Attention Mission Specialist:

Your assignment is to complete all missions and sub-missions located on these cards.

Once all missions are completed, start thinking creatively and searching out new projects to build and share online.

Experiment with the DuinoKit and have fun learning. Use your mind, be creative and think outside the box. Dream, Inspire and Invent!

PS: This Message will self destruct in 5 minutes.
Be sure to visit the “Show and Tell” forum on the DuinoKit.com website. This forum will allow easy access to project code and these mission cards for building and modifying projects.

Although there are several different programming environments and languages that can be used with the Arduino microprocessor, we suggest getting started with the typical language (C++) and suggested programming environment provided on the Arduino.cc website.
Getting Started – Arduino Software
Available at www.Arduino.cc
Or www.Create.Arduino.cc

If this is your first time working with Arduino, you will need to install the programming environment from one of these two recommended sources.

www.Create.Arduino.cc Offers a simple browser-base plug-in platform with pre-installed drivers and online program uploading and storage for your projects.

www.Arduino.cc is the standard programming environment, but sometimes it has difficulties with drivers or ports for programming. Click in the Download link and look for the program installer and instructions for your operating system.

If having problems, please consult the Arduino.cc forum, the DuinoKit.com “Show and Tell” forum or send a message to us at Support@DuinoKit.com

Thank you for choosing the DuinoKit prototyping system. Please share your learning and creations on our Show and Tell forum.
After installing the Arduino software (IDE), be certain to select Arduino NANO for the board type and the Atmega328 for the processor.

Each DuinoKit uses an Arduino NANO 3.0 w/ ATmega328 chip. This will need to be selected in your Arduino software (IDE). The “Port” will most likely be the most recent port found after attaching the Arduino Nano to the USB cable.

NOTE:
In place of the traditional Arduino software, some users prefer the new site www.Create.Arduino.cc for online code storage and compiling of projects. This also seems to be a good choice if you are having trouble getting the drivers installed on your computer.
MISSIONS TO ACCOMPLISH

Here are your missions, should you choose to complete them.

- Mission #1 - Blinking LED
  - Sub-mission

- Mission #2 - Scrolling LEDs
  - Sub-mission

- Mission #3 - Displaying Text on an LCD
  - Sub-mission

- Mission #4 - Going Portable
  - Sub-mission

- Mission #5 - Light Dimmer
  - Sub-mission

- Mission #6 - Distance & Sound
  - Sub-mission

- Mission #7 - Weather Station
  - Sub-mission

- Mission #8 - Real Time Clock
  - Sub-mission

- Mission #9 - Understanding Data
  - Sub-mission

- Mission #10 - Memory Game
  - Sub-mission

Parent/Guardian
Sign to verify that all missions have been completed
### Missions to Accomplish

Here are your missions, should you choose to complete them.

<table>
<thead>
<tr>
<th>Mission #11 - Color Changing LED</th>
<th>Mission #17 - Infrared Code Detector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-mission</td>
<td>Sub-mission</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mission #12 - Lie Detector</td>
<td>Mission #18 - Infrared Light Control</td>
</tr>
<tr>
<td>Sub-mission</td>
<td>Sub-mission</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mission #13 - Morse Code Device</td>
<td>Mission #19 - Fortune Teller</td>
</tr>
<tr>
<td>Sub-mission</td>
<td>Sub-mission</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mission #14 - Rotary Encoder</td>
<td>Mission #20 - Reaction Timer</td>
</tr>
<tr>
<td>Sub-mission</td>
<td>Sub-mission</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mission #15 - Guess a Number</td>
<td>Ongoing Mission - “Show and Tell”</td>
</tr>
<tr>
<td>Sub-mission</td>
<td>Sub-mission</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Mission #16 - Light Change Alarm</td>
<td></td>
</tr>
<tr>
<td>Sub-mission</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Parent/Guardian**

Sign to verify that all missions have been completed.
MISSION #1 - BLINK AN LED

For our first mission, we start with a basic concept. We will learn how to turn a LED (Light Emitting Diode) on and off using the Arduino microprocessor. This introduces the concept of sketches, which are just a series of instructions that are carried out when executed or started by a computer.

The Arduino goes through the instructions or “Sketch” in a sequence and this program will run every time the Arduino is switched on. If you have not already installed the Arduino IDE (Integrated Development Environment) on your computer, please follow the Getting Started Handout included with your DuinoKit or you can get the instructions and software by going to www.Arduino.cc. You will then upload the sketch, or code, to your Arduino processor.

This project code can also be found under File-->Examples-->Basics-->01.Blink

![Arduino Board with LEDS and Resistors](image)

All Project code (or “Sketches”) are available to copy/paste from the “Show and Tell” portion of the DuinoKit website.
Step 1: Select an LED on the DuinoKit. You may choose any of the LED lights and connect the left hand side pin (positive) to pin 13 located below the Arduino Processor. Connect the other side (negative/ground) and to the GND pin (short for ground) located beneath the Arduino Processor. The LEDs have a resistor built in.

