NYU TANDON SCHOOL OF ENGINEERING

Promoting robotic design and entrepreneurship experiences among students and teachers

Lesson 8: Basic Arduino Programming

Innovative Technology Experiences for Students and Teachers (ITEST), Professional Development Program, July 2017-19

Mechatronics, Controls, and Robotics Laboratory, Department of Mechanical and Aerospace Engineering, NYU Tandon School of Engineering

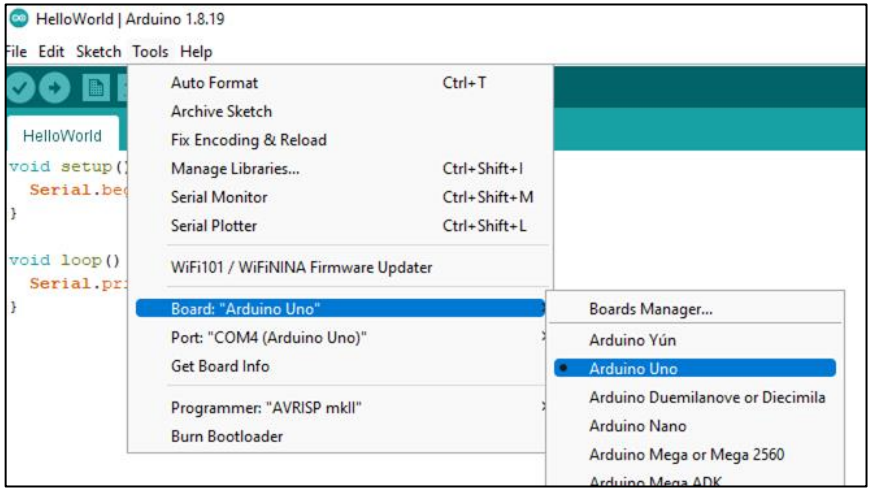
CONTENTS

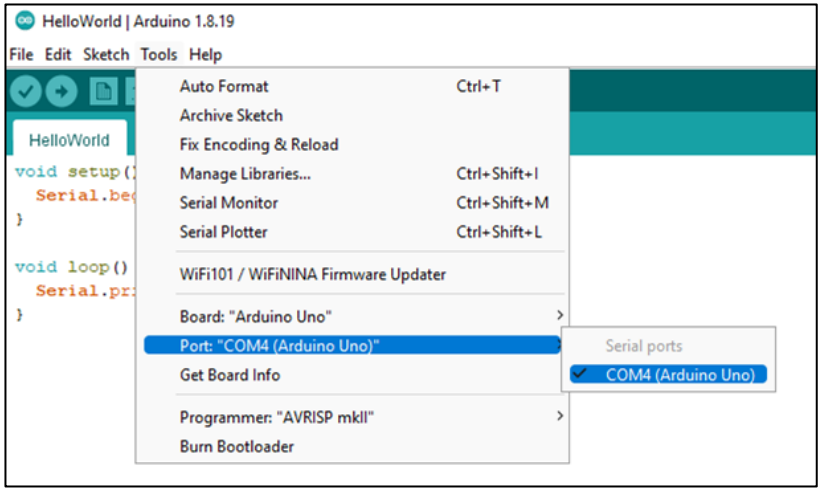
- Introduction to Arduino environment
- Hello world from Arduino
- Writing to the serial monitor
- Reading from the serial monitor
- Arithmetic operations
- Conditional operators
- Loops

- **TASK/ACTIVITY:** Arduino Hands-on session

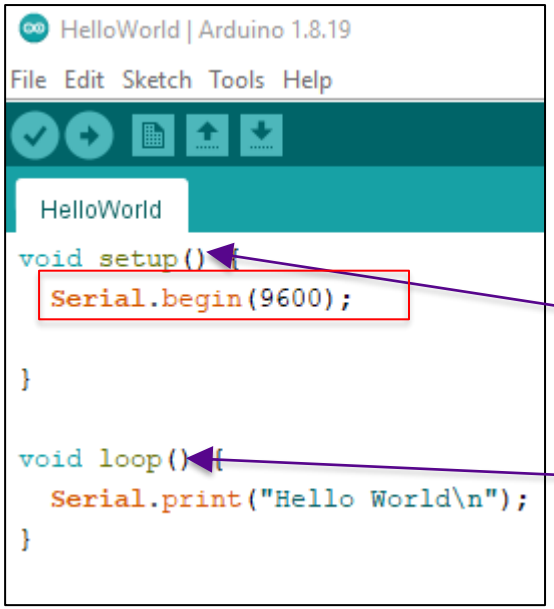
ARDUINO ENVIRONMENT

- Select the board **Arduino Uno** and the port showing in the **Serial ports** section





