Here’s a compact Arduino UNO command cheatsheet to help you quickly reference essential functions and commands for your projects:

### **1. Basic Structure**

c++

Copy code

void setup() {

// Initialization code here (runs once)

}

void loop() {

// Main code here (runs repeatedly)

}

### **2. Pin Mode Commands**

* **Set Pin Mode (Input/Output):**

cpp

Copy code

pinMode(pin, mode);

* pin: Pin number (e.g., 13)
* mode: INPUT, OUTPUT, or INPUT\_PULLUP

### **3. Digital I/O Commands**

* **Read Digital Pin (HIGH or LOW):**

cpp

Copy code

digitalRead(pin);

* Returns HIGH (1) or LOW (0)
* **Write to Digital Pin (Set HIGH or LOW):**

cpp

Copy code

digitalWrite(pin, value);

* pin: Pin number
* value: HIGH or LOW

### **4. Analog I/O Commands**

* **Read Analog Pin (0 to 1023 range):**

cpp

Copy code

analogRead(pin);

* pin: Analog input pin (e.g., A0)
* **Write to Analog Pin (PWM output):**

cpp

Copy code

analogWrite(pin, value);

* pin: Pin number (PWM-capable pins: 3, 5, 6, 9, 10, 11)
* value: 0 (off) to 255 (full brightness)

### **5. Timing Functions**

* **Pause Execution (milliseconds):**

cpp

Copy code

delay(milliseconds);

### **7. Math Functions**

* **Basic Arithmetic:**

cpp

Copy code

int sum = a + b;

int difference = a - b;

int product = a \* b;

int quotient = a / b;

int remainder = a % b;

cpp

Copy code

* **If-Else:**

cpp

Copy code

if (condition) {

// If condition is true

} else {

// If condition is false

}

* **For Loop:**

cpp

Copy code

for (int i = 0; i < 10; i++) {

// Code to repeat 10 times

}

* **While Loop:**

cpp

Copy code

while (condition) {

// Code to repeat as long as condition is true

}

* **Switch-Case:**

cpp

Copy code

switch (variable) {

case value1:

// Code for value1

break;

case value2:

// Code for value2

break;

default:

// Code if no case matches

break;

}

// Define the pin where the LED is connected

const int ledPin = 13; // On Arduino Uno, the built-in LED is on pin 13

void setup() {

// Initialize the digital pin as an output

pinMode(ledPin, OUTPUT);

}

void loop() {

// Turn the LED on (HIGH is the voltage level)

digitalWrite(ledPin, HIGH);

// Wait for 1 second (1000 milliseconds)

delay(1000);

// Turn the LED off by making the voltage LOW

digitalWrite(ledPin, LOW);

// Wait for 1 second

delay(1000);

}

### **Explanation:**

1. **pinMode(ledPin, OUTPUT);** — Sets the LED pin (pin 13 in this case) as an output so we can send voltage to it.
2. **digitalWrite(ledPin, HIGH);** — Turns the LED on by applying a HIGH (5V) signal to the pin.
3. **delay(1000);** — Pauses the program for 1000 milliseconds (1 second).
4. **digitalWrite(ledPin, LOW);** — Turns the LED off by applying a LOW (0V) signal to the pin.
5. **delay(1000);** — Pauses the program for 1 second again before looping back to the top and repeating the process.

### **How to Run the Code:**

1. Connect the long leg (anode) of the LED to digital pin 13 on the Arduino.
2. Connect the short leg (cathode) to a 220Ω resistor, and then connect the other end of the resistor to the GND pin on the Arduino.
3. Upload the code to your Arduino using the Arduino IDE.

Once uploaded, the LED should blink on and off every second.

### **Optional: Controlling LED with a Button**

Here’s an extended version that uses a button to turn the LED on or off.

cpp

Copy code

const int ledPin = 13; // LED connected to pin 13

const int buttonPin = 2; // Button connected to pin 2

int buttonState = 0; // Variable to store the button state

void setup() {

pinMode(ledPin, OUTPUT); // Initialize LED pin as an output

pinMode(buttonPin, INPUT); // Initialize button pin as an input

}

void loop() {

// Read the state of the button

buttonState = digitalRead(buttonPin);

// Check if the button is pressed

if (buttonState == HIGH) {

// Turn the LED on

digitalWrite(ledPin, HIGH);

} else {

// Turn the LED off

digitalWrite(ledPin, LOW);

}

}

### **Explanation:**

1. **buttonPin** is set to pin 2, where a button is connected.
2. The button is read using **digitalRead(buttonPin)**.
3. If the button is pressed (button state HIGH), the LED is turned on; otherwise, it’s turned off.

### **How to Set Up:**

1. Connect one side of the button to pin 2 and the other side to GND.
2. If needed, add a pull-up resistor (or use the internal pull-up feature with pinMode(buttonPin, INPUT\_PULLUP);).
3. The LED will turn on when the button is pressed and off when released.

This is a simple and effective way to control an LED using either a timer (blinking) or user input (button).