

ÁREA DE TECNOLOGÍA ELECTRÓNICA - URJC

Arduino Notebook

Notebook – Traffic light

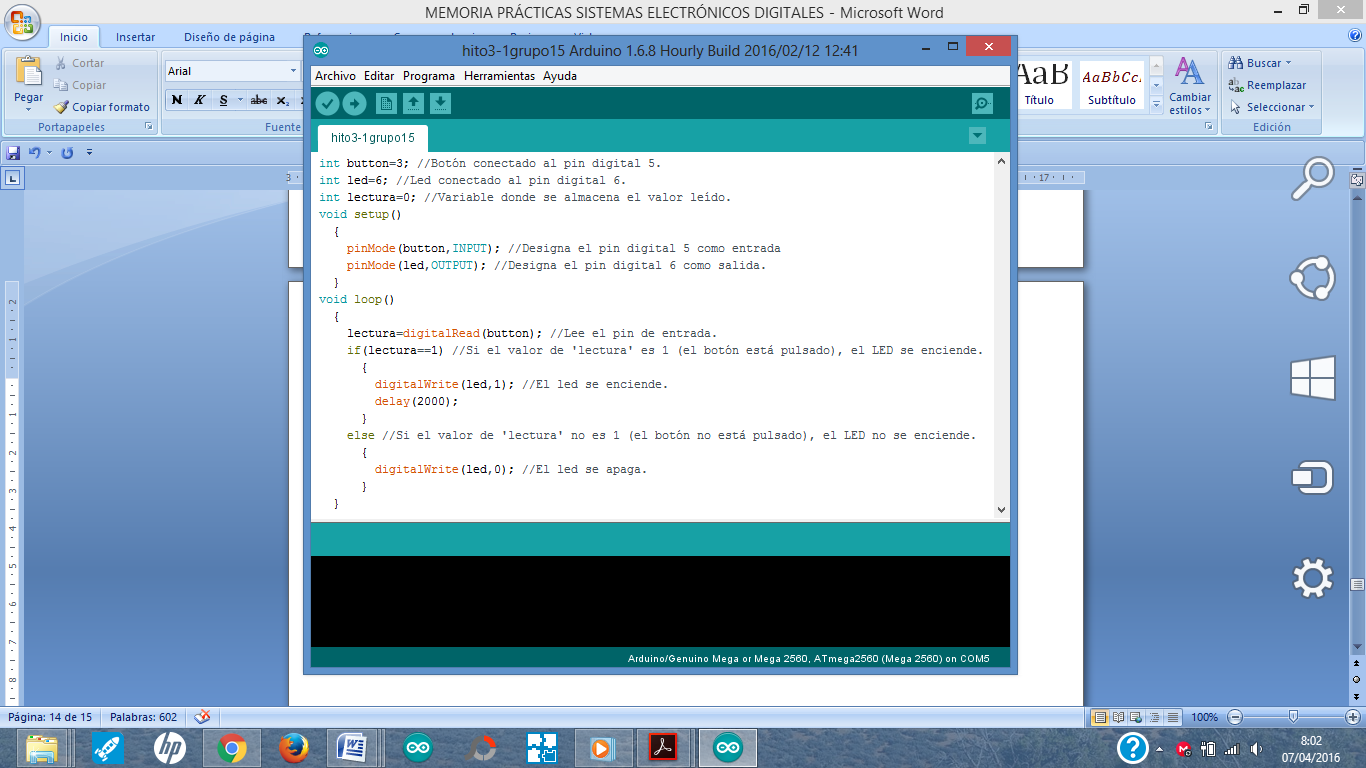
Parte 1: Checking hardware

In this chapter, we will start working with some hardware elements that you can connect to Arduino. Then, in the second part we give you more specific examples of the practice kit.

At the end of the module you will be able to develop electronic prototypes integrating different input elements (sensors, analog input, digital, buttons, switches) and output (LEDs, servos, screens). We recommend you to follow the book that comes with the kit for the assembly of the different elements. You will see that it is a plate that is superimposed on your Arduino board, but if you have an Arduino Uno you can also work without problems with this script.