REFRESHER: ARDUINO SKETCH



- Intuitive programming language like C
- Code is case sensitive
- Statements are commands and must end with a semi-colon (;)
- Single line comments follow a //
- Multi-line comments begin with /* and end with */

- Void setup – code inside here runs only once during setup (configure pins, communication, interrupts, etc.)

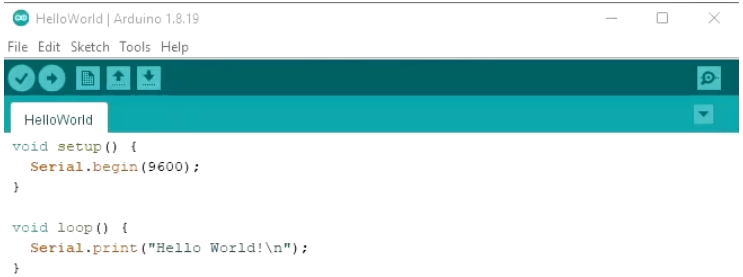
- Void loop – code inside here runs infinitely

- Serial.begin() – bit rate with which binary data is exchanged between Arduino and PC, in the figure provided, 9600 bits per second are exchanged between Arduino and a connected computing device through a USB port

HELLO WORLD

```
void setup() {
  Serial.begin(9600);
  //Open serial monitor and set baudrate to 9600
}

void loop() {
  Serial.print("Hello World\n");
  //Prints Hello World on Serial monitor repeatedly
}
```

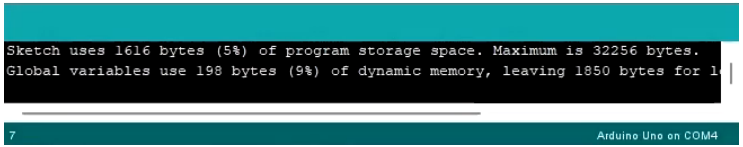
Program



- To insert in a new line

Serial.print(“Hello World**\n**"); or

Serial.println(“Hello World");



Video

NOTE: make sure that the baud rate defined in the serial monitor and Serial.begin() is the **same**

PRINTING ON SERIAL MONITOR

Serial.print():

- Prints data to the serial monitor as human-readable text

For example:

- When no output formatter has been specified, ASCII characters are printed

```
Serial.print(78);              // displays "78"
Serial.print(1.23456);         // displays "1.23"
Serial.print('N');             // displays "N"
Serial.print("Hello world.")   // displays "Hello world."
```

- When the format is mentioned, its data type is printed

```
Serial.print(78, BIN);        //displays "1001110"
Serial.print(78, OCT);        //displays "116"
Serial.print(78, DEC);        //displays "78"
Serial.print(78, HEX);        //displays "4E"
Serial.print(1.23456, 0);     //displays "1"
Serial.print(1.23456, 2);     //displays "1.23"
Serial.print(1.23456, 4);     //displays "1.2346"
```

REFRESHER: VARIABLES

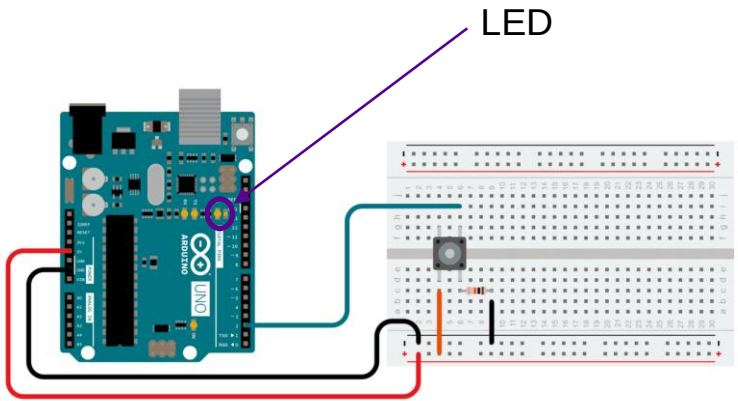
Type	Bytes	Bits	Example
boolean	1	8	Boolean led_on =True;
char	1	8	char char_1 = x;
int	2	16	int temp = 48;
float	4	32	float height = 2.5;
long	4	32	long time = 5;

