

FREE

IEEE - CSULB Branch
April 9 2021
Kiyo Terao
John Abraham

Agenda

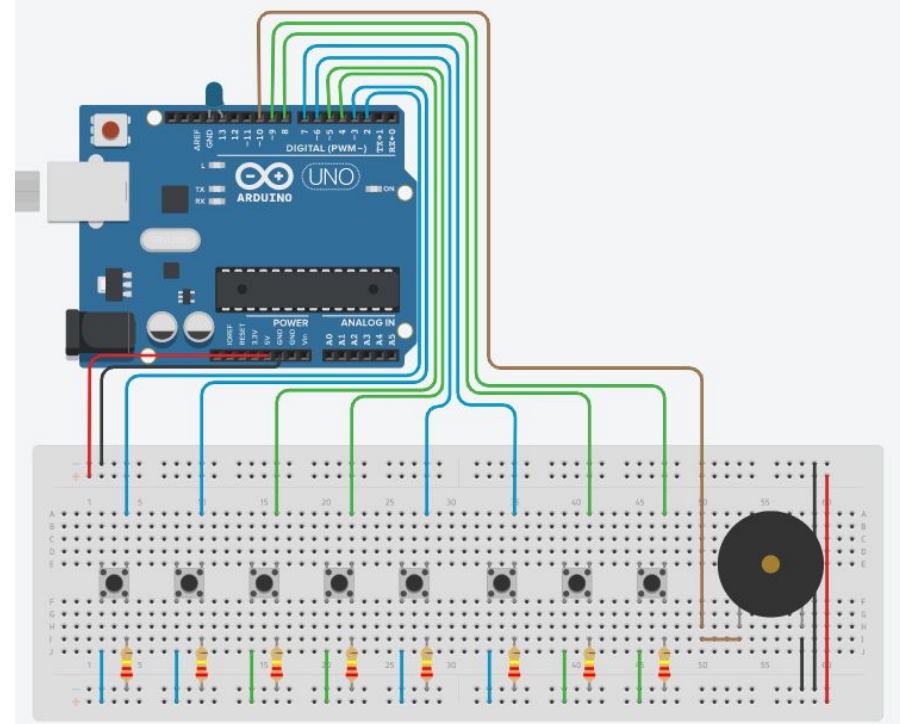
- Project
- Software setup
 - Arduino IDE
- Arduino Programming Language
 - Functions
 - Values
 - Structures

FREE

Code Link:

Project- Arduino Piano

- Sound Reactive Arduino Floor Piano
- Modules used
 - Arduino Uno/nano
- Components
 - Resistors
 - Jumper wires
 - Piezo



Software Setup

- ▶ Google “Arduino IDE”
 - Click on the first link ([Software | Arduino](#))
 - Scroll down and look for “Arduino IDE 1.8.13 “
 - Click on your respective Operating System (i.e. Windows, Apple) to download file.