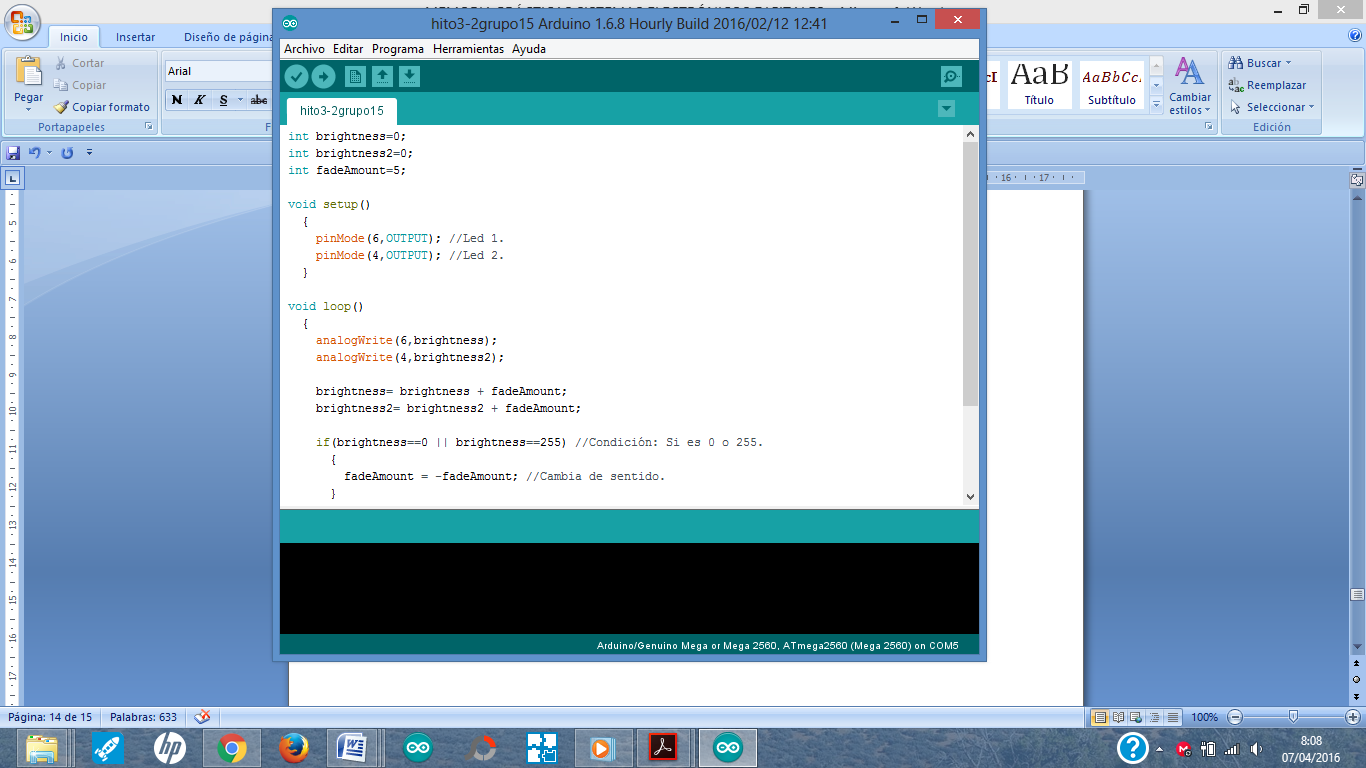
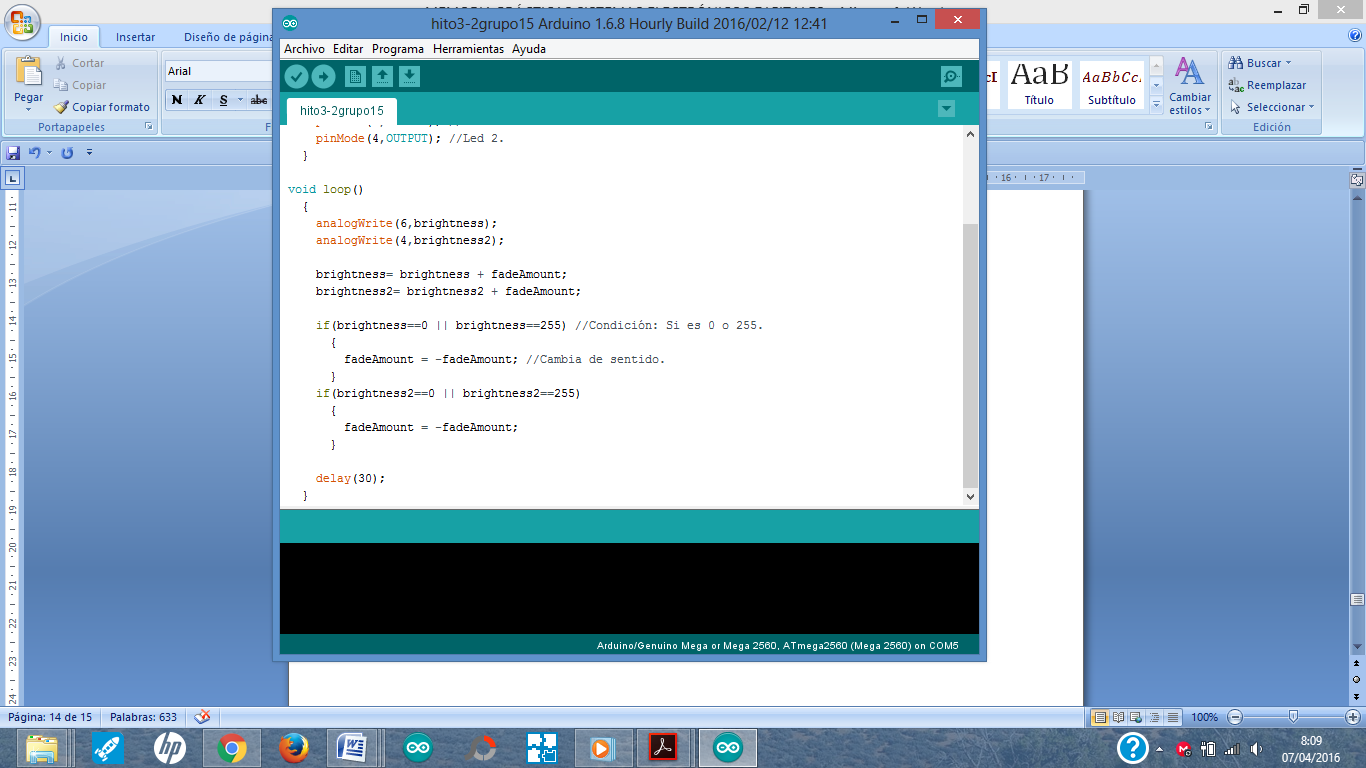
2. HITO 2.1