REFRESHER: VARIABLE SCOPE

```
int LEDpin = 13;
int ButtonPin = 2;

void setup() {
  pinMode(LEDpin, OUTPUT);
  pinMode(ButtonPin, INPUT);
}

void loop() {
  int buttonValue = digitalRead(ButtonPin);
  digitalWrite(LEDpin, buttonValue);
}
```

Global

Local to loop()



Source

READING DATA FROM SERIAL MONITOR

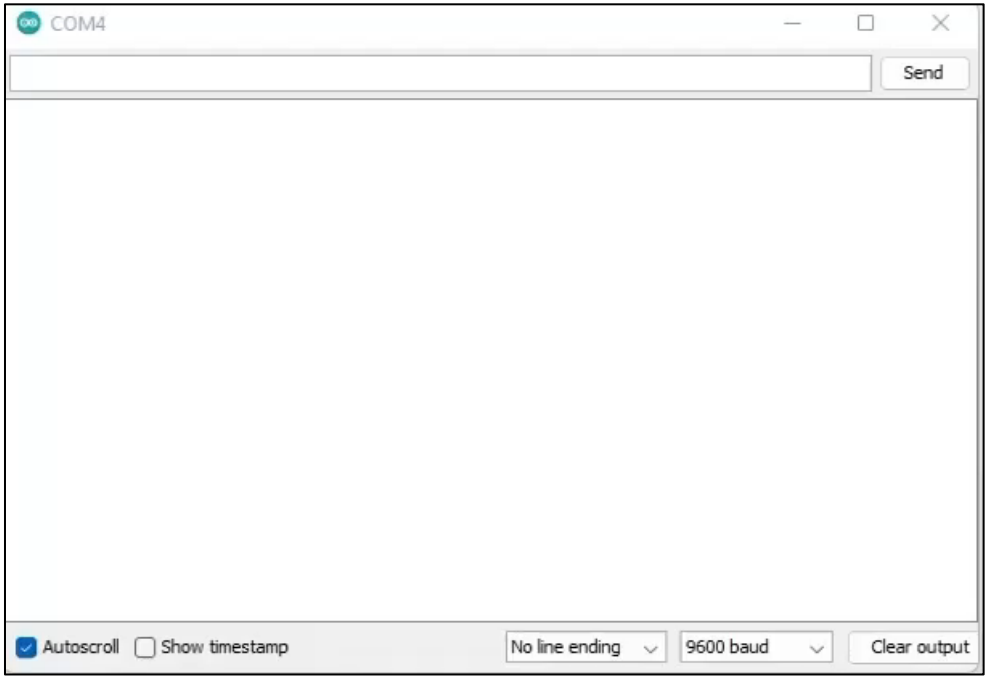
ReadFromSerialMonitor_test | Arduino 1.8.19

File Edit Sketch Tools Help

ReadFromSerialMonitor_test

```
char incomingByte;
//Setup a varaible to read input

void setup() {
  Serial.begin(9600);
  //Open serial monitor and set baudrate (bits per second) to 9600
}

void loop() {
  // enters if loop only when data is entered
  if (Serial.available() > 0) {
    incomingByte = Serial.read();
    //Read the incoming byte
    Serial.print("I received ");
    Serial.println(incomingByte);
    //Prints out entered characters
  }
}
```