Step 2: Connect the Arduino Nano Board to your computer by using the USB cable provided. It will connect from the Arduino Nano Board to the USB plug on your computer.

Step 3: Open the Arduino IDE program on the computer, select “File” on the top tool bar. Then select “Examples” and “0.1 Basic” then select “Blink” from the drop down menu.

Step 4: Verify the sketch and then upload to your DuinoKit by pressing the Arrow pointing to the right on the tool bar or Selecting the “File” menu and then selecting “Upload”.

The LED should start to flash. The code on the computer screen is called a sketch. The sketch controls all the action in this project, but how does it work? In simple terms this is what the program does:
1. We made pin 13 an output
   (the output here is to power the LED)
2. We start a loop
3. The LED is switched on
4. Wait for 1 second
5. The LED is switched off
6. Wait for 1 second
7. Go back to the start of loop

LEDs require a resistor to limit the current or they will burn out. These resistors have been built into the DuinoKit design so you do not have to worry about damage to the LEDs.

YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!

- [ ] Edit code to change the blink rate of the LED
- [ ] Modify the wiring and/or code to blink two LEDs
- [ ] Modify the wiring and/or code to alternate LEDs on/off
MISSION #2 - SCROLLING LEDs (KNIGHT RIDER)

This mission is modelled after the K.I.T.T. Car from Knight Rider, a popular television series from the 80's. The K.I.T.T. car had lots of LEDs and gadgets to create many special effects. So, in order to learn more about LEDs and sequential programming we thought this would be a nice fit. Instead of turning each LED on and off individually as we did in the first project, this sketch uses the “For Loop” command and shows how to use an “array”. An array is a variable with multiple parts. It might help think of an array as an ice cube tray. Each separate cube represents a different value in the array.

We added in a current limiting resistor to all the LEDs and connected all the (-) of the LED arrays here together to make the hook-up easier.
Step 1: Select any six of the blue LEDs that you want. Connect a short wire from the left hand side (positive) of each LED to the pins 2 through 7 on the Arduino microprocessor. Your first LED will go to pin 2. The second LED to pin 3 and so forth until you reach pin 7.

Step 2: For this step you need to ground the LEDs. So in order to do this. You will run a small wire from the last pin at the bottom on the negative side of the row of Blue LEDs to the GND or (ground) pin underneath the Arduino Board. By selecting the last pin it will ground all the LEDs on the row so you don't have to ground each LED individually.

Step 3: Connect the Arduino Board to your computer using the USB cable provided.

Step 4: Open the program “Arduino IDE” on your computer. You need to copy and paste the “sketch” or code for this project from DuinoKit.com on our “Show and Tell” forum. It is titled “Mission #2 - Scrolling LEDs”.

Step 5: Verify your sketch and then upload to your Arduino by pressing the Arrow key on the toolbar or going to File menu and clicking on Upload.

This example shows you how you can turn on a sequence of pins whose numbers are neither contiguous nor necessarily sequential. To do this, you can put the pin numbers in an array and then use “for Loops” to iterate over the array. This example makes use of 6 LEDs connected to the pins 2-7 on the Arduino and the resistors are built into the DuinoKit. The order of the LEDs is determined by their order in the array, not by their physical order. This technique of putting the pins in an array is very handy. You don't have to have the pins sequential to one another, or even in the same order. You can rearrange them however you want. This can be a fun program to modify and change for different effects.

YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!

☐ Edit code and wiring to scroll 10 LEDs in order and adjust the timing.

☐ Can you program the LEDs to only travel in one direction then restart?

☐ What does K.I.T.T. stand for and how much did the car sell for in 2014?
MISSION #3 - DISPLAY TEXT ON AN LCD

LCD stands for liquid crystal display.

The LCD can be hooked to various different pins on the Arduino, the key is the one line of code that tells the Arduino how to send data to the LCD. Look in the code and see if you can make sense of the wiring and the following lines of code.