Arduino IDE 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer

Windows ZIP file

Windows app Win 8.1 or 10 

Linux 32 bits

Linux 64 bits

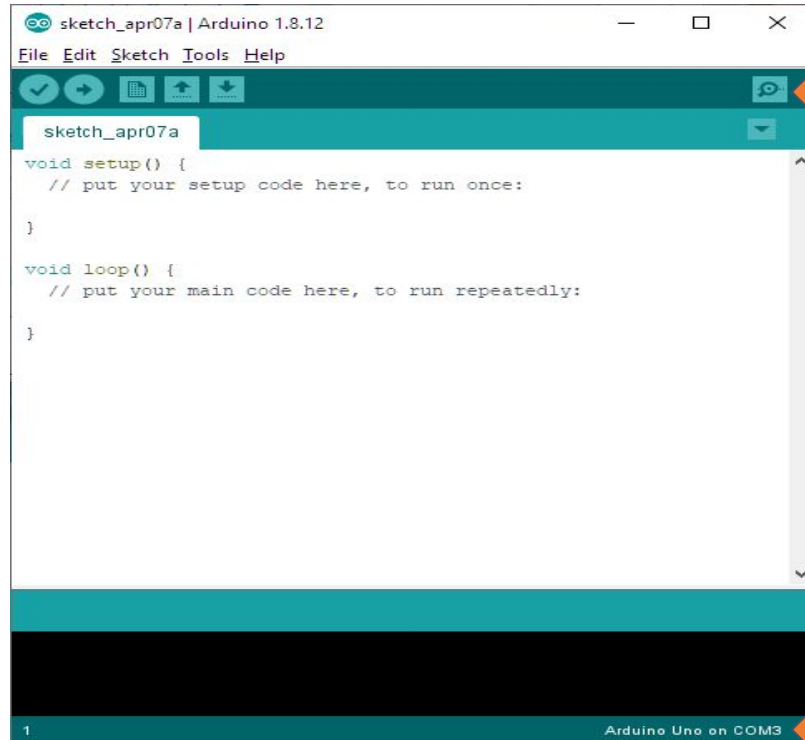
Linux ARM 32 bits

Linux ARM 64 bits

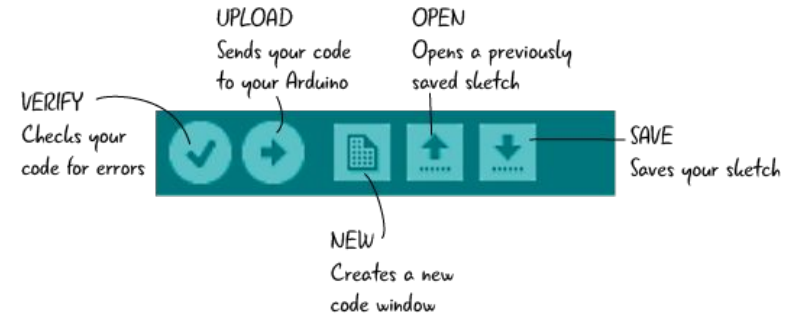
Mac OS X 10.10 or newer

[Release Notes](#) [Checksums](#) (sha512)

Software Setup - continued



Serial Monitor



Arduino board

Arduino Programming Language -Functions

- ▶ Functions
 - Controls arduino board and performs multiple computations/tasks
 - Help the programmers stay organized
- ▶ Common functions
 - pinMode()
 - digitalRead()
 - digitalWrite()
- ▶ Advanced
 - tone()

Anatomy of a C function

Datatype of data returned, any C datatype.

Parameters passed to function, any C datatype.

"void" if nothing is returned.

Function name

```
int myMultiplyFunction(int x, int y){  
    int result;  
    result = x * y;  
    return result;  
}
```

Return statement, datatype matches declaration.

Curly braces required.

RETURN TYPE :
is the type of the value returned by the function Can be any C data type

Function name :
is the identifier by which the function can be called

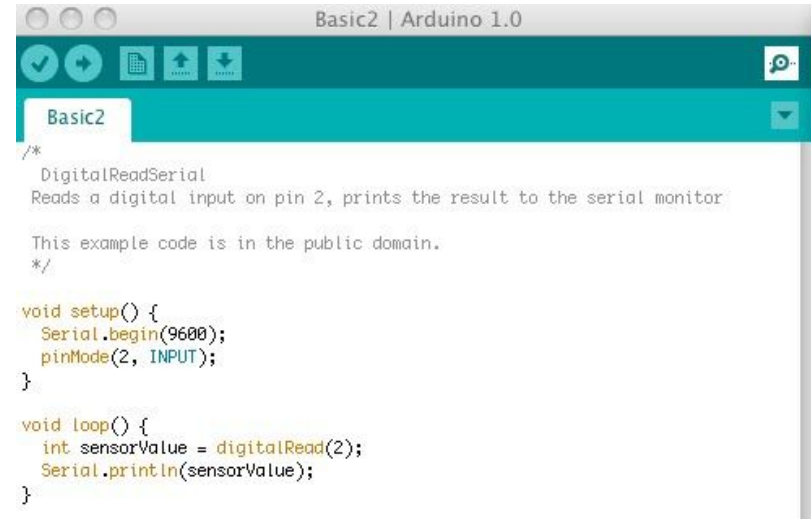
argument :
Parameters passed to function , any C data type

```
Return type function name ( argument1 , argument2 , ... )  
{  
    Statements  
}
```

Statements or function body

Functions > Digital I/O > pinMode()