Program



Video

READING DATA FROM SERIAL MONITOR

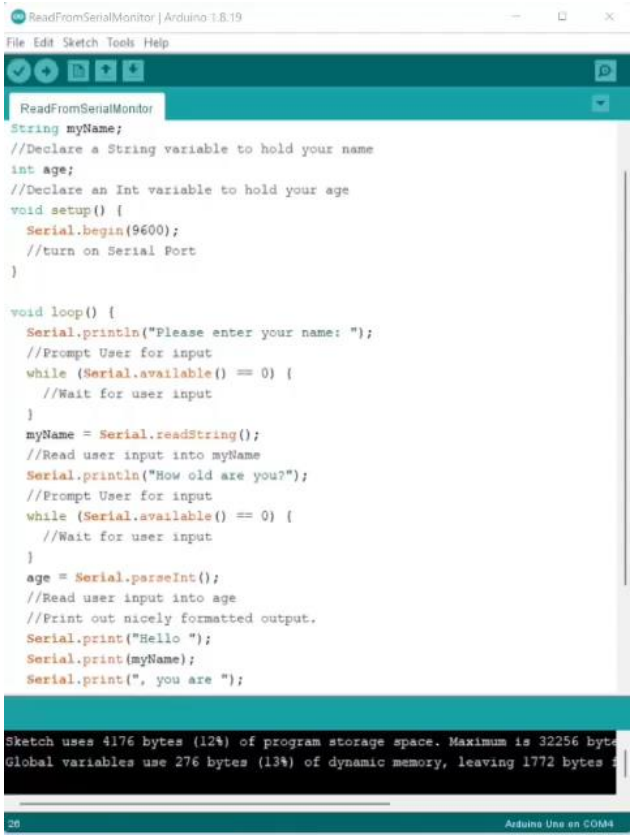
```
String myName;
//Declare a String variable to hold your name
int age;
//Declare an Int variable to hold your age

void setup() {
  Serial.begin(9600);
  //turn on Serial Port
}

void loop() {
  Serial.println("Please enter your name: ");
  //Prompt User for input
  while (Serial.available() == 0) {
    //Wait for user input
  }
  myName = Serial.readString();
  //Read user input into myName
  Serial.println("How old are you?");
  //Prompt User for input
  while (Serial.available() == 0) {
    //Wait for user input
  }
  age = Serial.parseInt();
  //Read user input into age

  //Print out nicely formatted output.
  Serial.print("Hello ");
  Serial.print(myName);
  Serial.print(", you are ");
  Serial.print(age);
  Serial.println(" years old");
  delay(5000);
}
```

Program

READING DATA FROM SERIAL MONITOR

```
String myName;
//Declare a String variable to hold your name
int age;
//Declare an Int variable to hold your age

void setup() {
  Serial.begin(9600);
  //turn on Serial Port
}

void loop() {
  Serial.println("Please enter your name: ");
  //Prompt User for input
  while (Serial.available() == 0) {
    //Wait for user input
  }
  myName = Serial.readString();
  //Read user input into myName
  Serial.println("How old are you?");

  //Prompt User for input
  while (Serial.available() == 0) {
    //Wait for user input
  }
  age = Serial.parseInt();
  //Read user input into age

  //Print out nicely formatted output.
  Serial.print("Hello ");
  Serial.print(myName);
  Serial.print(", you are ");
  Serial.print(age);
  Serial.println(" years old");
  delay(5000);
}
```

Program



Video

ARITHMETIC OPERATION - ADDITION

Working with integers

Example-1

```
int x, y; //input variables
int z; //output variable
x = 20;
y = 50;
z = x + y; // z is 70
```

Working with floating numbers

Example-2

```
float x, y; //input variables
float z; //output variable as float data type
x = 20.1;
y = 50.5;
z = x + y; // z is 70.6
```

Example-3

```
float x, y; //input variables
int z; //output variable as int data type
x = 20.1;
y = 50.5;
z = x + y; /* z is 70 as the decimal portion
              is neglected*/
```

ARITHMETIC OPERATIONS

Subtraction

Example

```
int x, y; //input
int z; //output variable
x = 70;
y = 50;
z = x - y; // z is 20
```

Multiplication

Example

```
int x, y; //input
int z; //output variable
x = 7;
y = 5;
z = x * y; // z is 35
```

Division

Example

```
int x, y; //input
int z; //output variable
x = 10;
y = 5;
z = x / y; // z is 2
```

CONDITIONAL OPERATIONS

Operator	Meaning	Syntax	Example
==	equal to	x == y // x is equal to y	12 == 10 is FALSE or 12 == 12 is TRUE
!=	not equal to	x != y // x is not equal to y	12 != 10 is TRUE or 12 != 12 is FALSE
<	less than	x < y // x is less than y	12 < 10 is FALSE or 12 < 12 is FALSE or 12 < 14 is TRUE
>	greater than	x > y // x is greater than y	12 > 10 is TRUE or 12 > 12 is FALSE or 12 > 14 is FALSE
<=	less than equal to	x <= y // x is less than or equal to y	12 <= 10 is FALSE or 12 <= 12 is TRUE or 12 <= 14 is TRUE
>=	greater than equal to	x >= y // x is greater than or equal to y	12 >= 10 is TRUE or 12 >= 12 is TRUE or 12 >= 14 is FALSE