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
// LCD PinOut (RS, E, D4, D5, D6, D7)
```

**NOTE:** The // before a line is for making a comment in the code. Notice how the text is grey in color. This line serves as nothing more than a comment and is not processed by the Arduino.

Additional information on the LCD display can be found here: http://www.arduino.cc/en/Tutorial/LiquidCrystal
Step 1: Using the pins located around the Arduino board as well as the LCD pins, Wire the Arduino board according to the chart to the right.

Step 2: Connect the Arduino Board to your computer using the USB cable provided.

Step 3: Open the Arduino IDE on your computer. You need to copy and paste the “sketch” or code for this project from DuinoKit.com on our “Show and Tell” forum. It is titled “LCD Display”.

Step 4: Verify your sketch and then upload to your Arduino by pressing the Arrow key on the tool bar or going to the File menu and clicking on Upload. After uploading the sketch you will see the LCD light up. It will say “Inspire & Invent”. If you don’t see it right away then turn the potentiometer knob labelled “contrast” to the left and right and you will notice that the contrast on the LCD will change from light to dark.

Typical Wiring Chart for LCD Display

| LCD Pin D7  | Arduino Pin 2 |
| LCD Pin D6  | Arduino Pin 3 |
| LCD Pin D5  | Arduino Pin 4 |
| LCD Pin D4  | Arduino Pin 5 |
| LCD Pin E   | Arduino Pin 11|
| LCD Pin RS  | Arduino Pin 12|
| LCD (-) Negative | LCD Pin RW |
| LCD Pin RW  | LCD Pin GND  |
| LCD (+) Positive | LCD Pin VDD |
| LCD Pin VDD | Arduino 5V   |
| LCD Pin GND | Arduino GND  |

Your Mission.... Should You Choose To Accept It!

☐ Edit text to display your name.

☐ What does LCD stand for?

☐ Can you figure out what pins control the LCD backlight?
MISSION #4 - GOING PORTABLE

We are going to learn a quick and simple hookup so that we can now run our projects on battery power. There is not much to this project and the Arduino takes care of switching from battery to USB power as needed. The Arduino NANO (and most other Arduino compatible microprocessor boards) have a built-in power regulator that will allow the microprocessor to be hooked up directly up to 20 volts. The DuinoKit Jr. only provides a 9V power output. We are going to hook our Arduino microprocessor directly to the 9V connection on the battery. The DuinoKit also has a mini LED that will light up when the power switch is on.

Although it is stated the Arduino can be hooked up to 20 volts, it is suggested not to exceed 12v for continued use.

Be sure to check this if you build a project with a different Arduino board.
The Arduino NANO board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts."

We have found that using 9V works well. You can simply connect the + end of your battery to Arduino VIN and the - end to Arduino ground (fig 1). You should see the green light on the Arduino turn on to indicate that it is powered.

Switch override terminals
Notice the switch symbol on top.
Onboard Switch
Power Indicator LED (light)
Tiny surface mounted resistor to limit the current to the LED

WARNING...
Do NOT hook the 9V connectors anywhere except VIN and GND pins!

YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!

☐ If you leave the switch on with no wires hooked up, will the battery still be drained? Why or why not?

☐ Add battery function to one of your previous projects.

☐ Use a jumper wire across override terminals to bypass switch.
MISSION #5 - LIGHT DIMMER

In this project, we will take an LED and create a dimmer by adding a potentiometer. A potentiometer is a variable resistor that changes resistance when you turn the knob. They are commonly used to control electrical devices such as volume on audio equipment. The potentiometer, “pot” for short, creates a continuous analogue signal which represents physical measurements.

This is where things get a bit tricky, the Arduino is a digital machine and can only read a digital input (numbers). So the analogue signal from the pot needs to be changed into numbers (in this case voltage). Different analogue devices will have various outputs depending on the device and circumstances. We will use the MAP function to change the output value (1-1023) to a brightness level for this mission. Look in the code to see where this is taking place.

A potentiometer has 3 pins. The middle pin is the signal and goes to the Arduino analogue pin (A0 - A5) to read the value. The side pins go to the +5V and GND. Which way around doesn’t matter.
Humans perceive the world in analogue. Everything we see and hear is a continuous transmission of information to our senses. This continuous stream is what defines analogue data. Digital information on the other hand, estimates analogue data using only numbers. The LED is actually being switched on and off very quickly. This is so quick that our eyes compensate and we see continuously lit LED but at varying light levels. This is known as Persistence of Vision.

**Step 1:** Using a small wire, connect one side of the Potentiometer to the +5V pin beside the Arduino. Connect the other side of the potentiometer to the GND. It doesn't matter which side is which.

**Step 2:** Connect the positive (+) pin of an LED to pin D9. Connect the other side of the LED the negative (-) to one of the GND (ground) pins beside the Arduino.

**Step 3:** Connect the Arduino Board to your computer using the USB cable provided.

**Step 4:** Open the Arduino IDE on your computer. You need to copy and paste the “sketch” or code for this project from www.DuinoKit.com on our “Show and Tell” forum. It is titled “Light Dimmer”.

**Step 5:** Verify your sketch and then upload to your Arduino by pressing the Arrow key on the tool bar or going to File menu and clicking on Upload. After uploading the sketch you will see the LED light up. If you turn the potentiometer to the left and right you will notice that the LED is getting brighter and dimmer.

**YOUR MISSION...SHOULD YOU CHOOSE TO ACCEPT IT!**

- [ ] Add another LED to this project.
