (zero) which represents the percentage of students who got the questions both incorrect or both correct. The third section is a ‘1’ which represents the percentage of students who got the answer incorrectly in the first survey and correctly after the lab. In Table 4 and 5 we see that for group 1 (non-Arduino lab) the comfort level for if-else has a positive change of +0.5. This in comparison to group 2, we can see that the mean comfort is only +0.3. We can see that the comfort levels mean change in group 1 was +0.8 and for group 2 was +0.9. The comfort levels don’t indicate much difference between the two groups. Looking at the correctness for the two groups, the main noticeable difference is in the negative change. 20.6% students have been affected negatively by the lab in group 1, whereas none of the students in group 2 have been affected negatively (0.0%).

<table>
<thead>
<tr>
<th></th>
<th>Group 1</th>
<th></th>
<th></th>
<th>Group 2</th>
<th></th>
<th></th>
<th></th>
<th>Test of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Before (N=51)</td>
<td>After (N=35)</td>
<td>Both (N=34)</td>
<td>Before (N=39)</td>
<td>After (N=41)</td>
<td>Both (N=36)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IF Comfort Correct %</td>
<td>Mean</td>
<td>Mean</td>
<td>Change</td>
<td>Mean</td>
<td>Mean</td>
<td>Change</td>
<td>Mean</td>
<td>Mean</td>
</tr>
<tr>
<td>Comfort</td>
<td>6.9</td>
<td>7.3</td>
<td>+0.5</td>
<td>7.7</td>
<td>8.1</td>
<td>+0.3</td>
<td>0.3723</td>
<td></td>
</tr>
<tr>
<td>Correct</td>
<td>84.3%</td>
<td>71.4%</td>
<td>-1</td>
<td>89.7%</td>
<td>87.8%</td>
<td>-1</td>
<td>0.5454</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>70.6</td>
<td>0</td>
<td>77.8</td>
<td>0</td>
<td>1</td>
<td>11.1</td>
<td></td>
</tr>
<tr>
<td>While Correct %</td>
<td>Mean</td>
<td>Mean</td>
<td>Change</td>
<td>Mean</td>
<td>Mean</td>
<td>Change</td>
<td>Mean</td>
<td>Mean</td>
</tr>
<tr>
<td>Comfort</td>
<td>5.6</td>
<td>6.6</td>
<td>+0.8</td>
<td>6.3</td>
<td>7.2</td>
<td>+0.9</td>
<td>0.843</td>
<td></td>
</tr>
<tr>
<td>Correct</td>
<td>37.3%</td>
<td>42.9%</td>
<td>-1</td>
<td>35.9%</td>
<td>63.4%</td>
<td>-1</td>
<td>0.0114</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>64.7</td>
<td>0</td>
<td>72.2</td>
<td>0</td>
<td>1</td>
<td>27.8</td>
<td></td>
</tr>
</tbody>
</table>

Table 6. Summary of data with Tests of Differences

Table 6 shows the change data as well as a column for the P-value between the two groups. The two means for comfort level were used to do a 2-sample t-test comparing the before and after
between group 1 and group 2. The p-values for the comfort levels are not statistically different. The same is for the while loop questions.

For the correctness chi-squared test comparisons were done and all but the while loop questions are not significant. The p-value of 0.0114 (highlighted in Table 6) is small. Here we can see that 14.7% improved in group 1 and 27.8% improved in group 2. In Group 1, 20.6% students did worse while no one did worse in group 2. This is significant, showing that the use of Arduino labs helped students learn the while loop concept.

Figure 20 shows the questions asked for the comfort level to both the groups before and after the labs. Results from these questions are shown in Tables 4, 5 & 6 for comfort levels.

![Comfort level questions](image)

**Figure 20. Comfort level questions asked to both groups, before and after the labs**

*(Questions and survey asked can be found in Appendix A7.)*
A second focus of this study is to gauge the interest of students in engineering and computing. Students were asked if the Arduino helped them increase their interest in engineering and whether it helped them understand the concepts better.

Figure 21 shows percentage of students confirming (values more than or equal to 5) Arduino helped them 77.50% for if-else and 85% for while loops.

The data is the percentage of the dark colored area for the graph in Figure 21. There are two colors here in the graph; the darker color denotes the students who think that the Arduino was of help to
them. The value 5 was for those who felt neutral about the use of Arduino. However, from the graph and the data we can clearly see that the students were more better able to understand the concept with the use of Arduino.

The results in the graphs favor the use of Arduino. To confirm this data is not an anomaly it requires more data collection with the use of Arduino. Overall, the lab was a success and the students learned the concepts they needed to and they appreciated the use of Arduino. In the next chapter the results, problems, and what work there is that can be done to better improve Arduino as a tool required for students in first-year engineering are discussed.
Chapter 3 – Discussion, Conclusions and Future Work

This chapter will discuss the results seen in the previous chapter. It will also include the conclusions that can be drawn from these results and where to go from here. It will also include several problems and potential fixes for the system discussed and the labs created.

3.1 Discussion

The goal of this research is to increase interest and performance in programming for students by using Arduino in classes for first-year engineering students at UC. With just one test case scenario, not much can be said; however, the results of this experiment show that the students overall had an increase in performance when they were learning with Arduino. The p-value for the correctness in solving while loop questions shows that the Arduino labs helped increase students’ performance. It was also seen that students had improved confidence and better ability to make use of the two fundamental programming concepts they were required to learn during the lab. Students were given the option to make comments at the end of the survey. Most students left this section blank.