The following is an example to test the use of the "button" module - button and interrupt. This program turns on the LED when the button is pressed, if the button is not pressed, the LED is off.



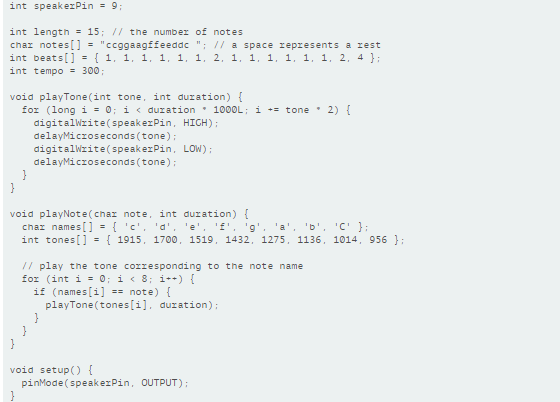
2. HITO 2.2

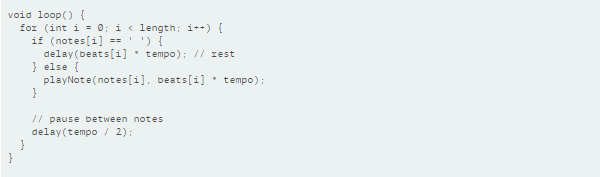
The following is an example to control the brightness of 2 different leds. Once done change the code to work with 3 leds.