- [ ] Edit code such that one LED gets brighter while the other becomes dimmer
- [ ] Wire the CDS cell in place of the potentiometer to emulate adaptive lighting
  (See Mission #9 for more info about the CDS cell or photo resistor)
MISSION #6 - DISTANCE SENSOR SOUND DEVICE

For this mission you will explore the ultrasonic distance module and connect it to the piezo buzzer to create different pitched sounds based upon the distance from the sensor.

Step 1: Wire the Arduino Nano according to the chart to the left.
Step 2: Connect the Arduino Nano to your computer using the USB cable provided.
Step 3: Open the Arduino IDE on your computer. You need to copy and paste the “sketch” or code for this project from DuinoKit.com on the “Show and Tell” forum. It is titled “Mission #6 – Distance Sensor Sound Device.”
Step 4: Verify the sketch and upload to the Arduino Nano by pressing the Arrow key on the toolbar or going to the File menu and clicking on Upload.

Wiring Chart

Ultrasonic Sensor
VCC connects to +5V
Trig connects to Pin2
Echo connects to Pin3
GND connects to GND

Piezo Buzzer
+ to D4
- to GND
The Distance "Ping" Module (also sometimes called an Ultrasonic sensor) uses high frequency sound waves (not audible to humans) to perform calculations to determine distances. Most Distance sensors have two "eyes" -- a wave is emitted from the "eyes" and a reflection or "bounce back" will also be detected. The waves travels at a specific speed, so by measuring the time it takes for a wave to emit, bounce off an object (like a wall or pet), and return to the "eyes", the distance between the Distance module and a nearby object can be determined. This is similar to the sonar “sight” used by live bats.

A piezo buzzer (also called a piezo disc) does exactly what it says -- it buzzes. But the sound it produces can range from a high pitch or low pitch. A simple circuit can be built that will change the pitch of the buzzer depending on how close an object (such as your hand) is to the Ultrasonic sensor.

Voltage to the piezo buzzer is used to create sound. A ceramic plate inside the buzzer vibrates based on voltage pulses that are received by the Nano.

**YOUR MISSION...SHOULD YOU CHOOSE TO ACCEPT IT?**

- [ ] Edit the code so the pitch goes down as your hand gets closer.
- [ ] Edit code so there is no noise until within range (simple alarm)
- [ ] Edit code so it beeps on/off with various frequency.
  Think of making the buzzer “blink” on and off as the distance changes.
MISSION #7 – WEATHER STATION

This project utilizes the Digital Humidity and Temperature Sensor and demonstrates how to measure the temperature and humidity and display it on the LCD screen. Knowing the relative humidity indoors is important because humidity affects comfort, health, and the condition of household possessions. When indoor humidity is too high (above 55%), people tend to consider the room to be hot and stuffy even though the temperature may be at a level that is normally comfortable. When the indoor humidity drops below 25%, the dry air also reduces comfort. Keeping humidity in the optimal range indoors helps to control bacteria, viruses, fungi (including mold), allergies, and asthma. Measuring indoor humidity provides a way to check how well humidifiers, dehumidifiers, and air conditioners are working. Proper humidity control can also help equipment performance and reduce energy consumption. This hygrometer is a basic relative hygrometer. It will show the temperature and the relative humidity.
Step 1: Use the wiring layout chart to the left to build your project. The pin on the left side will connect to the corresponding pin on the right side. So connect the DHT-11 VCC pin will connect to the Arduino 5v Pin. Then continue down the chart until finished.

Step 2: Connect the Arduino Board to your computer using the USB cable provided.

Step 3: Open the Arduino IDE on your computer. You need to copy and paste the “sketch” or code for this project from DuinoKit.com on our “Show and Tell” forum. It is titled “The Weather Station”.

Step 4: If your sketch is producing errors when trying to compile and upload, you may need to install a module library that will instruct the Arduino how to handle data from the DHT-11 sensor. The library file and instructions are available on the “Show and Tell” forum.

Wiring Layout Chart

<table>
<thead>
<tr>
<th>DHT-11 VCC</th>
<th>Arduino 5v</th>
</tr>
</thead>
<tbody>
<tr>
<td>DHT-11 DAT</td>
<td>Arduino Pin 8</td>
</tr>
<tr>
<td>DHT-11 GND</td>
<td>Arduino GND</td>
</tr>
<tr>
<td>LCD GND</td>
<td>Arduino GND</td>
</tr>
<tr>
<td>LCD VDD</td>
<td>Arduino 5V</td>
</tr>
<tr>
<td>LCD RS</td>
<td>Arduino Pin 12</td>
</tr>
<tr>
<td>LCD RW</td>
<td>Arduino GND</td>
</tr>
<tr>
<td>LCD E</td>
<td>Arduino Pin 11</td>
</tr>
<tr>
<td>LCD D4</td>
<td>Arduino Pin 5</td>
</tr>
<tr>
<td>LCD D5</td>
<td>Arduino Pin 4</td>
</tr>
<tr>
<td>LCD D6</td>
<td>Arduino Pin 3</td>
</tr>
<tr>
<td>LCD D7</td>
<td>Arduino Pin 2</td>
</tr>
<tr>
<td>LCD (+)</td>
<td>Arduino 5V</td>
</tr>
<tr>
<td>LCD (-)</td>
<td>Arduino GND</td>
</tr>
</tbody>
</table>

YOUR MISSION...SHOULD YOU CHOOSE TO ACCEPT IT!

- Edit the code so the temperature is displayed in degrees Celsius.
- At what temp does Celsius and Fahrenheit have the same value?
- Can you set the piezo buzzer or LED to turn on when the temp gets too high? (Breath on the sensor to easily see temp change)
MISSION #8 – REAL TIME CLOCK

For this mission we are going to make a clock display with larger number readout. We are going to use the Real Time Clock module for the project and this requires installing an additional library file to your Arduino software so the microprocessor knows how to read the data from the clock module. This library can be found online on the DuinoKit “Show and Tell” forum with installation instructions.