Below are all the remarks that students from the Arduino lab (Group 2) gave:

<table>
<thead>
<tr>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware helps, but its my four group members to not evenly divide work load and learn.</td>
</tr>
<tr>
<td>Arduino labs would be beneficial if the structure was more simple and instructions were more clear</td>
</tr>
<tr>
<td>Faulty hardware, only one TA knew what the problems were and her to fix issues. Poor instructions</td>
</tr>
<tr>
<td>too many things that could go wrong that we aren't aware of. (not enough power coming from computer)</td>
</tr>
<tr>
<td>Liked the arduinos</td>
</tr>
<tr>
<td>Didn't take first survey</td>
</tr>
<tr>
<td>I enjoy the arduino labs, i think they help visualize whats going on within MATLAB coding.</td>
</tr>
<tr>
<td>I like arduino labs but sometimes the instructions are confusing.</td>
</tr>
</tbody>
</table>

Table 7. Students Remarks on the Group 2 Lab (Arduino Lab)
The setup process for the use of the system with Arduino needs to be made cleaner. This is required to make the lab time more focused on the lab work at hand. The labs required the students to make the circuits. During lecture, showing students how to make use of a breadboard and how the rows and columns are connected in a breadboard can help deduct time from their lab. The use of only two labs with the system might not be able to fully show any positive or negative results of the use of Arduino appropriately. Making more labs that teach a variety of topics with the use of Arduino is a requirement to be able to make clearer conclusions.

3.2 Conclusions

The results show very little increase in performance from the students. The only conclusive result is the confidence value for the while loop questions asked. The results barely show inflections in interest level of the students. With the amount of data and the results of the data it is hard to come to any conclusion. It is not sufficient to say that the students were more interested in engineering due to Arduino. It is however clear that the use of Arduino and the thermostat lab improved students’ performance for while loops.

The students in engineering need to learn to program but also need to be able to work on hardware. This project shows that the use of Arduino probably helps students understand programming concepts. Students who worked on the labs with Arduino better understood programming concepts, these students created a thermostat. They saw their code in action. The research requires more data to make a strong conclusion.
The conclusion shown here helps the department of engineering education at UC to move forward with the use of Arduino Uno in their courses. It is a cost-effective hardware tool that the department can implement. Being able to show statistically significant increase in performance makes a strong case for the use of Arduino in not only ENED1090/1091 but other engineering courses as well.

### 3.3 Future Work

There is much that can be done to make the course and the use of Arduino easier for first-year engineering students. One of the steps to take this further would be to make a widget to be able to have setting up the Arduino more seamless. This might take the form of a small graphical user interface (GUI) for the students to use to set the ports in the Arduino. The GUI would feature a widget that the students can install. This widget would be an Arduino Uno GUI that they could configure and set up. Once the widget runs, the application would set up the Arduino and the students would be able to use the GUI. This would require creation of batch files to talk between Arduino and Matlab. The GUI would feature options that the Arduino requires. Features to set up connections, and to set up the pins on the Arduino would be included.

For the future there also need to be more labs throughout the year teaching the students more programming concepts with more sets of projects. These labs be used for not only the first semester of MATLAB courses but also for the second semester. There can be a multitude of different labs with Arduino. A simple list of things that can be done to further the research would be to:

1. have solo labs with Arduino
2. have homework assignments that require the use of Arduino
3. make more Arduino labs and apply them throughout ENED 1090/1091
The design of the GUI is currently limited to what the MATLAB widget can receive and send from the Arduino. A potential project moving forward could be a mini compiler to set up Arduino Sketches (sketches are code written in C for compiling into an Arduino). This would work as a translator between MATLAB code and C code for the Arduino. This way, the possibilities of what the Arduino can do would be limited to the ability of the students to write MATLAB code.
References


Appendix

A – 1: Downloading and Installing Arduino Package for MATLAB

Step 1: Open Matlab

Step 2: Make sure you are under the “Home” ribbon bar.

Step 3: Click on the down arrow under the “Add-Ons”. Refer image 1.

Step 4: Make sure to click on the “Get Hardware Support Packages”. Refer image 1.

Image 1.

(Note: If you have not logged into your Mathworks account then after this step you will be asked to log into it. Hit login and enter your credentials to login)

Step 5: Then click “Install from internet”, and click “Next>”. Refer image 2.
Image 2.

Step 6: Search for Arduino on the “Support for:” column. Once you find it, click on it and select the “Install” check box. You can choose to click on both, or just the first one. It is only required that you have the first one checked as shown in image 3.

Image 3.

Step 7: After hitting next, it will download and install the packages. Make sure to follow the install wizard. To confirm that you successfully installed these packages, Click on the down arrow on the “Add-ons” and then click on “Manage-Add Ons”. Refer image 1.

Step 8: This will open up a new window “Add-On Manager”. Make sure that the “MATLAB Support Package for Arduino Hardware” is on the list. Refer image 4.

Image 4.
A – 2: Thermostat Lab

Lab: Design of a Digital Thermostat

In this lab, we will again be using the Arduino device and your knowledge of MATLAB to design a digital thermostat. Historically, thermostats worked based on the properties of different materials. A bar with two different metals was used, with each metal expanding at a different rate depending on temperature. As the temperature increased or decreased, the bar would bend in different directions to either engage the furnace for heating or the air conditioner (AC) for cooling.

![Bimetalic Thermostat](image)

**Figure 1: Bimetalic Thermostat**

With the increasing prevalence of small-scale computing systems and the design of electronic sensors for measure a variety of physical attributes, such as temperature, thermostats have moved away from the old bimetal systems to digital systems. In the digital system, the voltage produced by a temperature sensor is measured using a small computing system, which is then converted into a temperature, which is used to determine whether to turn on the furnace or the AC. The user of the thermostat can often program different profiles for different days and different temperatures for different times of the day. Many new digital thermostats are also equipped with WIFI, allowing the user to program the thermostat using his/her smart phone.