3. HITO 2.3

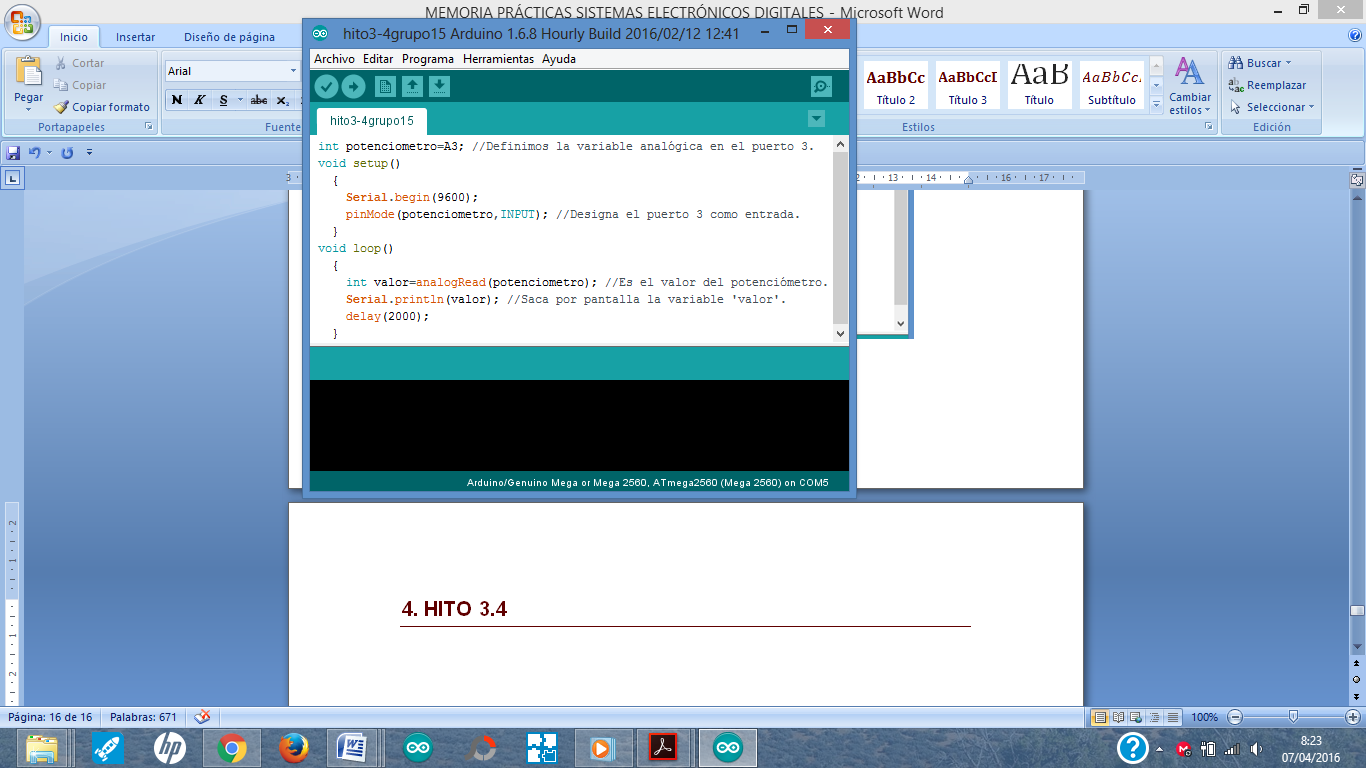
The following is an example to allow you to use the "buzzer" module - which basically allows you to emit sound from your Aruduino plate (eg alarm or music type) by connecting the buzzer to the development board.





