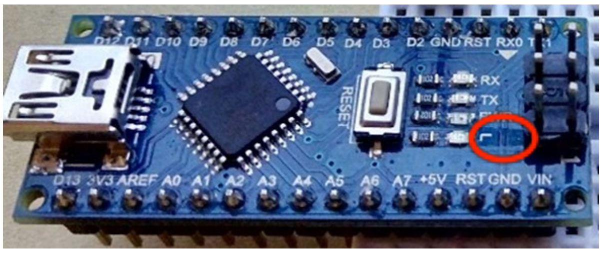
# Hardware Worksheet

The aim of this worksheet is to help you get to grips with the Arduino development platform.
Follow each step in the instructions carefully (the hints and tips given will help you out)
If you get stuck or need any help, just ask !

## Task 1 – Lights

In this task, we are going to do the simplest possible thing with the Arduino – flash a built-in LED


D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 GND RST RX0 TX1
RESET
RX
TX
D13 3V3 AREF A0 A1 A2 A3 A4 A5 A6 A7 +5V RST GND VIN

(1) Make sure the USB cable is plugged in (connecting the Arduino to the computer)
(2) Open up the Arduino development environment and enter the following code into the editing area:

```
// This section gets run once, when Arduino first starts up
void setup()
{
 // Switch pin number 13 (the LED pin) to output mode
 pinMode(13,OUTPUT);
}

// This section gets run over and over again (after startup)
void loop()
{
 // Set pin 13 to “high” (i.e. ON)
 digitalWrite(13,HIGH);
 // Wait for 1 second (1000 milliseconds)
 delay(1000);
 // Set pin 13 to “low” (i.e. OFF)
 digitalWrite(13,LOW);
 // Wait for 1 second (1000 milliseconds)
 delay(1000);
}
```

(3) Save the file onto the desktop, with the filename “Controller”
(4) Select the “Arduino Nano” board using the “Tools>Board” menu
(5) Select the USB port that the Arduino is attached to using the “Tools>Port” menu
(hint: the port to use will *probably* be the last one in the list !)
(6) Upload your code onto the Arduino using the upload button: ➔

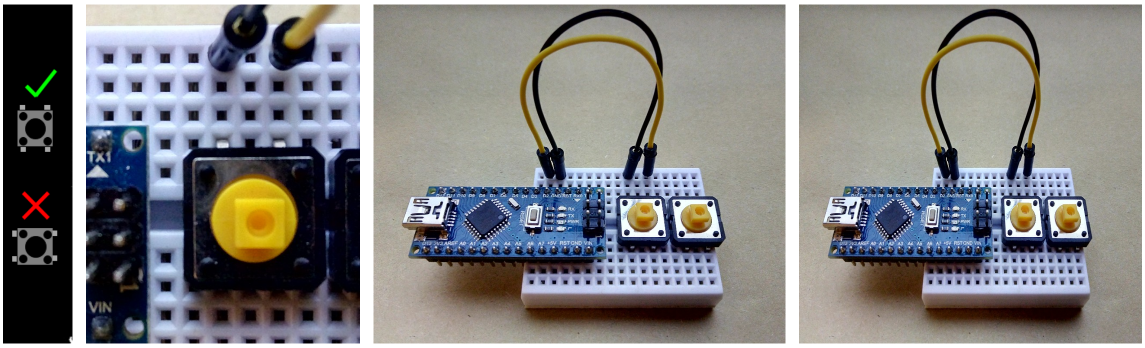
The lights on the Arduino should flash for a few seconds and then you should get a “Done uploading” message just above the black box on the Arduino editor. It you get any orange error messages then something is wrong with your code, or your Arduino isn’t plugged in. Check everything and then ask for help if you can’t see what’s wrong.

If everything worked OK, the LED circled in the diagram above should flash slowly (once per second).