![Digital Thermostat](image)

**Figure 2: Example of a Digital Thermostat**

In order to create your thermostat system, you will be required to use your knowledge of conditional structures and loops. You will work in groups of three to design and test your system.

**A. Building the Circuit**

The circuit you must build for this lab is shown below in figure 3. If you are not adept at reading a circuit diagram, follow the picture steps below to create the circuit. To construct it, you will need the following items:
• 2x Light Emitting Diode (LED)
• 2x 1kΩ resistor
• 1x LM335 temperature sensor
• 1x yellow connector wire
• 1x orange connector wire
• 1x red connector wire
• 2x blue connector wire
• 2x black connector wire

Figure 3: Circuit Diagram

Notes:
• Just like the LED that was used in Lab 4, LEDs have a polarity. Connect the longer lead from the LED to the Positive (D−) connections and the shorter lead to the connections with the 1kΩ resistors.
Picture steps to setup circuit:

- Red arrows indicate Longer Leads
- Dark Blue indicates same row
- Insert the LEDs as shown
- Insert one side of the Resistor (R1) to the same row as the LEDs
- Connect the other side of R1 to an open row (to a row where nothing else is connected)

- Insert second resistor (R2) to a row, and another end to the + column at the edge*
- Insert Red wire to temp sensor in the same row as R2.
- Insert Black wire to temp sensor in same row as open row of R1
- Insert Black wire to the same row.
• Insert Red wire from 5V into the + column (This is to power your circuit)
• There is one Red wire from the TempSensor and another that goes into the 5V.

• Insert Blue wire into same row as the red wire to the temp sensor
• Connect the other end to A0
• Now connect the blue wires to the open rows of the LEDs and connect them to ~D6 and ~D5.

• Finally, connect the GND from the Arduino to the – (or the same row as the black wires).

Notes:
• There are variations to this design, following the pictures will make it easier for the TA’s to check as well as for you to follow. Understanding ground and power is the importance of this circuit diagram.
B. Software Check for Arduino

*Note: if you used your computer on Lab 4 and you have not uninstalled any software, you can skip to step 4 to check MATLAB communication with the Arduino*

1. Form teams of 3 students at your recitation tables making sure that at least one team member has the Arduino software installed

2. Ask your T.A. to give you one of the Arduinos

3. **Software Check:** Type the following command in MATLAB:

   ```matlab
   >> a = arduino
   ```

   If you get an output like this then you are good to go:

   ```matlab
   a =

   arduino with properties:
   Port: 'COM3'
   Board: 'Uno'
   AvailablePins: {'D2-D13', 'A0-A5'}
   Libraries: {'I2C', 'SPI', 'Servo'}
   ```

   If you are on a MacBook, part with the Port will look more like:

   ```matlab
   Port: '/dev/tty.usbmodem1421'
   ```

4. Run the following commands and make sure your LED’s light up:

   ```matlab
   >> configurePin(a,'D6','DigitalOutput');
   >> configurePin(a,'D5','DigitalOutput');
   >> writePWMVoltage(a,'D5',5)
   >> writePWMVoltage(a,'D6',5)
   ```

   At this point make sure that both your LEDs turn on. If they do, just run the same “writePWMVoltage” command again but instead of 5 type 0. This is to pass 0 volts to your LEDs.

   ```matlab
   >> writePWMVoltage(a,'D5',0)
   >> writePWMVoltage(a,'D6',0)
   ```

   **If you get the message where the Arduino doesn’t respond and it says something along the lines of “Cannot find PORT”, then don’t fret. Go to your devices manager on your computer. Find the port number and then run the command again, with an update**

   ```matlab
   >> a = arduino('COM5','Uno')
   ```
* assuming that your arduino is plugged into COM5. Make sure to write the correct COM port number when you find where your Arduino is plugged in.

**NOTE:** If you lose connectivity with the Arduino at any point, type `clear all` at the command prompt. This will clear out the old session and allow you to start a new one.

### C. Simple Arduino Input/Output Commands

1. Make sure that you have the circuit wired and connected to the Arduino following the diagrams shown in Part A. Run the following commands by copying them into a script file and running it.

```markdown
clear all;  
a = arduino;  
configurePin(a,'D6','DigitalOutput');  
configurePin(a,'D5','DigitalOutput');  
configurePin(a,'A0','AnalogInput');  
configurePin(a,'A5','AnalogInput');
```

These commands will set up a new communication session with the Arduino. Adds two analog inputs and also sets up two digital outputs. If you set up the circuit appropriately, the digital outputs should lead into the LEDs and one of the analog input should come from the Temp sensor.

2. Execute the following line of code at the command prompt:

```markdown
>> writePWMVoltage(a,'D5',5);
```

Record what happens to the LEDs – check you circuit:

3. Execute the following line of code at the command prompt:

```markdown
>> writePWMVoltage(a,'D6',5);
```

Record what happens to the LEDs:

4. Execute the following line of code at the command prompt:

```markdown
>> writePWMVoltage(a,'D5',5);  
>> writePWMVoltage(a,'D6',5);
```

Record what happens to the LEDs:

5. Execute the following line of code at the command prompt:

```markdown
>> writePWMVoltage(a,'D5',0);
```
>> writePWMVoltage(a,'D6',0);

Record what happens to the LEDs:

6. Execute the following line of code at the command prompt:

   >> Status = readVoltage(a,'A5')

   Record the value of Status

7. Move your orange A5 wire to the row with your GND cable. Execute the following line of code at the command prompt:

8. Move your A5 wire back to where it was with the red 5V wire

   >> Status = readVoltage(a,'A5')

   Record the value of Status

9. Check with your T.A. to see if all your answers for this part are OK!

**Explanation:**
The `readVoltage` and `writePWMVoltage` commands allow you to read data from your input lines or send out to your output lines, respectively. However, as you saw, the voltage will remain at the level set until you change the value again.

