Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

Basic Data Types

a. int (Integer)

- **Size**: 2 bytes (16 bits)
- **Range**: -32,768 to 32,767
- **Usage**: Stores whole numbers (positive and negative).

Example:

```
int myNumber = 100;
```

long (Long Integer)

- **Size**: 4 bytes (32 bits)
- **Range**: -2,147,483,648 to 2,147,483,647
- **Usage**: Used when you need a larger range of integers.

Example:

```
long largeNumber = 1000000;
```

float (Floating Point)

- **Size**: 4 bytes (32 bits)
- **Range**: ±3.4028235E+38 (6-7 decimal digits of precision)
- **Usage**: Stores numbers with decimal points (fractional numbers).

Example:

```
float temperature = 36.5;
```

d. double (Double Precision Floating Point)

- **Size**: Same as float (4 bytes) on most Arduino boards (for precision).
- **Usage**: Generally treated the same as float in Arduino.

Example:

```
double pi = 3.14159265;
```

Email: zahraa.hashim@uomus.edu.iq

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

2. Character and Boolean Types

a. char (Character)

- **Size**: 1 byte (8 bits)
- **Range**: -128 to 127 or 0 to 255
- **Usage**: Used to store characters or small integers.

Example:

```
char letter = 'A';
```

b. unsigned char

- **Size**: 1 byte (8 bits)
- **Range**: 0 to 255
- **Usage**: Stores unsigned 8-bit integers (no negative values).

Example:

```
unsigned char smallValue = 200;
```

c. bool (Boolean)

- **Size**: 1 byte (8 bits)
- **Values**: true or false
- **Usage**: Stores logical values (on/off, true/false).

Example:

```
bool isOn = true;
```

3. Unsigned Data Types

a. unsigned int

- **Size**: 2 bytes (16 bits)
- **Range**: 0 to 65,535
- **Usage**: Stores only positive integers.

Email: zahraa.hashim@uomus.edu.iq

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

Example:

```
unsigned int positiveNumber = 40000;
```

b. unsigned long

- **Size**: 4 bytes (32 bits)
- **Range**: 0 to 4,294,967,295
- **Usage**: Stores large positive integers.

Example:

```
unsigned long veryLargeNumber = 3000000000;
```

---

4. Special Data Types

a. byte

- **Size**: 1 byte (8 bits)
- **Range**: 0 to 255
- **Usage**: Stores small positive integers (often used for raw data or sensor values).

Example:

```
byte sensorValue = 150;
```

b. word

- **Size**: 2 bytes (16 bits)
- **Range**: 0 to 65,535
- **Usage**: Stores unsigned 16-bit integers, similar to unsigned int

Example:

```
word someWord = 50000;
```

c. string

- **Size**: Variable (depends on the length of the string)

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

- **Usage**: Stores text or sequences of characters (not the same as String object).

Example:

```
char myText[] = "Arduino";
```

---

5. Arrays

An **array** is a collection of variables of the same type, accessible by index.

Example (array of integers):

```
int numbers[5] = {10, 20, 30, 40, 50};
```

---

6. The String Object

Arduino also supports the **String** class, which provides more flexibility when working with strings of text.

Example:

```
String myMessage = "Hello, Arduino!";
```

Unlike a char array, String objects can be manipulated easily with functions like concat(), substring(), etc.

Summary Table of Arduino Data Types

Data Type	Size	Range
int	2 bytes	-32,768 to 32,767
unsigned int	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647

Email: zahraa.hashim@uomus.edu.iq

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

Data Type	Size	Range
unsigned long	4 bytes	0 to 4,294,967,295
float	4 bytes	±3.4028235E+38 (7 decimal digits precision)
double	4 bytes	Same as float on most Arduino boards
char	1 byte	-128 to 127 or 0 to 255
bool	1 byte	true or false
byte	1 byte	0 to 255
word	2 bytes	0 to 65,535
String	Variable	Depends on the length of the text

Conclusion

Arduino supports a variety of **data types** for storing different kinds of data, from simple integers and floating-point numbers to characters and strings. Understanding these types allows you to manage memory efficiently and write effective code for your projects.

1. Structure of an Arduino Program

Arduino programs (called **sketches**) consist of two main functions:

a. setup()

This function runs once when the Arduino is powered on or reset. It is used to initialize variables, pin modes, start using libraries, etc.

Email: zahraa.hashim@uomus.edu.iq

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

b. loop()

This function runs repeatedly in a cycle, after the setup() function finishes. It's where the main logic of the program goes.

```
void setup() {
  // This code runs once
}

void loop() {
  // This code runs continuously
}
```

---

2. Basic Syntax

a. Comments

Comments are used to explain code and are ignored by the compiler.

- **Single-line comment**: // This is a comment
- **Multi-line comment**:

```
/*
   This is a multi-line comment
*/
```

b. Semicolons

Each statement in Arduino must end with a semicolon (;).

