Looping and testing conditions

In most applications, microcontrollers are running a program with an infinite loop that:

1) Checks the conditions on one or more pins that have been designated as inputs
2) Based on those conditions, the program changes (or doesn't change) the state of pins designated for output
3) Goes back through the loop.

Example program:

```
 #include <LiquidCrystal.h>

int switchPin = 5;
int pinVal;

void setup() {
  pinMode(switchPin, INPUT);
}

void loop(){
 lcd.clear();
 pinVal = digitalRead(switchPin);
 if (pinVal == LOW) {
    lcd.print("I'm on.");
  } else {
    lcd.print("I'm off.");
  }
}
```

Compiler directives

Global variables

Initialization section ("setup")

Infinite loop

Digital I/O

The setting of pins for either output or input is done with a "pinMode(num, OUTPUT)"; or "pinMode(num, INPUT);" statement. For example, to set digital pin 7 as an output, the statement is:

```
pinMode(7, OUTPUT);
```

To set digital pin 7 as an input, the statement is:

```
pinMode(7, INPUT);
```

To enable the internal resistor to make it HIGH by default: `pinMode(7, INPUT_PULLUP);`

The pin assignments only need to be done once and should be set up at the beginning of the program (not as part of the infinite loop) since it would be a waste of computational cycles to continually re-assign them.

The process of storing the state of a pin ("reading") is done using the "digitalRead()" statement. "Setting" the state of a pin ("writing") as 0 or 5 volts is done using the "digitalWrite()" statement.

Example: If pin 7 has been set as an output, the following would set the voltage between pin 7 and Ground to be 0:

```
digitalWrite(7, LOW);
```

Conversely, the following statement would set the voltage between pin 7 and Ground to be 5 volts:

```
digitalWrite(7, HIGH);
```

Digital example: If digital pin 7 has been defined in as an input (with "`pinMode(7,INPUT);`"), this statement would read the state of the pin and store it (as a 1 or 0) in a variable named "s_in":

```
s_in = digitalRead(7);
```

-And the following code would read the state the pin and if it is high, will change the argument to "lcd.print()":

```
if (s_in == HIGH) {
  lcd.print("I'm off.");
 }
```

For the digital ports on the chip, these are *boolean* operations. When there's 5 volts between a pin and ground, it can be referred to as "on" or "high" or "true". When the voltage between a pin and ground is zero, it can be referred to as "off" or "low" or "false".

Analog I/O

The “analog” pins A0 thru A5 can be used to read a variable (from 0-5) voltage.

The resolution of the A-to-D converter is 10 bits (0-1023). If you want to store values in one byte (8 bits), you can use the “map()” function. It has 5 arguments: the variable storing the input, the input range and the output range:

```
map(A_input, 0, 1023, 0, 255);
```

Variables that will be accessed by different functions in your program are called “global variables” and need to be defined at the beginning of your code, before setup(), outside of any function. Here is a definition of a variable called “a_in” that will be used to store integers (whole numbers):

```
int a_in;
```

“Reading” the value of an analog input is generally done by an assignment statement that stores the value of the analog input in a variable. This example looks at the voltage on pin A5 and stores it as a number in a variable called “a_in”. The default range of the number is 0 to 1023. The second line of code scales this range, “re-maps it” to a range between 0 and 255, and then stores it back in the “a_in” variable.

```
a_in = digitalRead(A5);
a_in = map(a_in, 0, 1023, 0, 255);
```

Conditional operators:

if - else

Probably the most common conditional operator is the “if” test. Syntax: the keyword “if” is followed by a boolean condition in parentheses, and this is followed by semi-colon-terminated statements (enclosed in curly brackets) to perform should the boolean test return 1, or “TRUE”.

```
if (a_in > 2) {
   lcd.print(“Thank you, Mom.”);
  }
```

The “if” test can be extended with the keyword “else” to specify statements to be performed if the boolean operation in the “if” test returns 0 or “FALSE”:

```
if (c_in == 0) {
  lcd.print(“Switch is open.”);
} else {
  lcd.print(“Switch is closed.”);
}
```

“else if” can be used in situations when you need to test for more than just 2 conditions:

```
if (a_in <= 3){
  lcd.print(“cold”);
 } else if(a_in > 3 && scaled <= 6){
  lcd.print(“medium”);
 } else if(a_in > 6){
  lcd.print(“hot”);
 }
```

Boolean operations

```
(a == 1)          // true if a is equal to 1
(a != 1)          // true if a is NOT equal to 1
(a <= 1)          // true if a is less than or equal to 1
(a >= 1)            // true if a is greater than or equal to 1
(a < 10 && a > 5)       // true if a < 10 AND a > 5
(b < 10 || b > 5)       // true if b < 10 OR b > 5
((a > 5) && (b != 10))  // true if a is greater than 5 and
                        // b is not equal to 10
```