LOGICAL OPERATIONS

Logical operators are used to compare two or more expressions and return a TRUE or FALSE depending on the operator

There are three logical operators AND, OR, and NOT that are often used in if statements

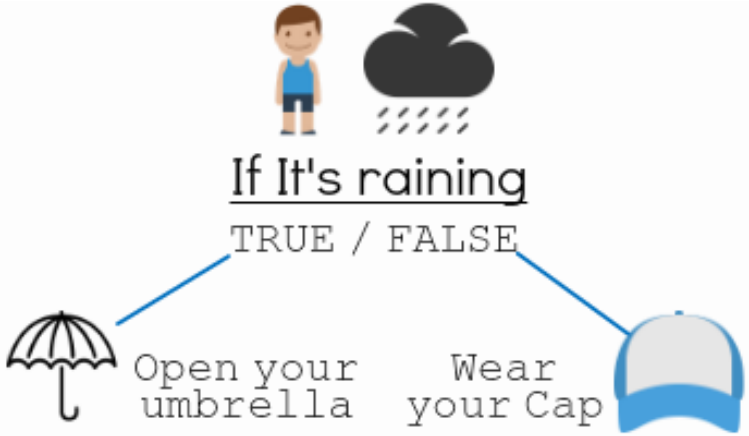
Logical AND:
if (x>0 && x<5) //true if both expressions are true

Logical OR:
if (x>0 || x<5) //true if either expressions are true

Logical NOT:
if (!x>0) //true only if expression is false

if-else STATEMENT

- The if () statement is the most basic of all programming control structures and can specify whether something should happen depending on whether a particular condition is true or not.
- It looks like this:



can be any expression that evaluates to a true or false

```
if (Raining){
Open_your_umbrella;
}
else{
Wear_your_cap;
}
```

if-else STATEMENT

There's also the else-if, where you can check a second condition if the first is false :

```
if (someCondition)
{ // do something if the condition is true
}
else if (anotherCondition)
{ // do something only if the first condition is false
  // and the second condition is true
}
else
{   // do something if both the conditions are false
}
```

```
if (Raining){
Open_your_umbrella;
}
else if(feeling_cold){
Wear_your_jacket;
}
else{
Wear_your_cap;
}
```

NESTED if-else STATEMENT

if-else statements with many conditions to check with:

```
if (someConditionA)
{
// do something if someConditionA is true
   if (someConditionB)
      {   // do something if someConditionB is true
      }
   else
      {   // do something if someConditionB is false
      }
}
else
{   // do something if someConditionA is false
}
```

```
if (Raining){
    if(feeling_cold){
   Open_your_umbrella &
   Wear_your_jacket;
    }
    else{
     Open_your_umbrella;
    }
}
else{
Wear_your_cap;
}
```

COMPOUND if-else STATEMENT

if-else statements with many conditions to check with:

```
if (someConditionA && someConditionB)
{
// do something if someConditionA and someConditionB
are true
}
else if (someConditionA && someConditionC)
{
  // do something if either of someConditionA and
someConditionB is true
}
else
{
// do something if the above conditions are false
}
```

```
if (Raining_and_cold){
Open_your_umbrella &
    Wear_your_jacket;
    }
else if(Raining_and_sunny){
Open_your_umbrella;
    }

else{
Wear_your_cap;
}
```

SWITCH CASE STATEMENT

switch keyword

Switch variable - can be char or int

```
switch (switch_var) {
    case '1':
      // statements placed here run if switch_var == '1'
    break;

    case '2':
      // statements placed here run if switch_var == '2'
    break;

    default:
      // statements placed here run if if no case found
    break;
}
```

Opening brace of switch body

int or char constant to check for

If case matches, statments are run followed by break which breaks out of the switch body