---

3. Variables and Data Types

a. Declaring Variables

Variables in Arduino must be declared with a data type before they are used. Some commonly used data types are int, float, char, bool, etc.

```
int ledPin = 13;   // Declare an integer variable and assign value 13
```

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

float temperature; // Declare a floating-point variable

b. Constants

You can use the keyword const to declare variables whose values won't change.

```
const int sensorPin = A0;  // Constant variable
```

---

4. Control Structures

Arduino supports control structures like **if-else**, **for**, **while**, **switch-case**. These structures control the flow of execution in a program.

a. if-else

```
if (condition) {
  // Code to execute if condition is true
} else {
  // Code to execute if condition is false
}
```

Example:

```
int sensorValue = analogRead(A0);
if (sensorValue > 500) {
  digitalWrite(ledPin, HIGH); // Turn LED on
} else {
  digitalWrite(ledPin, LOW);  // Turn LED off
}
```

b. for loop

The for loop is used when you know how many times you want to repeat a block of code.

```
for (int i = 0; i < 10; i++) {
  // Code to execute 10 times
}
```

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

c. while loop

The while loop runs a block of code as long as the specified condition is true.

```
while (condition) {
  // Code to execute as long as the condition is true
}
```

---

5. Functions

Functions in Arduino allow you to break the code into smaller, reusable chunks. There are built-in functions as well as user-defined ones.

a. Built-in Functions

- **pinMode(pin, mode)**: Sets the mode of a specific pin (INPUT, OUTPUT, or INPUT_PULLUP).
- **digitalWrite(pin, value)**: Sets the specified pin to HIGH (5V) or LOW (0V).
- **digitalRead(pin)**: Reads the state (HIGH or LOW) from a digital pin.
- **analogRead(pin)**: Reads the value (0-1023) from an analog input pin.
- **analogWrite(pin, value)**: Outputs an analog value (PWM) to a pin (range 0-255).

Example:

```
pinMode(13, OUTPUT);    // Set pin 13 as output
digitalWrite(13, HIGH); // Turn on LED on pin 13
```

b. User-defined Functions

You can create your own functions to encapsulate reusable code.

```
void myFunction() {
  // Code for the custom function
}

void loop() {
```

```
  myFunction(); // Call the custom function
}
```

Example:

```
void blinkLED() {
  digitalWrite(13, HIGH); // Turn LED on
  delay(1000);            // Wait for 1 second
  digitalWrite(13, LOW);  // Turn LED off
  delay(1000);            // Wait for 1 second
}

void loop() {
  blinkLED(); // Call the blinkLED function
}
```

---

6. Pin Configuration

Arduino boards have both **digital** and **analog** pins.

- **Digital Pins**: Used for both input and output of digital signals (HIGH or LOW).
- **Analog Pins**: Used for reading analog signals (from sensors) and generating analog output via PWM.

a. Digital Pins

- **pinMode(pin, mode)**: Sets a pin as INPUT or OUTPUT.
- **digitalWrite(pin, value)**: Sets a pin HIGH (5V) or LOW (0V).
- **digitalRead(pin)**: Reads the value (HIGH or LOW) from a digital pin.

Example:

```
pinMode(13, OUTPUT);      // Set digital pin 13 as output
digitalWrite(13, HIGH);   // Turn on LED on pin 13
```

Al-Mustaqbal University
Department of Medical Instrumentation Techniques Engineering
Class: four
Subject: Advanced logic design
Lecturer: Dr. Zahraa hashim kareem
Lecture- 3: Arduino programming language-part2

b. Analog Pins

- **analogRead(pin)**: Reads analog input (0-1023) from a pin.
- **analogWrite(pin, value)**: Writes an analog value (PWM) to a pin (0-255).

Example:

```
int sensorValue = analogRead(A0); // Read analog value from sensor
analogWrite(9, sensorValue / 4);  // Output PWM value to pin 9
```

---

7. Libraries

Arduino has many **libraries** that extend its functionality, allowing you to interface with sensors, motors, displays, etc.

a. Including Libraries

To use a library, include it at the beginning of the sketch:

```
#include <Servo.h>  // Include the Servo library
```

b. Using Library Functions

After including the library, you can use its functions to control hardware. For example, the **Servo** library is used to control servo motors.

Example:

```
#include <Servo.h>  // Include Servo library

Servo myServo;      // Create a Servo object

void setup() {
  myServo.attach(9);  // Attach servo to pin 9
}
```


```
void loop() {
  myServo.write(90);  // Move servo to 90 degrees
  delay(1000);
  myServo.write(0);   // Move servo to 0 degrees
  delay(1000);
}
```

---

8. Communication

Arduino can communicate with computers, other Arduinos, or components via **Serial Communication**.

a. Serial Communication

Serial communication is used for debugging or exchanging data between the Arduino and a computer. The Serial library handles communication over the USB port.

```
void setup() {
  Serial.begin(9600);  // Start serial communication at 9600 baud
}

void loop() {
  int sensorValue = analogRead(A0);    // Read sensor value
  Serial.println(sensorValue);         // Print sensor value to Serial Monitor
  delay(1000);                         // Wait for 1 second
}
```

---

9. Timers and Delays

- **delay(milliseconds)**: Pauses the program for a specified number of milliseconds.

Example:


```
delay(1000);  // Wait for 1 second
```

- **millis()**: Returns the number of milliseconds since the program started. Useful for non-blocking timing logic.

Example:

```
unsigned long startTime = millis();  // Record start time

if (millis() - startTime > 1000) {
  // Code that runs after 1 second has passed
}
```

---

10. Interrupts

Interrupts allow you to stop the normal flow of the program to respond to external events, such as a button press. Arduino provides two functions for this:

- **attachInterrupt()**: Attaches an interrupt to a specific pin.
- **detachInterrupt()**: Removes the interrupt from the pin.

Example:

```
 attachInterrupt(digitalPinToInterrupt(2), handleInterrupt, RISING);

void handleInterrupt() {
  // Code to run when interrupt occurs}
```