Module libraries are necessary to use different commands in code to interact with various modules. As you get more involved with Arduino, you will find and maybe even start to write custom libraries for your projects.

Wiring the Clock
The Real Time Clock (RTC) requires:
SDA to Arduino A4
SCL to Arduino A5
Instead of using the typical little characters on the LCD display, these custom characters were designed (by my son Isaac) for a large digit display for the time.

To set the time on the RTC, we will upload the code the first time and this will set you computer time in the RTC. The we will comment the time setting lines and reload the code again. If we leave in this time setting code, every time the Arduino is restarted, it will default back to the time you compiled your sketch and uploaded your code.

The A4 and A5 pins on the Arduino use a special communication protocol called I2C. This allows multiple modules to be attached to the same two pins for data transfer.

Wiring Layout Chart

<table>
<thead>
<tr>
<th>RTC SDA</th>
<th>Arduino Pin A4</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTC SCL</td>
<td>Arduino Pin A5</td>
</tr>
<tr>
<td>RTC GND</td>
<td>Arduino GND</td>
</tr>
<tr>
<td>RTC VCC</td>
<td>Arduino 5V</td>
</tr>
<tr>
<td>LCD GND</td>
<td>Arduino GND</td>
</tr>
<tr>
<td>LCD VDD</td>
<td>Arduino 5V</td>
</tr>
<tr>
<td>LCD RS</td>
<td>Arduino Pin 12</td>
</tr>
<tr>
<td>LCD RW</td>
<td>Arduino Pin 11</td>
</tr>
<tr>
<td>LCD E</td>
<td>Arduino Pin 10</td>
</tr>
<tr>
<td>LCD D4</td>
<td>Arduino Pin 5</td>
</tr>
<tr>
<td>LCD D5</td>
<td>Arduino Pin 4</td>
</tr>
<tr>
<td>LCD D6</td>
<td>Arduino Pin 3</td>
</tr>
<tr>
<td>LCD D7</td>
<td>Arduino Pin 2</td>
</tr>
<tr>
<td>LCD (+)</td>
<td>Arduino Pin 13</td>
</tr>
<tr>
<td>LCD (-)</td>
<td>Arduino GND</td>
</tr>
</tbody>
</table>

**YOUR MISSION...SHOULD YOU CHOOSE TO ACCEPT IT!**

☐ Identify which 2 pins on the LCD are used to power the backlight LED.

☐ Edit wiring to make your clock run on battery power. (Not recommended for long durations as it will drain the battery)

☐ Edit code so that you can adjust the brightness of the backlight on the LCD. Future missions will make this adjustment automatic.
MISSION #9 - UNDERSTANDING DATA IN SERIAL MONITOR

This would be a good time to do a little background discussion to learn more about receiving data values from sensors and then learning how to respond to this data. We will be reading an analogue value from the CDS cell (light sensor) and then translating this data into a value to change the brightness of an LED.

There are essentially 2 different types of data that we can read and process, analogue and digital. An analogue value is simply a voltage from 0 to 5V and will be read as an integer value from 0 to 1024 for most sensors. Digital values can take on very different forms depending on the sensor.

For this mission we are going to write a script that will look for and display both types of data signals and feed the values through the USB cable so we can start to understand what is happening as we begin creating more complex projects.

See back side of mission card for information on displaying potentiometer value is serial monitor.
After uploading a sketch, the button in the corner will open a new window were communication to/from the Arduino is possible through the USB cable.

In most, but not all of the Arduino microprocessors, opening this window will reset the sketch and run from the beginning.

Once hooking an analogue component/module to the AO pin on the Arduino you will be able to view the output for this component in the serial monitor on your computer. This will typically be a number from 0 to 1023. The Map function was used to convert this to a 1-10 value.


Now try hooking the pot in place for the CDS cell. How does the range of values vary?

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

- [ ] Edit code so that it will display values 1-100 while viewing serial monitor.
- [ ] Try hooking up the potentiometer in place of the CDS cell.
  
  See Mission #5 if unsure how to hook up the potentiometer.
- [ ] Edit MAP function so that the CDS returns values 1-10 based on your current lighting situation.
MISSION #10 - MEMORY GAME (SIMON)

Now it is time for a little bit of fun..... You may recall playing the memory game “Simon” where you have to repeat a series or colored light sequences. After finding code for this project online we thought it would be good to add in a score keeper to track current and high scores while playing.

See back side of this mission card displays the LCD wiring diagram used for keeping track of your score.

This classic game can be played using only the buttons and LEDs as displayed on this side.
This is a continuation of wiring for the memory game. Only the buttons and LEDs are necessary for game playing. Wiring in the LCD will allow for a digital score keeper.

The LCD wiring in this project was changed from the typical hookup pins to Analogue pins. Can you find the line in the code where these changes are made to correspond to this wiring diagram?

YOUR MISSION...SHOULD YOU CHOOSE TO ACCEPT IT!

☐ There is an error in the wiring or code matching the buttons and LEDs, Can you find and fix this error?

☐ Hook up this project to run on battery power

☐ Attach a volume control to the buzzer
MISSION #11 - COLOR CHANGING LED

I'm sure you have seen the color changing night lights or LED lights that are getting very popular these days. This mission will help you understand a little bit better how it is possible to obtain all the different colors and how multi-color LEDs actually work.