**D. Setting up your Temperature Sensor**
As was mentioned in the background information, the temperature sensor will output a voltage depending on the temperature it is experiencing. Follow the steps below to correctly convert your voltage reading into a temperature in Fahrenheit.

1. Make sure that your temperature sensor is at room temperature (i.e. you haven’t been holding it or giving it an icy glare), then start by running an input scan to get the current voltage reading of your sensor. Find the thermostat in the room and record the temperature. The temperature at your specific location may be a bit different, but we’ll assume the room is the same temperature throughout. (Hint: Use the `readVoltage` value for the ‘A0’)

   **Temperature Sensor Voltage:** ________________________________
   **Room Temperature (in °F):** _________________________________

2. The voltage of the temperature sensor changes by 10mV per degree Kelvin. Since Celsius and Kelvin change at the same rate, we can use Celsius instead of Kelvin to set up our system. Convert the room temperature you read off of the thermostat in the room to Celsius.

   **Room Temperature (in °C):** _______________________________
3. The voltage output of the temperature sensor can be expressed using the following equation:

\[ V_{\text{sensor}} = 0.01 \times (T - T_o) + V_o \]

where \( T \) is the current temperature (in °C), \( T_o \) is the reference temperature (measured in step 1, in °C), \( V_o \) is the reference voltage for \( T_o \) (in V), and \( V_{\text{sensor}} \) is expected voltage measurement for the current temperature (in V).

While knowing what the voltage will be based on the current temperature is nice, we actually need to go in the opposite direction: we will measure the voltage and we need to turn that value into a temperature. Solve the \( V_{\text{sensor}} \) equation for \( T \) and plug in your values for the reference voltage and temperature:

**Expression for current temperature based on voltage measurement:**

\[ T = \frac{V_{\text{sensor}} - V_o}{0.01} + T_o \]

**E. Creating the Thermostat System**

Download the script Thermostat_Template.m from Blackboard and complete the required sections using the information from parts C and D and the descriptions below.

1. In the section marked PART E.1, write commands that will ask the user (1) for the desired temperature to maintain and (2) for the acceptable range around the desired temperature.
2. In the section marked PART E.2, you will write the majority of the commands to implement your thermostat system. The code will need to perform the following operations:
   a. Continue to perform the following steps for the thermostat system until the A5 channel drops below 2, indicating that the thermostat has been turned off.
   b. Measure the voltage on the temperature sensor and convert it to a temperature in Fahrenheit (remember that the temperature in the equation you created before is in Celsius!).
   c. Display the current temperature to the command window (the `clc` command may be useful here).
   d. Based on your temperature, do the following:
      i. If the temperature rises above your acceptable range, turn on the AC by illuminating the LED on line ~D5 and displaying a message to the user that the AC is on.
      ii. If the temperature falls below your acceptable range, turn on the Furnace by illuminating the LED on line ~D6 and displaying a message to the user that the furnace is on.
      iii. Once the AC or the Furnace is on, you should only turn them off once the temperature gets back to desired temperature. This means that to turn off the AC, the temperature must be at or below the desired temperature. Similarly, for the furnace, the temperature must be at or above the desired temperature.
e. Include the following command to pause the program for 1 second between voltage measurements:

```
    pause(1);
```

3. In the section marked PART E.3, write a command that will send out a 0V value to both LEDs and will also display a message to the user that the thermostat is no longer running.

4. To test your script, you will need a way to both heat up and cool down the temperature sensor. To heat up the sensor, you can use your fingers or hand (unless you’re a White Walker) and to cool it down, you will need to fill up a cup of water from the local drinking fountain in which you can stick the sensor.

**Notes:**
- Use the `writePWMVoltage` and `readVoltage` commands to read your voltage measurements and to set the voltage level of the LEDs.

   **When you have your thermostat system working, demonstrate it to your TA.**

**Paste your script below:**

**To be turned in:**

- **Everyone** should submit the m-file your group created during this lab activity, Thermostat_Template.m, and a copy of this lab report with the all calculations, measurements, and observations entered. The team member who has this information should e-mail these documents to the other members of his/her team.
- Please place the names of your team members in the header of your script file, at the beginning of your Word document, and in the submission box.
%% Lab: Test for connection of Arduino

% house cleaning
clear; % clears out everything in workspace
close all; % closes all figures
clc; % clears out the command window

% set up connection with Arduino
fprintf('Setting up connection to Arduino...
');
a = arduino;
configurePin(a, 'D6', 'DigitalOutput');
configurePin(a, 'D5', 'DigitalOutput');
configurePin(a, 'A0', 'AnalogInput');
configurePin(a, 'A5', 'AnalogInput');
fprintf('Testing connection speed...
');
pause(1);

% running sequence to test computer communication speed with Arduino
for index = 1:10
    tic
    writePWMVoltage(a, 'D6', 0)
    p = toc;
    readVoltage(a, 'A0');
    q = toc;
    test(index) = p + q;
end

% setting up the ideal arduino_rate for computer
arduino_rate = 1/(mean(test) + 0.02); % adding an error margin

fprintf('Speed test all done...