- ▶ What is “pinMode()”
 - Configure the pins to act as an input or an output
 - Inside of the “void setup()”
- ▶ Syntax
 - pinMode(pin,mode)
 - Pin
- ▶ Parameters
 - pin:arduino pin number
- ▶ Example
 - ```
void setup() {
 pinMode(cNote,INPUT); //
 sets the digital pin 2 = cNote as
 input
}
```



The screenshot shows the Arduino IDE interface with a sketch named "Basic2". The code in the editor is as follows:

```
/*
 DigitalReadSerial
 Reads a digital input on pin 2, prints the result to the serial monitor

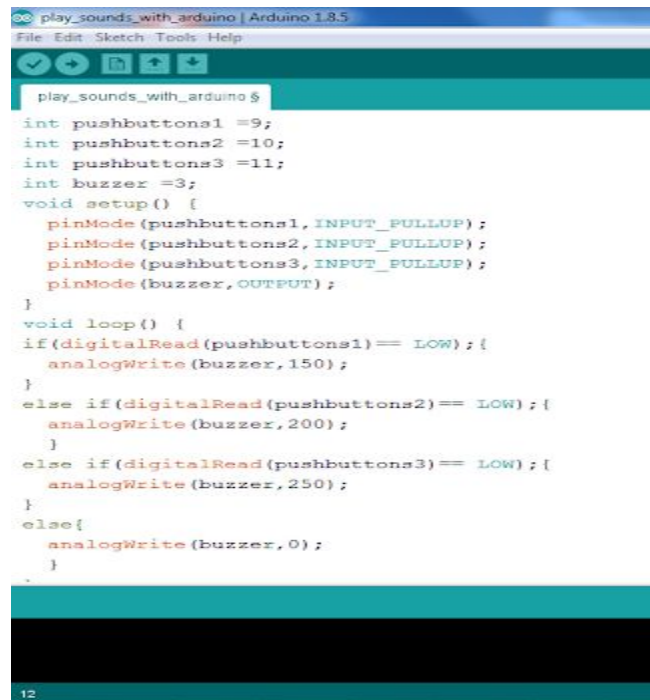
 This example code is in the public domain.
 */

void setup() {
 Serial.begin(9600);
 pinMode(2, INPUT);
}

void loop() {
 int sensorValue = digitalRead(2);
 Serial.println(sensorValue);
}
```

# Functions > Digital I/O > digitalWrite()

- ▶ What is “digitalRead()”
  - Reads the value from a specified digital pin
    - HIGH or LOW
  - Inside of the loop
- ▶ Syntax
  - digitalRead(pin)
- ▶ Parameter
  - pin : pin number you want to read
- ▶ Example
  - ```
void loop() {  
    val = digitalRead(inPin); // read  
    the input pin  
    digitalWrite(ledPin, val); // sets  
    the LED to the button's value  
}
```

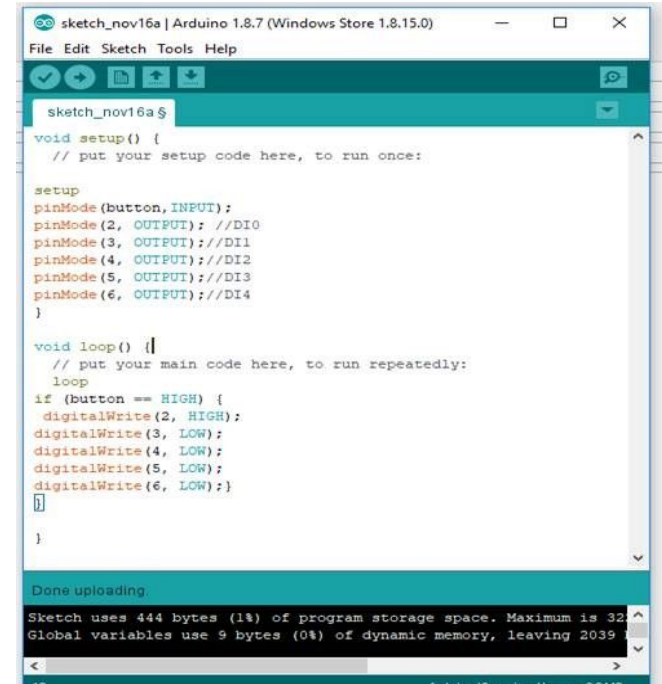


```
play_sounds_with_arduino | Arduino 1.8.5
File Edit Sketch Tools Help

play_sounds_with_arduino $
int pushbuttons1 =9;
int pushbuttons2 =10;
int pushbuttons3 =11;
int buzzer =3;
void setup() {
    pinMode(pushbuttons1, INPUT_PULLUP);
    pinMode(pushbuttons2, INPUT_PULLUP);
    pinMode(pushbuttons3, INPUT_PULLUP);
    pinMode(buzzer, OUTPUT);
}
void loop() {
    if(digitalRead(pushbuttons1)== LOW){
        analogWrite(buzzer,150);
    }
    else if(digitalRead(pushbuttons2)== LOW){
        analogWrite(buzzer,200);
    }
    else if(digitalRead(pushbuttons3)== LOW){
        analogWrite(buzzer,250);
    }
    else{
        analogWrite(buzzer, 0);
    }
}
```