Closing brace of switch body

SWITCH CASE STATEMENT EXAMPLE

```
void setup() {
  Serial.begin(9600);
} // Initialize serial port
void loop() {
  if (Serial.available()) { // Check if at least one character available
    char ch = Serial.read();
    switch (ch) {
      case '1':
        Serial.println("You entered 1"); break;
      case '2':
        Serial.println("You entered 2"); break;
      case '+':
        Serial.println("You entered +"); break;
      case '-':
        Serial.println("You entered -"); break;
      case 10: //eliminate line feed
        break;
      default :
        Serial.print(ch);
        Serial.println(" was received but not expected");
        break;
    }
  }
}
```

Program

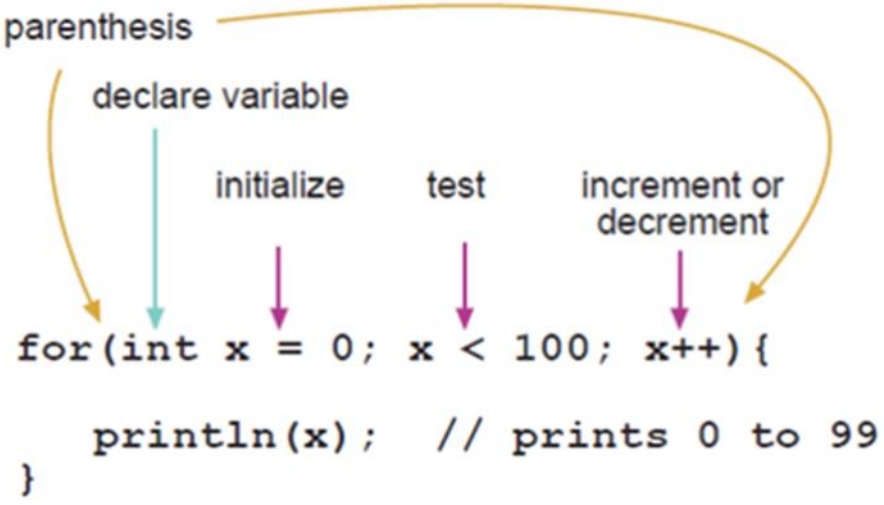
Output displayed on Serial Monitor

```
You entered 2
You entered 1
You entered -
You entered +
a was received but not expected
```

LOOP: for

- The for statement is used to repeat a block of statements enclosed in curly braces
- An increment counter is used to increment and terminate the loop
- The for statement is useful for any repetitive operation and is often used in combination with arrays to operate on collections of data/pins

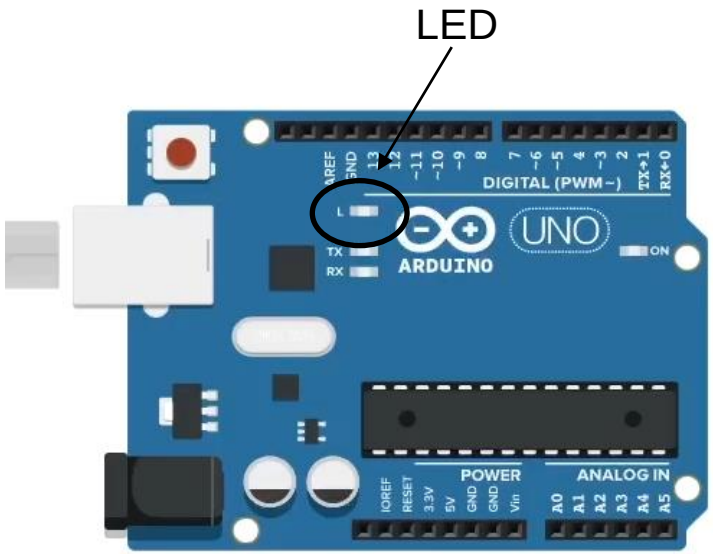
```
for(initializing;condition;increment){
  //statements;
  }
```