Arduino %s connected at port:%s
', a.Board, a.Port);
fprintf('You may begin coding! 

');

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % SECTION E.1
desired_temp = input('What is your desired temperature? ');
range_temp = input('What is your acceptable range of temperatures? ');
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% % SECTION E.2
status = readVoltage(a, 'A5');
AC = 0;
Furnace = 0;
while status > 4
    status = readVoltage(a, 'A5')
    data = readVoltage(a, 'A0')
    temp_C = ((data-2.9912)/0.01) + 22.222;
    temp_F = (9/5)*temp_C + 32;
    clc;
    fprintf('Current Temperature: %0.2f *F
', temp_F);
    % Turning on AC
    if temp_F > (desired_temp + range_temp)
        writePWMVoltage(a, 'D5', 0);
        writePWMVoltage(a, 'D6', 5);
        AC = 1;
        Furnace = 0;
        fprintf('AC is on.
');
    % Turning on Heat
    elseif temp_F < (desired_temp - range_temp)
        writePWMVoltage(a, 'D5', 5);
        writePWMVoltage(a, 'D6', 0);
        AC = 0;
        Furnace = 1;
        fprintf('Furnace is on.
');
    % Turning off both AC and furnace
    else
        writePWMVoltage(a, 'D5', 0);
        writePWMVoltage(a, 'D6', 0);
        AC = 0;
        Furnace = 0;
        disp('here')
    end
    pause(1);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% SECTION E.3
writePWMVoltage(a,'D5',0);
writePWMVoltage(a,'D6',0);
fprintf('Thermostat is off.\n');
A – 4: Survey (Before)

3/1/2017

Models 1: Survey (Before)

1. How comfortable are you writing code that involves using "If Else"?
   Mark only one oval.

   |   |   |   |   |   |   |   |   |   |   |
   |   |   |   |   |   |   |   |   |   |   |
   | Uncomfortable | | | | | | | | | Very Comfortable |

2. In your own words describe an "if...elseif...else" structure

   
   
   
   

Answer the next question based on this code snippet.

```c
1 2 3 4 5 6 7 8 9 10
1 a = 25;
2 if (a == 10)
3   b = 10;
4 elseif (a > 5 || a < 33)
5   b = 35;
6 else
7   b = 0;
8 end
9 fprintf('b = %i',b)
```

3. What would the output for the if...else code above be?

   
   
   
   

4. How comfortable are you writing code that involves a "While Loop"?
   Mark only one oval.

   |   |   |   |   |   |   |   |   |   |   |
   |   |   |   |   |   |   |   |   |   |   |
   | Uncomfortable | | | | | | | | | Very Comfortable |

https://docs.google.com/forms/d/1H2yX5TjZxtP3t2DdpB8ElxqgE1HyB5G+j-5FjkNMKwbo8M/edit

1/3
5. In your own words describe a While Loop

Answer the next question based on this code snippet.

```plaintext
1  c = 0;
2  a = 15;
3  while a > 10
4     c = c+1;
5     a = a-1;
6  end
7  fprintf('C = %i', c)
```

6. What would the output for the “while loop” code above be?

7. Gender
   - Mark only one oval.
   - Male
   - Female
   - Choose not to answer

8. What program/major are you in?

9. Last 5 of your M number. (only to match 1st survey with 2nd survey)

10. Instructor
    - Mark only one oval.
    - Dr. Kastner
    - Dr. Torsella
Models 1: Survey (After)

1. How comfortable do you feel writing code that involves using "If Else" structure, now?  
   Mark only one oval.

   1  2  3  4  5  6  7  8  9  10
   Uncomfortable  Very Comfortable

Answer the next question based on this code snippet.

```
1  p = 25;
2  if (p == 10)
3      q = 31;
4  elseif (p > 5 & p < 23)
5      q = 41;
6  else
7      q = 15;
8  end
9  printf('Q = %i', q)
```

2. What would the output be for the above code snippet?

3. How much did the lab help you realize the concepts of an if else structure?  
   Mark only one oval.

   1  2  3  4  5  6  7  8  9  10
   Not at all  Extensively

4. How comfortable do you feel writing code that involves a "While Loop", now?  
   Mark only one oval.

   1  2  3  4  5  6  7  8  9  10
   Uncomfortable  Very Comfortable

https://docs.google.com/forms/d/1j2Mw4d4qiJaHfplQjIVYaKxT_plEnf08oRl4n_KKOM/edit
5. How much did including the hardware help reinforce the concept of if else?  
Mark only one oval.

Not at all  ○ ○ ○ ○ ○ ○ ○ ○ ○ ○  Strongly

6. Answer the next question based on this code snippet.

```plaintext
1
2
3 while p > 10
4     q = q+3;
5     p = p-10;
6 end
7 fprintf('Q = %i', q)
```

6. What would the output for the "while loop" code above be?

---

7. How much did the lab help you to understand while loops?  
Mark only one oval.

Not at all  ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Extensively

8. How much did including the hardware help reinforce the concept of while loop?  
Mark only one oval.

Not at all  ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Strongly

9. Last 5 of your M number. (only to match 1st survey with 2nd survey)

10. Instructor  
Mark only one oval.

○ Dr. Kastner
○ Dr. Torsella

https://docs.google.com/forms/d/1j2Mw4v1qj/0/p/0u3YVXhKJF_g/16f/0?ed=0KOMx/edit
A – 6: Raw Data – Complete Data