Function > digital I/O > digitalWrite()

- ▶ What is “digitalWrite()”
 - Used to write a HIGH or a LOW value
 - If OUTPUT with pinMode()
 - Voltage will be set to 5V/3.3V for HIGH and 0V for LOW
- ▶ Syntax
 - digitalWrite(pin, value) //value HIGH/LOW
- ▶ Parameter
 -
- ▶ Example
 - ```
void loop() {
 val = digitalRead(inPin); // read
 the input pin
 digitalWrite(ledPin, val); // sets
 the LED to the button's value
}
```



The screenshot shows the Arduino IDE interface. The title bar reads "sketch\_nov16a | Arduino 1.8.7 (Windows Store 1.8.15.0)". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains icons for opening files, saving, compiling, and uploading. The main text area shows the following code:

```
sketch_nov16a $

void setup() {
 // put your setup code here, to run once:

 setup
 pinMode(button, INPUT);
 pinMode(2, OUTPUT); //DI0
 pinMode(3, OUTPUT); //DI1
 pinMode(4, OUTPUT); //DI2
 pinMode(5, OUTPUT); //DI3
 pinMode(6, OUTPUT); //DI4
}

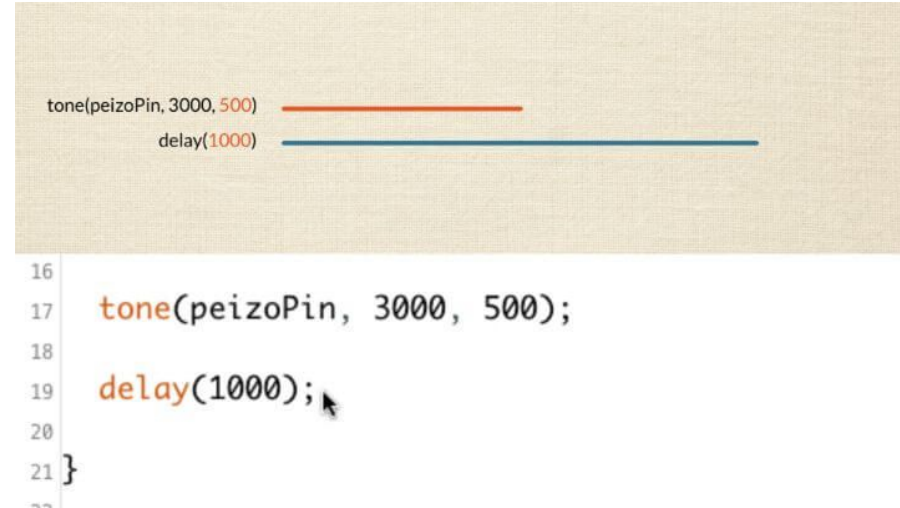
void loop() {
 // put your main code here, to run repeatedly:
 loop
 if (button == HIGH) {
 digitalWrite(2, HIGH);
 digitalWrite(3, LOW);
 digitalWrite(4, LOW);
 digitalWrite(5, LOW);
 digitalWrite(6, LOW);
 }
}
```

At the bottom, a status bar indicates "Done uploading." and provides memory usage information: "Sketch uses 444 bytes (1%) of program storage space. Maximum is 32256 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes free."

