**Structure:**

Arduino programs can be divided in three main parts: **Structure, Values** (variables and constants), and **Functions**.

Let us start with the **Structure**. Software structure consists of two main functions −

* Setup( ) function
* Loop( ) function

Void setup ( ) {

}

Void Loop ( ) {

}

# Data Types:

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines.

The following data types that we use during Arduino programming

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| void | Boolean | char | Unsigned char | byte | int | Unsigned int | word |
| long | Unsigned long | short | float | double | array | String-char array | String-object |

**Variable Scope**

Variables in C programming language, which Arduino uses. Variables are types

* Inside a function or a block, which is called **local variables**.
* In the definition of function parameters, which is called **formal parameters**.
* Outside of all functions, which is called **global variables**.

# Operators:

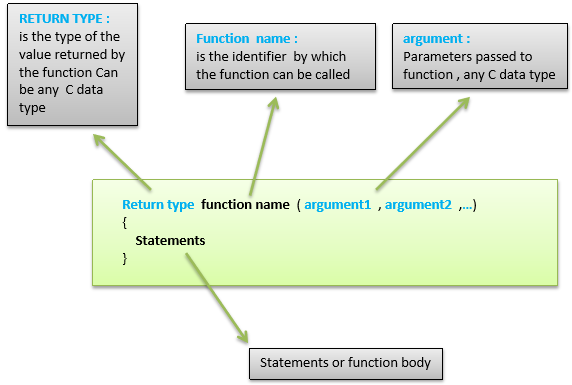
An operator is a symbol that tells the compiler to perform specific mathematical or logical functions. C language is rich in built-in operators and provides the following types of operators −

* Arithmetic Operators
* Comparison Operators
* Boolean Operators
* Bitwise Operators
* Compound Operators

# Functions:

Functions allow structuring the programs in segments of code to perform individual tasks.

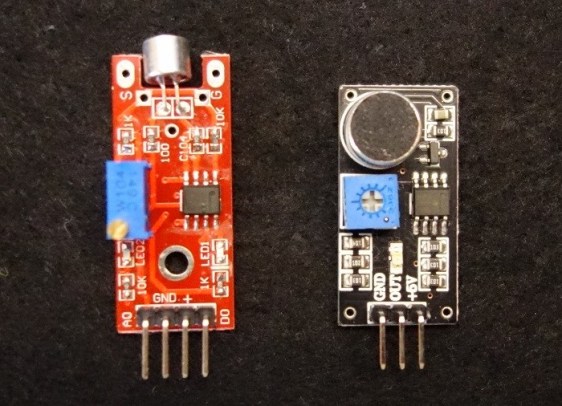
The most common syntax to define a function is −



**Microphone Sound Sensor**

The microphone sound sensor, as the name says, detects sound. It gives a measurement of how loud a sound is.

There are a wide variety of these sensors.  In the figure below you can see the most common used with the Arduino.

****

At the leftmost side, you can see the KY-038 and at the right the LM393 microphone sound sensor.

Both sensor modules have a built-in potentiometer to adjust the sensitivity of the digital output pin.

**Pin wiring**

Wiring your sensor to the Arduino is pretty straightforward:

|  |  |
| --- | --- |
| **Pin** | **Wiring to Arduino** |
| A0 | Analog pins |
| D0 | Digital pins |
| GND | GND |
| VCC | 5V |

If you’re using the LM393 module, you should connect the OUT pin to an Arduino digital pin.

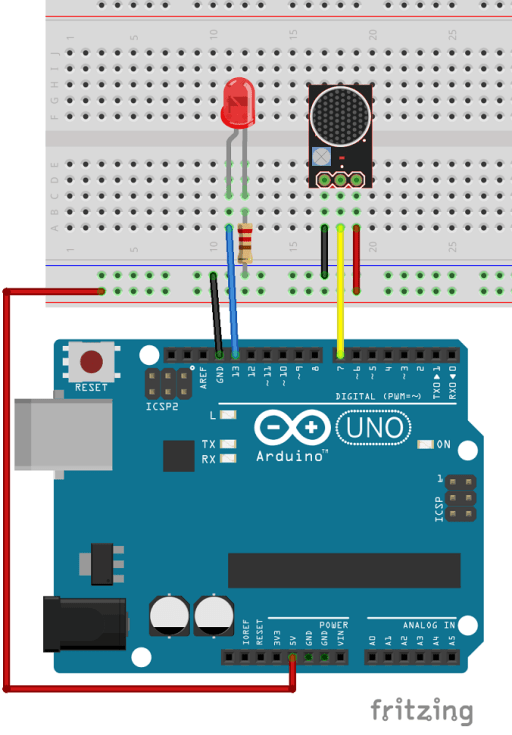
### Parts required

For this example you’ll need the following components:

* [1x Microphone sound sensor](https://makeradvisor.com/tools/microphone-sound-sensor-ky-038/)
* [Arduino UNO](https://makeradvisor.com/tools/compatible-arduino-uno-r3-board/) – read [Best Arduino Starter Kits](https://makeradvisor.com/best-arduino-starter-kits/)
* [1x Breadboard](https://makeradvisor.com/tools/mb-102-solderless-breadboard-830-points/)
* [1x LED](https://makeradvisor.com/tools/3mm-5mm-leds-kit-storage-box/)
* [1x 220 Ohm resistor](https://makeradvisor.com/tools/resistors-kits/)
* [Jumper wires](https://makeradvisor.com/tools/jumper-wires-kit-120-pieces/)

### Schematics

Assemble all the parts by following the schematics below:



### Code

Upload the following code to your Arduino board.

/\*

\* Rui Santos

\* Complete Project Details http://randomnerdtutorials.com

\*/

int ledPin=13;

int sensorPin=7;

boolean val =0;

void setup(){

pinMode(ledPin, OUTPUT);

pinMode(sensorPin, INPUT);

Serial.begin (9600);

}

void loop (){

val =digitalRead(sensorPin);

Serial.println (val);

// when the sensor detects a signal above the threshold value, LED flashes

if (val==HIGH) {

digitalWrite(ledPin, HIGH);

}

else {

digitalWrite(ledPin, LOW);

}

}