Data Before
Data After
### A – 7: Raw Data - Separated form

<table>
<thead>
<tr>
<th>Comfort with</th>
<th>What would th</th>
<th>How comfortable</th>
<th>What would th</th>
<th>Gender</th>
<th>What program</th>
<th>Last 5 of your</th>
<th>Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5</td>
<td>Male</td>
<td>Exploratory</td>
<td>48888</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>if something w b = 35</td>
<td>2</td>
<td>c = 1</td>
<td>Male</td>
<td>civil engineering</td>
<td>65810</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>8</td>
<td>An if else if str b=35</td>
<td>6</td>
<td>While loop 8</td>
<td>Male</td>
<td>Civil Engineer</td>
<td>21316</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>4</td>
<td>If something, t</td>
<td>35</td>
<td>While something</td>
<td>1 Male</td>
<td>Exploratory</td>
<td>21531</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>10</td>
<td>if checked con B = 35</td>
<td>10</td>
<td>while the conc C = 5</td>
<td>Female</td>
<td>Computer Sci</td>
<td>40712</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>10</td>
<td>If something o</td>
<td>35</td>
<td>While blank oc C = 6</td>
<td>Male</td>
<td>Electrical Eng</td>
<td>76721</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>6</td>
<td>B=35</td>
<td>7</td>
<td>a loop that inf</td>
<td>Male</td>
<td>Civil Engineer</td>
<td>37360</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>7</td>
<td>Its a loop that</td>
<td>35</td>
<td>I'm really confi I have no idea</td>
<td>Male</td>
<td>Exploratory</td>
<td>43226</td>
<td>Dr. Torsella</td>
</tr>
<tr>
<td>8</td>
<td>So the progr B = 35</td>
<td>8</td>
<td>While a cond C = 4</td>
<td>Male</td>
<td>Exploratory</td>
<td>23332</td>
<td>Dr. Torsella</td>
</tr>
</tbody>
</table>

### Code Snippet

```matlab
if (logical true)
    statements;
else if (logical true)
    statements;
else
    statements;
end
```

### Description

- **If...elseif...else** structure is used to choose which block of code to run based on conditions. It allows for multiple decision points.
- The `if` statement checks if a condition is true. If true, it executes the code block associated with it. If false, it moves to the `elseif` statements.
- `else` is executed if none of the `if` or `elseif` conditions are met.
- This structure is based on user input. If a variable isn't defined in an `if` statement, `elseif` or `else` should cover it.

### Example

```matlab
if B = 35
    a loop that c = 5
    Male
    Electrical Eng
    4736
    Dr. Torsella
else
    B = 35
    a loop that c = 5
    Male
    Electrical Eng
    4736
    Dr. Torsella
end
```

- Variables `B` and `C` are used to check conditions.
- The code runs based on the condition met.

###繼續

- Use this structure when there are multiple conditions to check.
- It's a flexible way to handle conditional logic in programming.

---

**Before Lab for Group 1**

69
Comfort with "If Else"

Each line gives alternate outcomes because based on what the variable is
- if a condition is met, a function will be performed.
- else-if a in a secondary if statement + else is if neither condition is met, then that function will be performed.

It's like a sentence, if something is a certain condition, or elseif it's something else, etc., then a certain output takes place.

If the first condition in the if statement is true, it sets executed. If it is false, the first if else gets executed if it is true. When the other preceding conditions are false, the else statement gets executed regardless.

It's a way of testing multiple conditions that progressively become more exclusive, where you only test the later conditions if the ones before it fail.

Code will run if first statement works, the just moves down the list until a line is found that will run.

It's a conditional statement that runs a line of code based off of what condition(s) are true.

The first if is the 1st condition then else if is if the "if" falls within its conditions. The else is if all other elseifs don't satisfy the condition.

A conditional statement that will "loop" multiple times.

Conditional statement begins with if, if this statement is false the code continues to elseif. If both conditions are not met, the program goes to else.

It acts as an elimination form of selection, if conditions don't meet one requirement it tries the next then executes code when requirements are met.

A structure which compares values.

Conditional statement that will run if a condition is true. Once the statement is true it stops running the code.

It is conditional that matlab looks at 1st, elseif is the second condition that matlab looks at 2nd. end is the way to close it.

A command or list of commands will run while a specified condition is satisfied.

A loop that requires certain stipulations like if elseif and else.

Continuous loop until a condition is met.

A loop running until a condition is not met.

Repeats a select amount of code until it meets a certain condition.

Conditional statement that goes through a set once.

Continuous loop until a condition is met.

A loop that allows you to put code inside of code.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

While loops are continuous operations that run until the while statement is no longer true.

While loops basically repeat until the correct condition is not met.

A loop that requires certain stipulations like if elseif and else.

While something is going on, certain conditions take place.

A conditional loop when you don't know when the condition will change.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.

A while loop is a loop that only runs when the initial condition is true.