Example 2

# Function > Advance I/O > tone()

- ▶ What is tone()
  - Generates a square wave of the specified frequency on a pin
- ▶ Syntax
  - `tone(pin, frequency, duration)`
- ▶ Parameters
  - frequency of the tone in hertz
  - pin: to generate the tone
- ▶ Example
  - `tone(Piezo, c, 250); // plays note c`  
`digitalWrite(LED, HIGH);`

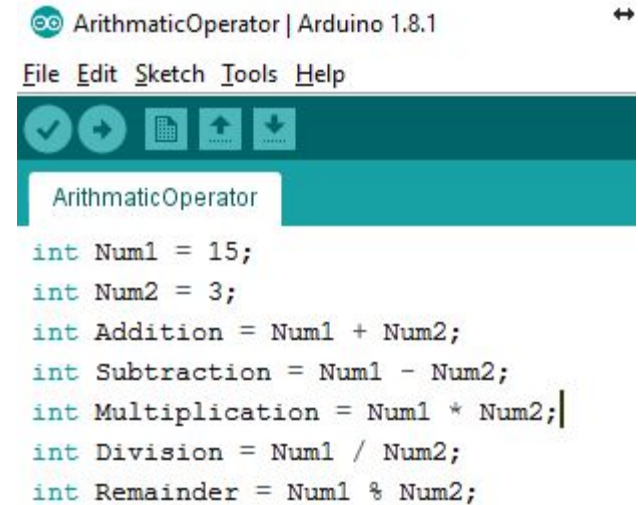


The diagram illustrates the execution of the `tone()` function. It shows a call to `tone(peizoPin, 3000, 500)` followed by a `delay(1000)` call. The `tone()` call is represented by a horizontal orange line, and the `delay()` call is represented by a horizontal blue line. Below the diagram, the corresponding code is shown in a text editor with line numbers 16 through 21.

```
16
17 tone(peizoPin, 3000, 500);
18
19 delay(1000);
20
21 }
```

# Arduino Programming Language - Variables(values & const)

- ▶ Variables (values & constants)
  - Way of naming and storing numerical values
  - A variable needs to be declared Data types and constants



ArithmeticOperator | Arduino 1.8.1

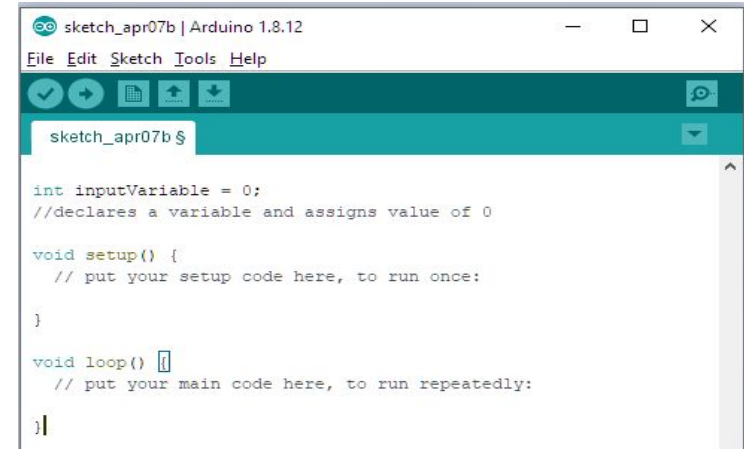
File Edit Sketch Tools Help

ArithmeticOperator

```
int Num1 = 15;
int Num2 = 3;
int Addition = Num1 + Num2;
int Subtraction = Num1 - Num2;
int Multiplication = Num1 * Num2;
int Division = Num1 / Num2;
int Remainder = Num1 % Num2;
```

# Variable > Data type > int

- ▶ What is “int”
  - Integers are primary datatype for storing numbers
  - Whole numbers
- ▶ Syntax
  - `Int var = val`
- ▶ Parameters
  - Var: variable name
  - Val: value assigned to the var
- ▶ Example code
  - `int cNote = 264; //frequency of key notes`  
`//declaring the value of the keynote`



```
sketch_apr07b | Arduino 1.8.12
File Edit Sketch Tools Help

sketch_apr07b $

int inputVariable = 0;
//declares a variable and assigns value of 0

void setup() {
 // put your setup code here, to run once:
}

void loop() {
 // put your main code here, to run repeatedly:
}
```