This project will step through code using a formula to automatically adjust the levels of Red, Blue and Green on the colored LED.

These levels could be adjusted through various sensors to modify the RGB inputs.

LEDs must be attached to PWM pins to vary the brightness of the LED.
After uploading a sketch, the button in the corner will open a new window where communication to/from the Arduino is possible through the USB cable.

In most, but not all of the Arduino microprocessors, opening this window will reset the sketch and run from the beginning.

The sketch for this mission activates both the serial monitor and the LCD display if they are hooked up. The wiring instructions are in the sketch itself to hook up the LCD.

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

☐ Open the serial monitor and see the color “values” as the LED changes

☐ Hook up the LCD to display the color values as the LED changes

☐ Wire the Red, Green and Blue LEDs individually (above buttons)
MISSION #12 - LIE DETECTOR

No “Secret Agent” tool kit is complete without a Lie Detector. This is a very simple project based on measuring voltage through a person’s fingertips. Don’t worry, you will not feel anything as the voltage is in microvolts. The idea is simple, when a person lies under questioning, they will increase sweat production and this project will sense this change in body chemistry.

As with any lie detector, you will need to get a base reading. This process has been automated with code and is self adjusting. Once adjusted, the DuinoKit will sense slight changes in skin resistance and voltage change indicating a possible lie.

Please keep in mind this is for entertainment and education purposes only!

We are not actually the CDS cell in this project, we are simply using the 10K resistor from this module as a “pull-down” resistor to get a clean base reading.
While developing this mission, the DuinoKit team decided this would make a fun “Learn to Solder” kit as a stand-alone project. To make this easier to a printed circuit board design, the wiring for the LCD was changed around a little bit and can be wired as shown here.

If you build this learn to solder kit, the wiring will be as show on this card just in case you want to modify or hack this project.

The circuit:
* LCD RS pin to A5
* LCD Enable pin A4
* LCD D4 pin to A3
* LCD D5 pin to A2
* LCD D6 pin to A1
* LCD D7 pin to A0
* LCD R/W pin to ground
* GND and +5V wires as shown

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

- Although it is not necessary for operation, hook up the LCD as shown for additional program feedback.
- Test the with your friends. Remember, this is not 100% accurate!
- Make better sensors by connecting wires to foil or a copper penny to have more contact with skin. The readings will be more accurate.
MISSION #13 - MORSE CODE DEVICE

Morse code was the only way to transmit messages over long distances many years ago. Although most communication methods are far more advanced, Morse code is still in use today. This mission will allow you to type in data in the serial monitor, send it to the Arduino and your message will them be translated into Morse code via light and sound.

There was no shortage of Morse Code devices online so we adapted one that we found, added in audio and volume control (See Mission #5).

With enough practice, you will start to recognize letters by hearing or seeing Morse Code. The most common universal distress call is “SOS”. Originally a short call for “Save Our Ship” can be translated “..._ _ _...”

After uploading the sketch and opening the serial monitor, type text in the top box and hit “SEND”. This action will send data from the computer to the Arduino to be translated into Morse Code.

DUINOKit.com
<table>
<thead>
<tr>
<th>A</th>
<th>J</th>
<th>S</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>K</td>
<td>T</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>L</td>
<td>U</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>M</td>
<td>V</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>N</td>
<td>W</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>O</td>
<td>X</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>P</td>
<td>Y</td>
<td>7</td>
</tr>
<tr>
<td>H</td>
<td>Q</td>
<td>Z</td>
<td>8</td>
</tr>
<tr>
<td>I</td>
<td>R</td>
<td>0</td>
<td>9</td>
</tr>
</tbody>
</table>

There are different codes for punctuation and spaces between words is represented by a brief pause. Search “Morse Code” or “CW Code” for more information.

Although the sub-missions listed below may be difficult at this time, please come back and try to get them figured out. You can also search for helpful information online and/or post solutions and questions on the www.DuinoKit.com/ShowAndTell message forum.

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

- [ ] Can you get the serial monitor to display the message you just sent?
- [ ] Learn to recognize and write your name in Morse code.
- [ ] Is there an Arduino device that will allow you to encode and decode Morse?
MISSION #14 - ROTARY ENCODER

Have you seen the volume knobs that spin indefinitely? They have been appearing a lot lately in car radio systems for volume and tuning. This nice little device is called a Rotary Encoder and there is one located in the DuinoKit. We are going to take a look at some basic code to display values in the serial monitor and also use the “map” function for this mission.

The hookup of the rotary encoder only requires 3 wires. For this little example we will not be using the switch which is activated by pressing down on the shaft of the rotary encoder. We will use this in our next mission....
After uploading a sketch, the button in the corner will open a new window where communication to/from the Arduino is possible through the USB cable.