EXAMPLE: for Loop

```
int LEDpin = 13;

void setup() {
  // initialize digital pin LEDpin as an output.
  pinMode(LEDpin, OUTPUT);
  for (int i = 0; i < 10; i++)
  {
    digitalWrite(LEDpin, HIGH);
    // turn the LED on (HIGH is the voltage level)
    delay(1000);
    // wait for a second
    digitalWrite(LEDpin, LOW);
    // turn the LED off by making the voltage LOW
    delay(1000);
    // wait for a second
  }
}

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

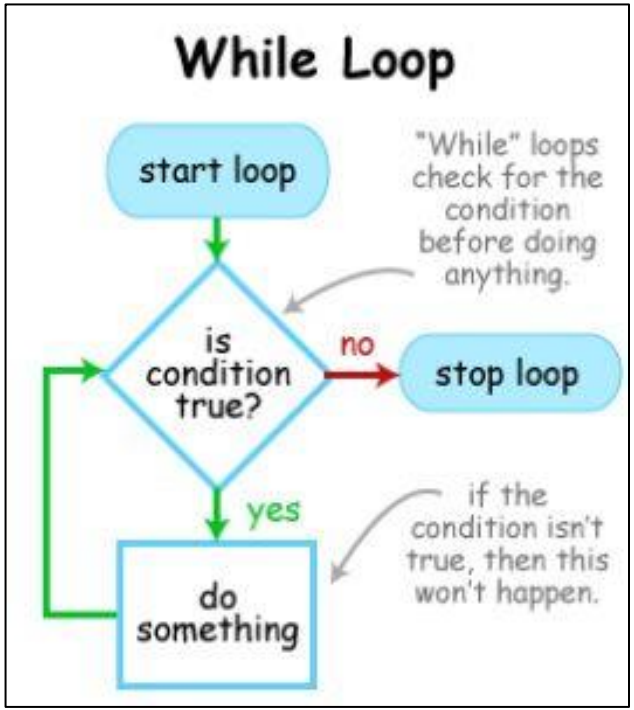
Program



LED blinking 10 times

Video

LOOP: while



While_Loop

```
int sum = 0;
void setup() {
  Serial.begin(9600);
  while (sum < 26) {
    Serial.print("sum = ");
    Serial.println(sum);
    delay(500);
    sum = sum + 5;
  }
}

void loop() {
}
```

Program

COM8

```
sum = 0
sum = 5
sum = 10
sum = 15
sum = 20
sum = 25
```

Output

LOOP: do-while

- A do-while loop works in the same manner as the while loop
- But the condition is tested at the end of the loop, so the do loop will always run at least once
- This is a bottom-driven condition

Do_While_Loop

```
int sum = 0;
void setup() {
  Serial.begin(9600);
  do {
    Serial.print("sum = ");
    Serial.println(sum);
    delay(500);
    sum = sum + 5;
  } while (sum < 26);
}

void loop() {
}
```

Program

COM8

```
sum = 0
sum = 5
sum = 10
sum = 15
sum = 20
sum = 25
```

Output

NYU TANDON SCHOOL OF ENGINEERING

Task / Activity:
Arduino hands on session

Innovative Technology Experiences for Students and Teachers (ITEST), Professional Development Program, July 2017-19

Mechatronics, Controls, and Robotics Laboratory, Department of Mechanical and Aerospace Engineering, NYU Tandon School of Engineering

TASK / ACTIVITY

Subtask 1– (a) Print your name continuously on the serial monitor in a new line

(b) Print your name only once on the serial monitor (Hint: where will you put the Serial.print() command so that the output is displayed only once?)

Subtask 2 – Write a program to declare variables (with the following names) that store this respective information:

1. **My_name:** Your name
2. **My_Grp_number:** Your group number
3. **My_Grp_age:** Average age of your team members

Print the above variables on the serial monitor

TASK / ACTIVITY

Subtask 3 – Write a program to

- Read a 3-digit number when you type it in the serial monitor using Serial.read()
- Store it in a variable
- Compute twice that number
- Print it on the serial monitor

Subtask 4 – Enter an integer on the serial monitor and check if it is odd or even

Serial monitor User Interface example:

(Input) Enter an integer: 27

(Output) 27 is an odd number

Subtask 5 – Write a program to blink internal led 10 times with time delays between blinks increasing by 1 second after every blink

TASK / ACTIVITY

Subtask 6 – Print the sum of the first 25 natural numbers using

a) while loop

b) do-while loop

c) for loop

Subtask 7 – Create an infinite loop

Solutions

NYU TANDON SCHOOL OF ENGINEERING

Thank You!

Questions and Feedback?

Innovative Technology Experiences for Students and Teachers (ITEST), Professional Development Program, July 2017-19
Mechatronics, Controls, and Robotics Laboratory, Department of Mechanical and Aerospace Engineering, NYU Tandon School of Engineering