A while loop allows you to perform commands while a condition is only true.
<table>
<thead>
<tr>
<th>Comfort if else</th>
<th>IF ELSE answer</th>
<th>Lab help with</th>
<th>If comfort writing</th>
<th>hardware help</th>
<th>answer while lab help</th>
<th>hardware help</th>
<th>Last 5 of your number. (only to match 1st survey with 2nd survey)</th>
<th>Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Q = 15</td>
<td>10 10</td>
<td>Q = 9</td>
<td>10</td>
<td>10</td>
<td>4736</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 15</td>
<td>2 1</td>
<td>3 2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>2 10</td>
<td>Q = 9</td>
<td>6</td>
<td>5</td>
<td>76721</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 q = 15</td>
<td>9 6</td>
<td>q = 9</td>
<td>7</td>
<td></td>
<td>69146</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Q = 41</td>
<td>10 10</td>
<td>Q = 9</td>
<td>10</td>
<td>7</td>
<td>40712</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Q = 41</td>
<td>10 7</td>
<td>Q = 3</td>
<td>10</td>
<td>10</td>
<td>2332</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 Q = 15</td>
<td>8 7</td>
<td>Q = 9</td>
<td>8</td>
<td></td>
<td>5480</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 q = 15</td>
<td>6 6</td>
<td>q = 9</td>
<td>6</td>
<td></td>
<td>65508</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 Q = 15</td>
<td>8 8</td>
<td>Q = 6</td>
<td>8</td>
<td>8</td>
<td>56010</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 Q=15</td>
<td>6 6</td>
<td>Q=3</td>
<td>7</td>
<td></td>
<td>60449</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 Q = 15</td>
<td>8 8</td>
<td>Q = 3</td>
<td>8</td>
<td></td>
<td>81670</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 Q=15</td>
<td>7 6</td>
<td>Q=12</td>
<td>6</td>
<td>5</td>
<td>17750</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 Q=41</td>
<td>8 7</td>
<td>12</td>
<td>7</td>
<td>6</td>
<td>67850</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>infinite loop or error</td>
<td>9 5</td>
<td>3 6</td>
<td>10</td>
<td>65810</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 Q = 15</td>
<td>9 9</td>
<td>Q = 3</td>
<td>9</td>
<td>9</td>
<td>62654</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 Q = 15</td>
<td>15 7</td>
<td>15</td>
<td>7</td>
<td></td>
<td>28310</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 Q = 15</td>
<td>8 7</td>
<td>inf</td>
<td>7</td>
<td>7</td>
<td>37360</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>6 7</td>
<td>Q=9</td>
<td>6</td>
<td>5</td>
<td>69269</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 Q = 41</td>
<td>9 9</td>
<td>Q = 3</td>
<td>10</td>
<td>8</td>
<td>17353</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Q = 41</td>
<td>9 9</td>
<td>12</td>
<td>9</td>
<td>9</td>
<td>16626</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 Q = 15</td>
<td>6 6</td>
<td>Q = 9</td>
<td>5</td>
<td>1</td>
<td>9104</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 Q = 15</td>
<td>6 6</td>
<td>Q = 9</td>
<td>5</td>
<td>1</td>
<td>32561</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>7 8</td>
<td>Q = 9</td>
<td>9</td>
<td>7</td>
<td>90873</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 Q = 15</td>
<td>6 6</td>
<td>Q = 9</td>
<td>6</td>
<td>6</td>
<td>71586</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 Q=15</td>
<td>41 10</td>
<td>9</td>
<td>13</td>
<td>10</td>
<td>35657</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 15</td>
<td>10 1</td>
<td>11</td>
<td>3</td>
<td>3</td>
<td>47850</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 15</td>
<td>8 7</td>
<td>3</td>
<td>7</td>
<td>7</td>
<td>91018</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Q=15</td>
<td>1 10</td>
<td>Q = 3</td>
<td>1</td>
<td>1</td>
<td>10910</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 Q=15</td>
<td>3 3</td>
<td>Q=9</td>
<td>3</td>
<td>3</td>
<td>74355</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 Q=41</td>
<td>8 8</td>
<td>Q=9</td>
<td>8</td>
<td>8</td>
<td>37743</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 q=15</td>
<td>7 1</td>
<td>q=9</td>
<td>3</td>
<td>5</td>
<td>45409</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 41</td>
<td>5 5</td>
<td>24</td>
<td>5</td>
<td>5</td>
<td>64974</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>10 8</td>
<td>Q = 9</td>
<td>10</td>
<td>10</td>
<td>6624</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>7 7</td>
<td>34</td>
<td>7</td>
<td>8</td>
<td>9503</td>
<td>Dr. Torsella</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After Lab for Group 1
<table>
<thead>
<tr>
<th>Comfort if else</th>
<th>IF ELSE</th>
<th>a lab help if el</th>
<th>comfort <em>While Loop</em> hardware help</th>
<th>answer <em>while</em> lab help</th>
<th>while hardware hel</th>