In most, but not all of the Arduino microprocessors, opening this window will reset the sketch and run from the beginning.

The sketch for this mission activates the serial monitor display.

Since this is mainly an example of how the rotary encoder works, we have not hooked in any additional components at this time. We will use this module in the next mission “Guess a Number”.

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

- [ ] Change the code to receive “mapped” values from -10 to 10
- [ ] Can you change code so that it will only display values from -10 to 10? (If the encoder is turned further, do not keep counting.)
- [ ] Add in code so that the switch (pressing the shaft down) will display text some additional text or light an LED.
MISSION #15 - GUESS A NUMBER

In this game, the computer will pick a number between 1 and 100 and you will try to guess it using the rotary encoder as an input device. Turn the shaft on the rotary encoder, when you have selected your number, press the shaft down to find out if your guess was too high or too low.

As you may have noticed in Mission #14, the rotary encoder can be a tricky module to work with. Feel free to edit this code to see how it changes the behaviour of the program.
The sketch for this mission activates both the serial monitor and the LCD display if they are hooked up. The wiring instructions are in the sketch itself to hook up the LCD.

The sketch demonstrates some of the challenges with using a rotary encoder. It is nice to have a free spinning module, however accurate input depends on coding, speed of rotation and also may require additional components for precise data input.

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

☐ Can you edit the code to pick a number between 1 and 50?

☐ Using code, can you eliminate the values that are out of range and have the counter only display values 1-100?

☐ Wire in the potentiometer and a button as an input device for your number guess. (Hint... See Mission #5 for potentiometer hookup ideas)
MISSION #16 – LIGHT CHANGE ALARM

One of the best detections that is difficult to overcome is the change in light. For this mission we set up a simple module (CDS Cell) to monitor the existing light level in a room and if it varies by a small amount, sound the buzzer as an alarm. This could be set up using a laser pointer or a flash light aimed at the CDS Cell and when the light beam is broken, the alarm will sound.

One problem we had to address it that light levels gradually change throughout the day as the sun moves or as or room lighting is adjusted. We could not set an absolute threshold value for the alarm, but made it to adjust throughout the day. Therefore gradual changes will not sound the alarm, only sudden “drastic” changes in light level will trigger the alarm event.

In a more advanced project this device could trigger a camera to take a picture or maybe an email or text message. Although these are beyond the scope and components of the DuinoKit, the idea of interfacing this simple alarm device is something to get you mind thinking about other ideas.
After uploading a sketch, the button in the corner will open a new window where communication to/from the Arduino is possible through the USB cable.

In most, but not all of the Arduino microprocessors, opening this window will reset the sketch and run from the beginning.

The sketch for this mission uses the serial monitor to display current and alarm light level readings.

Each light level reading displays in the serial monitor as the average value of 5 consecutive readings. This was done to filter out spikes that may be present with just one quick reading.

**YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!**

- ☐ Edit the code to make the device more and less sensitive
- ☐ Do you think this alarm is more or less effective than a distance or motion alarm? Defend your answer!
- ☐ Can you rewire this mission to detect quick movement of the potentiometer but not a slow adjustment?
MISSION #17 - INFRARED CODE DETECTOR

Infrared is a type of light that is not visible to the human eye. It is often used in TV remote controls for adjusting the channel or volume on the television. Some remote controls, IR may be barely visible, however you may see the IR LED show up better while pressing a button and looking through a digital camera of phone camera.

If you press a button on your remote while looking thorough a digital camera or video camera, you may see the IR LED light up. The interesting thing about IR technology is not really the fact that the IR lights up, but it is flashing very fast and actually transmitting a code via IR light. It is kind of like a high speed Morse code that we experimented with in Mission #13.

In this mission, we will be looking at these cryptic codes sent through the air via IR light. Once we learn how to read and send these codes, you now have a great near field communication system to transmitting and receiving data from one device to another.
After uploading a sketch, the button in the corner will open a new window where communication to/from the Arduino is possible through the USB cable.

If you do not already have the IR library installed, you may have to install this through the Arduino IDE. We did include the IR library file in the download directory on the Show and Tell section of www.DuinoKit.com website. If you download and open this file, you will notice the IRremote.h file along with the program file. When opening the sketch, this will automatically open TWO windows in the Arduino IDE. With this option, the IRremote library file does not need to be installed, however it is a good library to have on hand.

**YOUR MISSION…. SHOULD YOU CHOOSE TO ACCEPT IT!**

- Edit the code and wiring so that pressing any IR button will toggle a LED on / off.
- Edit the code so the a specific button will turn on and LED and a specific button press will turn the same LED off.