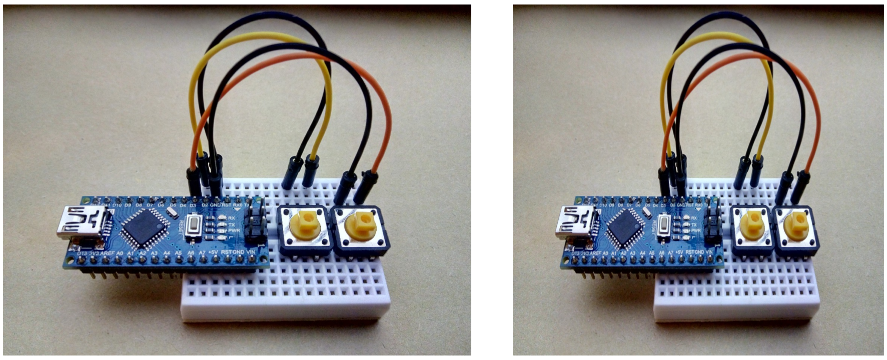
## Task 2 – Input

As well as outputting control signals with digitalWrite, we can also read in sensor values with digitalRead (for on/off components like buttons) or analogRead (for variable values like temperature or light levels). In this task, we are going to use digitalRead to detect when a button has been pressed:

(1) Make sure the USB cable is NOT plugged in (in case you accidently short the chip when wiring)
(2) Wire up the first button as shown in the images below:



(3) Now wire up the second button as shown in the image below:



(4) Connect the Arduino to the computer with the USB cable
(5) Change your code so it now looks like the following:

```
void setup()
{
 pinMode(2,INPUT_PULLUP);
 pinMode(3,INPUT_PULLUP);
 Serial.begin(9600);
}

void loop()
{
 if(digitalRead(2) == LOW) Serial.write('z');
 if(digitalRead(3) == LOW) Serial.write('x');
 delay(100);
}
```

(6) Upload your code onto the Arduino in the usual way.
(7) Once you get the “Done uploading” message, click on the “Serial Monitor” button 🔍

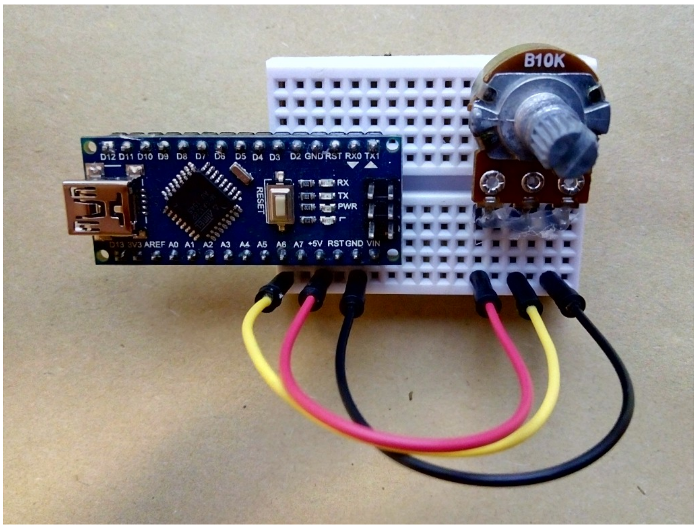
Each time you press a physical button, you should see a character being sent from the Arduino and appearing in the serial monitor window (the left button should be ‘z’, the right button should be ‘x’).

**Task 3 – Game controller**
Instead of receiving these characters in serial monitor, we can read them straight into an application:

(1) Quit the Arduino editor
(2) Open up the Pong game by double clicking on the PongGame.pde file
(3) Make sure you are in “Java” mode (see the drop-down menu on the right hand side)
(4) The press the “Run” button ▶

The Pong game will appear – you should now be able to control the bat with the physical buttons !

**Task 4 – Rotary Controller**
As an extra task (if you have time), replace the two buttons with a rotary controller (potentiometer) in order to provide the player with better control of the bat. Wire up the controller as shown in the image below:


B10K
D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 GND RST RX0 TX1
RESET
RX
TX
PWR
D13 3V3 AREF A0 A1 A2 A3 A4 A5 A6 A7 +5V RST GND VIN

You will need to write some new Arduino code (that makes use of analogRead) and also add more code to the main pong game !
This is a hard exercise – do ask the practical supervisors for help (if you get stuck)