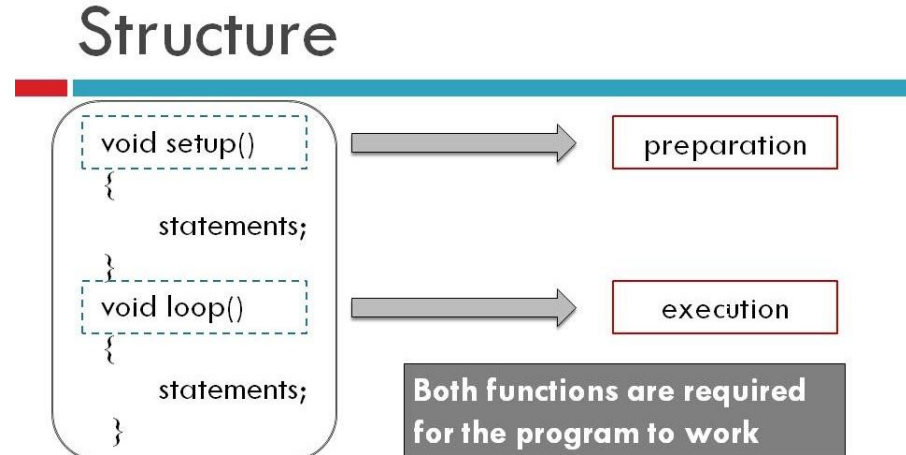
Example 2

# Variable > Data types > char

- ▶ What is char
  - Used to store character
    - Stored as numbers
  - Written in single quotes
    - 'A'
- ▶ Syntax
  - `Char var = val;`
- ▶ Parameter
  - Var:variable name
  - Val:value to assign variable
- ▶ Example
  - `char myChar = 'A';`  
`char myChar = 65; // both are equivalent`

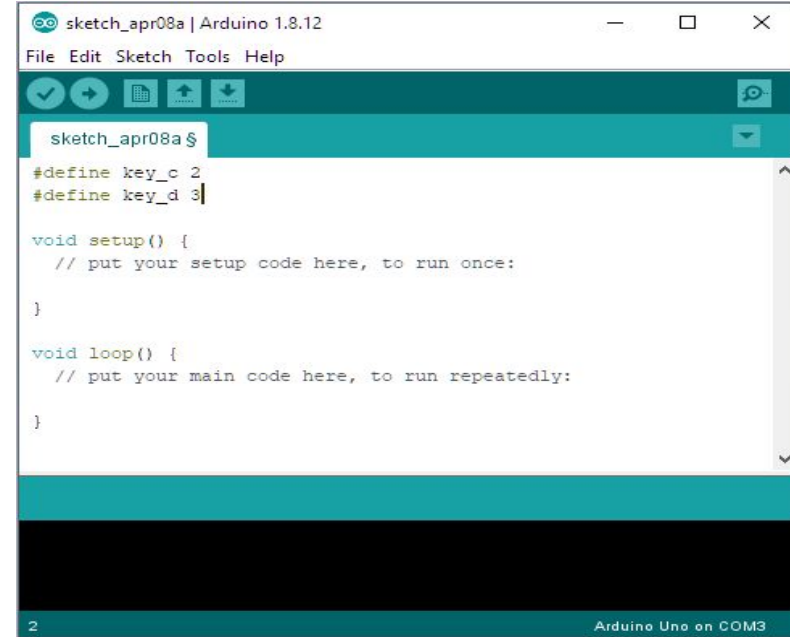
# Arduino Programming Language - Structure

- ▶ Structure
  - Elements of Arduino(C++) code
  - Allows us to combine data of different types together
  - Helps to construct a complex data type
  - Can store any type of data



# Structure > Syntax > #define

- ▶ #define
  - Allows the programmer to give a name to a constant value
- ▶ Syntax
  - #define constantName value
- ▶ Parameter
  - constantName: name of the macro to define
  - Value: value of the macro
- ▶ Example
  - `#define cNote 2`  
`// The compiler will replace any mention of cNote with the value 2 at compile time.`



```
sketch_apr08a | Arduino 1.8.12
File Edit Sketch Tools Help

sketch_apr08a $
#define key_c 2
#define key_d 3

void setup() {
 // put your setup code here, to run once:
}

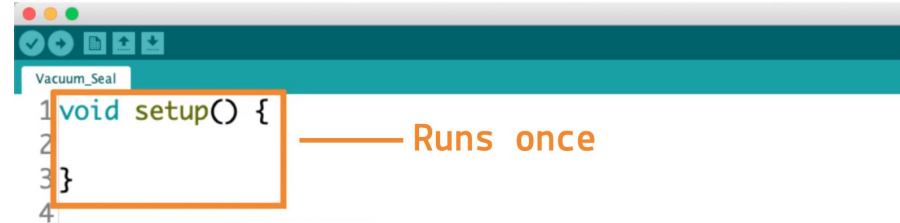
void loop() {
 // put your main code here, to run repeatedly:
}
```

2 Arduino Uno on COM3

Example

# Structure > Sketch > setup()

- ▶ What is “setup()”
  - Used for initializing
    - Variables
    - pinModes
    - libraries
  - Only run once
- ▶ Example
  - ```
void setup() { //initializing values  
    Serial.begin(9600);  
    pinMode(buttonPin, INPUT);  
}
```



```
1 void setup() {  
2  
3 }  
4
```

Runs once



```
1 void setup() {  
2  
3 /* Start Serial Communication*/  
4 Serial.begin(9600);  
5  
6 }
```

Example 2

Structure > Sketch > loop()

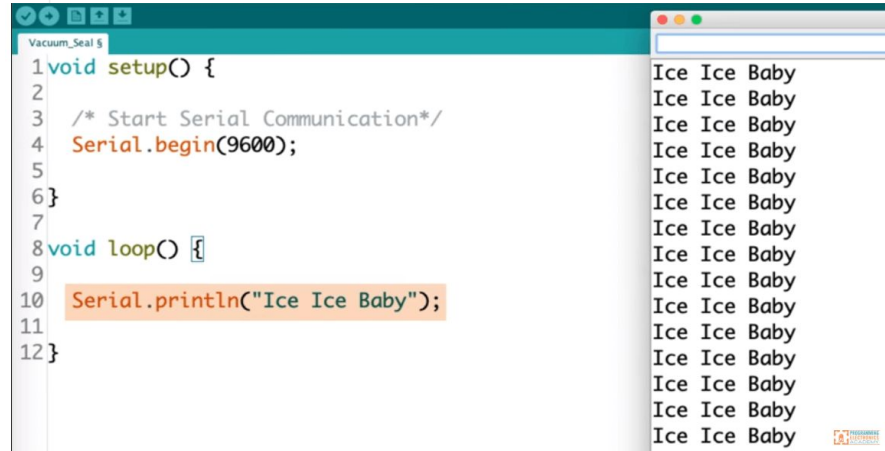
- ▶ What is “loop()”
 - A programming structure
 - Repeats a sequence of instruction until a specific condition is met
 - Allow your program to change and respond
 - Controls arduino board

- ▶ Example code (continued from “setup()”)

```
void loop() {  
    if (digitalRead(buttonPin) == HIGH)  
    {  
        Serial.write('H');  
    }  
    else {  
        Serial.write('L');  
    }  
    delay(1000); //milliseconds  
}
```

```
5 void loop() {  
6  
7     //Do first...  
8     //Do this next...  
9     //Do this too...  
10  
11 }
```

Runs over and
over and over...



The screenshot shows the Arduino IDE with a sketch named 'Vacuum_Seal 5'. The code in the editor includes a 'setup()' function that initializes serial communication at 9600 baud, and a 'loop()' function that prints 'Ice Ice Baby' to the serial monitor. The serial monitor on the right displays the output 'Ice Ice Baby' repeated 15 times, demonstrating the continuous execution of the 'loop()' function.

```
1 void setup() {  
2  
3     /* Start Serial Communication*/  
4     Serial.begin(9600);  
5  
6 }  
7  
8 void loop() {  
9  
10    Serial.println("Ice Ice Baby");  
11  
12 }
```

Structure > comparison operators > equalto

- ▶ What is ==
 - Comparison of one variable or constant against another (left and right)
- ▶ Syntax
 - `if x == y; // is true if x is equal to y and it is false if x is not equal to y`
- ▶ Parameter
 - `X:variable(int,float,double,byte,short,long)`
 - `y:variable or constant`
- ▶ Example
 - ```
if (x == y) { // tests if x is equal to y
 // do something only if the comparison
 result is true
}
```

```
int a=1;

int b=1,count=0;

if(a==b) //true

count++; //count increases
```