- Can you edit the code so that a specific series of button presses will turn an LED on and off. Think if this as sending a digital combination.
MISSION #18 - INFRARED LIGHT CONTROL

In the previous mission you learn how to view various IR codes sent from a IR remote control. In this mission we are going to actually “learn” several codes from the remote control and then use these codes to change the behaviour of an LED. The only addition we made was to hook an LED

Why in this wiring diagram does the (-) or GND pin of the LED NOT have to be directly alongside the (+) for the LED we are trying to use for this project?

Notice how pin D9 on the Arduino has an * next to it indicating “PWM”. Why is the PWM pin important for the operation of the LED in this mission?
After uploading a sketch, the button in the corner will open a new window where communication to/from the Arduino is possible through the USB cable.

Be sure to open the Serial Monitor when working with this program, it will prompt you how to learn the codes and about program operation. This is a handy feature to build into sketches as you write and debug their functionality.

If you do not already have the IR library installed, may have to install this through the Arduino IDE. We did not include the IR library file in the download directory so it will need to be installed for this mission to compile correctly.

**YOUR MISSION…. SHOULD YOU CHOOSE TO ACCEPT IT!?**

- □ Change the code and wiring to a non PWM pin and see how this program will operate.
- □ Can you edit the code to “learn” 8 or 10 IR key presses?
- □ Instead of changing the brightness of the LED, can you turn 5 different LED on and off?
MISSION #19 - FORTUNE TELLER

A great benefit in life is the ability to know or predict what is going to happen next or having a “sure fire” way to making a decision. For this mission we are going to create a device that will help you in these situations. Keep in mind, this in only for entertainment purposes. :)

When you press the button in this project, the LCD screen will randomly choose text from the given code and display it in the screen.
Originally this project was written to choose from a list of names in a teacher classroom. The students felt like the teacher would call on them and they were being “picked on” by the teacher during class. This device solved the problem because with the press of a button, a student was randomly chosen during class. Even after this device was introduced, the students then thought the “Pick-a-Kid” device “knew” who to call on.

This Fortune teller then needed to be upgraded to accommodated 4 different classes. See your “Mission Possible” below.

This project could serve as digital dice or playing cards for a board game. Maybe different buttons for different types of moves during the game.

YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!

☐ Add in a button to pick random numbers 1-10
☐ Attach all four buttons, each selecting data from 4 different lists.
☐ Attach the battery to go portable and have fun asking question with your friends.
MISSION #20 - REACTION TIMER

Now that you are near the end of your first set of missions, it is now time for a test. Being a good secret agent requires deductive reasoning, questioning and quick reactions. For this mission we are going to test your reaction time to various lights. After 10 reaction tests, you will be given your average score in 1/1000 of a second on the LCD display.

See back side of this mission card displays the LCD wiring diagram used for keeping track of your score.

Does the wiring for this project look at all familiar to something we have built before?
This is a continuation of wiring for the reaction tester. This is the same wiring as mission #10 ("Simon" memory game) except that we removed the buzzer and wired the buttons in the correct sequence.

The LCD wiring in this project was changed from the typical hookup pins to Analogue pins. Can you find the line in the code where these changes are made to correspond to this wiring diagram?

**YOUR MISSION...SHOULD YOU CHOOSE TO ACCEPT IT!**

- □ Hook up this project to run on battery power.
- □ Test your reaction time against someone else. Remember, the LOWEST time is the fastest.
- □ Look through code and see if you can make the buzzer sound if you press the wrong button during a test.
ONGOING MISSION - SHOW AND TELL

During the previous missions, you have worked with many components and really only started to get ideas of some of the things possible. This is your opportunity to collaborate on the DuinoKit “Show and Tell” forum and share some projects or comments on some of your missions.

DuinoKit.com, sign up for an account and go to the Show and Tell forum.
If you are new to the Show and Tell forum, we will have to activate your account to allow for posting and replying to topics in the forum. If you directly purchased from DuinoKit.com, your account is already activated with post/reply privileges.

If you are new to the forum, please create an account on the DuinoKit.com store front and then send an email to Activate@DuinoKit.com and include your email address and we can then activate your account with post/reply privileges.

The only way we can learn and grow is to share with others. Please join the DuinoKit.com community and share what you have learned with others as we keep making and learning.

Sharing with others is the best way to help out others and also learn as you keep building.

YOUR MISSION.... SHOULD YOU CHOOSE TO ACCEPT IT!

☐ Post/reply to at least one topic.

☐ Find another lesson online or something that you have learned and share!

☐ Develop a new “mission” and share it with code on the forum and email to: Missions@DuinoKit.com. We may be able to include your mission on a future mission card.