In order to program the Arduino, we need to use the Arduino IDE. An IDE is an Integrated Development Environment. Arduino software is written in “Sketches”. The sketches allow you to program the Arduino using simple code without having to know machine code.

**Computer hardware:**

1. Connect the board to your computer via USB. We don’t need to connect the power because the USB connection will deliver both data and power.
Computer software:


Environment setup and Embedded programming

2. Start the Arduino IDE like a regular program.

3. Setup and start the IDE. When you start the IDE, you’ll be presented with the default sketch. The verify button of the IDE will verify that there are no syntax errors in your code. The Upload button will upload your sketch to the board. The open button let you open saved sketches and files. The Tools tab contains various tools for working in the IDE.

4. setup( ) and loop ( ) functions
   3.1 setup( ) is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup function will only run once, after each powerup or reset of the Arduino board.[1] . setup( ) initializes and sets the initial values
   3.2 loop ( ) is called after creating a setup() function. The loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board.[2]
5. Select Computer hardware you are working with

In the Arduino IDE, select the board you’re using in Tools. In this example, we will be using Arduino Uno.
6. Setup communication mode

Set the port through Tools>Port> and select the port your board is connected to. This is important so the computer knows which port to read/write information to.

6. Add the onboard **led** as a field variable. The onboard led is commonly **pin 13**, if this doesn’t work then check the specifications of your board.
7. Configure input and output

Set the pin mode for the onboard led to output. This let’s the program know we’ll be writing to that pin instead of reading from it.

```cpp
int ONBOARD_LED = 13;

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

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

7. Setup a simple blink in the loop.

This can be done by lighting the led, waiting, then turning it off, then waiting again.

```cpp
void setup() {
    // put your setup code here, to run once:
    pinMode(ONBOARD_LED, OUTPUT);
}

void loop() {
    // put your main code here, to run repeatedly:
    digitalWrite(ONBOARD_LED, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(ONBOARD_LED, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
8. Verify your code to make sure that there are no syntax errors. It will prompt you to save your sketch if you haven’t already done so.
9. Press button “-” and upload your sketch to check that your program is working. The led built onto your board should be blinking now.

```cpp
int ONBOARD_LED = 13;

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

void loop() {
    // put your main code here, to run repeatedly:
    digitalWrite(ONBOARD_LED, HIGH); // turn the LED on (HIGH is the voltage level)
    delay(1000); // wait for a second
    digitalWrite(ONBOARD_LED, LOW); // turn the LED off by making the voltage LOW
    delay(1000); // wait for a second
}
```

10. Exercise by yourself

   Change the value delay (1000) to delay (2000) and press button “-”, observe the led blinking
   Change the first delay to delay (500) and press button “-”, observe the led blinking
   Choose different pin to work on different led.

References: