NEXTFAB

- Programming Basics: If/Else statements provide developer with options

```
 if (condition 1)
    {
       do something;
    }
else if (condition 2)
    {
       do something else;
    }
else
    {
       default something;
    }
```

- Programming Basics: Loops take a single command and repeat until a condition isn't true anymore, an interval is completed, or an interrupt occurs.
    - For loops - start from a point and iterate a variable over an interval

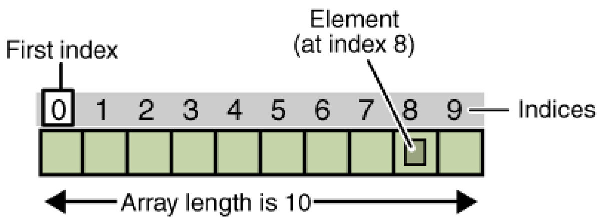
```
for( i = starting condition; until i = end condition; iterate)
  {
    do something cool to i;
   }
```

    - While and Do loops - check to see if a condition is true then loop when it is

```
while(condition == true)
   {
     do something cool;
   }

do
   {
     something cool;
   }
while(condition == true)
```

- Arrays- Data structures with a predefined size. Each item in the array is given an index number from 0 - N where N is the 1-size of array



- Declaring an array: all items in array must be same data type

```
int numbers[8] = {0,1,2,3,4,5,6,7};
```

```
 int LLED[4] = {RED, BLUE, GRN, ORA};
float deci[5] = {3.12,4.56,6.32,7.4,8.42};
```

- Functions
  - Provide easier debugging
  - Offer modularity and reuse
  - Can take a bunch of smaller source files, test them and import them into a larger source file
- Void functions do not return a variable

```
 void functionName(parameters)
{
  body of function;
}
```

- Return functions return a variable

```
 dataType functionName(parameters)
{
  body of function;
  return value;
}
```

- Parameters: Variables/values you pass into a function, names of parameters in a function are place holders and do not need to match the variables you're passing in provided they have the same data type.
- Libraries: Collections of functions that are pre-written for ease of use, have two parts
  - Header file (.h) - private and public methods and constructors, use these files to learn more about how to use the library
  - Pre-processer file (.cpp) - private and public methods that work on the hardware level with the Arduino, use these files when creating more complex devices to make sure libraries are not interfering with one another.
- Using libraries has two steps
- Call library: `#include <libraryName.h>`
- Create library object: `libraryName objectName(define parameters);`
  - Keep object names simple. Every time you call a function from a library you'll be typing it out
  - Example:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
//object name is "lcd" library name is "LiquidCrystal"
```

- Using a function from the library

```
lcd.print("hello, world!");  //lcd screen prints "hello, world!"
```