<th>Last 5 of you Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Q = 15</td>
<td>7</td>
<td>8</td>
<td>5 Q = 9</td>
<td>6</td>
<td>6</td>
<td>90304 Dr. Kastner</td>
</tr>
<tr>
<td>7 q = 15</td>
<td>2</td>
<td>5</td>
<td>6 q = 3</td>
<td>6</td>
<td>5</td>
<td>10504 Dr. Kastner</td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>9</td>
<td>9</td>
<td>10 Q = 9</td>
<td>9</td>
<td>9</td>
<td>85123 Dr. Kastner</td>
</tr>
<tr>
<td>6 q = 15</td>
<td>5</td>
<td>3</td>
<td>5 q = 9</td>
<td>5</td>
<td>5</td>
<td>57027 Dr. Kastner</td>
</tr>
<tr>
<td>8 q = 41</td>
<td>7</td>
<td>6</td>
<td>7 q = 3</td>
<td>6</td>
<td>7</td>
<td>17169 Dr. Kastner</td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>10</td>
<td>10</td>
<td>10 Q = 9</td>
<td>10</td>
<td>10</td>
<td>83833 Dr. Kastner</td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>10</td>
<td>10</td>
<td>6 Q = 9</td>
<td>10</td>
<td>6</td>
<td>54688 Dr. Kastner</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 q = 15</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>5</td>
<td>61844 Dr. Kastner</td>
</tr>
<tr>
<td>7 q = 15</td>
<td>4</td>
<td>5</td>
<td>4 q = 3</td>
<td>5</td>
<td>5</td>
<td>73723 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>4</td>
<td>7</td>
<td>6 q = 3</td>
<td>6</td>
<td>5</td>
<td>89568 Dr. Kastner</td>
</tr>
<tr>
<td>10 q = 15</td>
<td>1</td>
<td>10</td>
<td>1 q = 9</td>
<td>1</td>
<td>1</td>
<td>79818 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>7</td>
<td>8</td>
<td>5 Q = 9</td>
<td>6</td>
<td>6</td>
<td>77214 Dr. Kastner</td>
</tr>
<tr>
<td>7 q = 15</td>
<td>3</td>
<td>5</td>
<td>NA</td>
<td>4</td>
<td>5</td>
<td>1177 Dr. Kastner</td>
</tr>
<tr>
<td>7 q = 15</td>
<td>4</td>
<td>6</td>
<td>3 q = 3</td>
<td>2</td>
<td>1</td>
<td>4475 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>8</td>
<td>7</td>
<td>7 q = 9</td>
<td>7</td>
<td>7</td>
<td>25685 Dr. Kastner</td>
</tr>
<tr>
<td>9 q = 15</td>
<td>3</td>
<td>9</td>
<td>4 q = 9</td>
<td>4</td>
<td>4</td>
<td>79750 Dr. Kastner</td>
</tr>
<tr>
<td>7 Q = 41</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>58751 Dr. Kastner</td>
</tr>
<tr>
<td>5 q = 15</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2808 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>9</td>
<td>9</td>
<td>10 Q = 0</td>
<td>9</td>
<td>9</td>
<td>138 Dr. Kastner</td>
</tr>
<tr>
<td>10 q = 15</td>
<td>7</td>
<td>10</td>
<td>4 q = 9</td>
<td>6</td>
<td>5</td>
<td>29708 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>7</td>
<td>5</td>
<td>8 Q = 3</td>
<td>7</td>
<td>7</td>
<td>20287 Dr. Kastner</td>
</tr>
<tr>
<td>8 q = 41</td>
<td>7</td>
<td>7</td>
<td>8 q = 7</td>
<td>7</td>
<td>7</td>
<td>3909 Dr. Kastner</td>
</tr>
<tr>
<td>9 Q = 15</td>
<td>5</td>
<td>7</td>
<td>8 Q = 9</td>
<td>7</td>
<td>7</td>
<td>42030 Dr. Kastner</td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>8</td>
<td>10</td>
<td>7 Q = 9</td>
<td>7</td>
<td>7</td>
<td>2656 Dr. Kastner</td>
</tr>
<tr>
<td>10 q = 15</td>
<td>8</td>
<td>10</td>
<td>1 q = 9</td>
<td>10</td>
<td>1</td>
<td>31719 Dr. Kastner</td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>1</td>
<td>10</td>
<td>1 Q = 9</td>
<td>1</td>
<td>1</td>
<td>44245 Dr. Kastner</td>
</tr>
<tr>
<td>6 q = 15</td>
<td>7</td>
<td>7</td>
<td>5 q = 3</td>
<td>6</td>
<td>7</td>
<td>51489 Dr. Kastner</td>
</tr>
<tr>
<td>9 q = 15</td>
<td>5</td>
<td>7</td>
<td>8 inf loop</td>
<td>6</td>
<td>6</td>
<td>84675 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 15</td>
<td>8</td>
<td>8</td>
<td>9 Q = 9</td>
<td>9</td>
<td>10</td>
<td>7698 Dr. Kastner</td>
</tr>
<tr>
<td>10 Q = 15</td>
<td>10</td>
<td>10</td>
<td>6 Q = 9</td>
<td>10</td>
<td>7</td>
<td>4760 Dr. Kastner</td>
</tr>
<tr>
<td>10 q = 15</td>
<td>8</td>
<td>8</td>
<td>5 q = 9</td>
<td>8</td>
<td>5</td>
<td>2571 Dr. Kastner</td>
</tr>
<tr>
<td>8 Q = 41</td>
<td>8</td>
<td>8</td>
<td>5</td>
<td>9</td>
<td>5</td>
<td>13444 Dr. Kastner</td>
</tr>
<tr>
<td>9 Q = 15</td>
<td>9</td>
<td>9</td>
<td>9 Q = 3</td>
<td>9</td>
<td>9</td>
<td>345 Dr. Kastner</td>
</tr>
<tr>
<td>6 Q = 15</td>
<td>6</td>
<td>4</td>
<td>7 Q = 3</td>
<td>6</td>
<td>7</td>
<td>63452 Dr. Kastner</td>
</tr>
<tr>
<td>8 q = 15</td>
<td>5</td>
<td>8</td>
<td>5 q = 9</td>
<td>6</td>
<td>7</td>
<td>78508 Dr. Kastner</td>
</tr>
<tr>
<td>8 q = 15</td>
<td>4</td>
<td>8</td>
<td>3 q = 9</td>
<td>5</td>
<td>5</td>
<td>43340 Dr. Kastner</td>
</tr>
<tr>
<td>5 q = 15</td>
<td>5</td>
<td>5</td>
<td>5 q = 9</td>
<td>5</td>
<td>5</td>
<td>2009 Dr. Kastner</td>
</tr>
<tr>
<td>8 q = 15</td>
<td>8</td>
<td>8</td>
<td>6 q = 9</td>
<td>8</td>
<td>7</td>
<td>62527 Dr. Kastner</td>
</tr>
<tr>
<td>6 q = 15</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>9</td>
<td>6</td>
<td>98153 Dr. Kastner</td>
</tr>
</tbody>
</table>

After Lab for Group 2