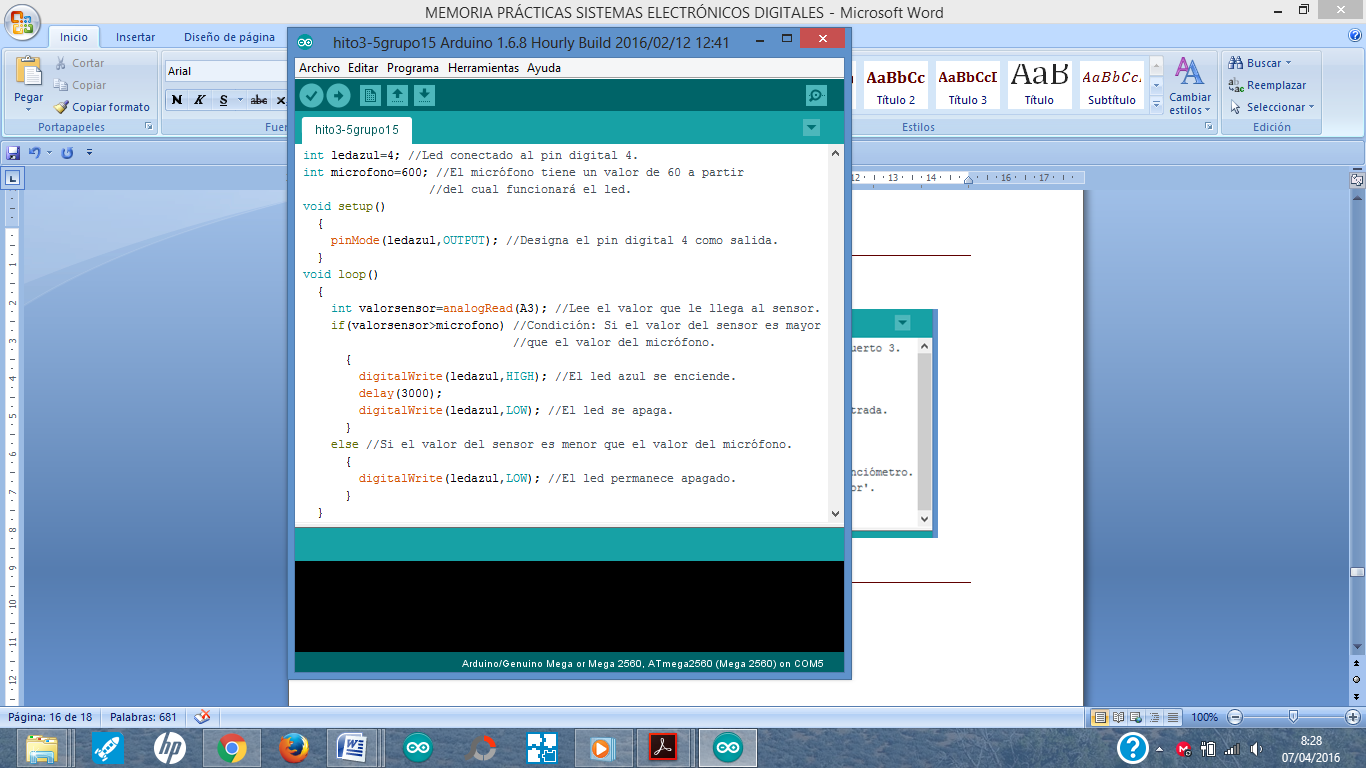
4. HITO 2.4

The following is an example to connect the sensor "Rotary angle sensor" that allows you to give information about the degrees of rotation scaled to the potentiometer. Once you get this program to work for you, you can include a sensor how to open doors or similar, and with the same program connecting in the same input pin you will have the value of the opening angle of the door.



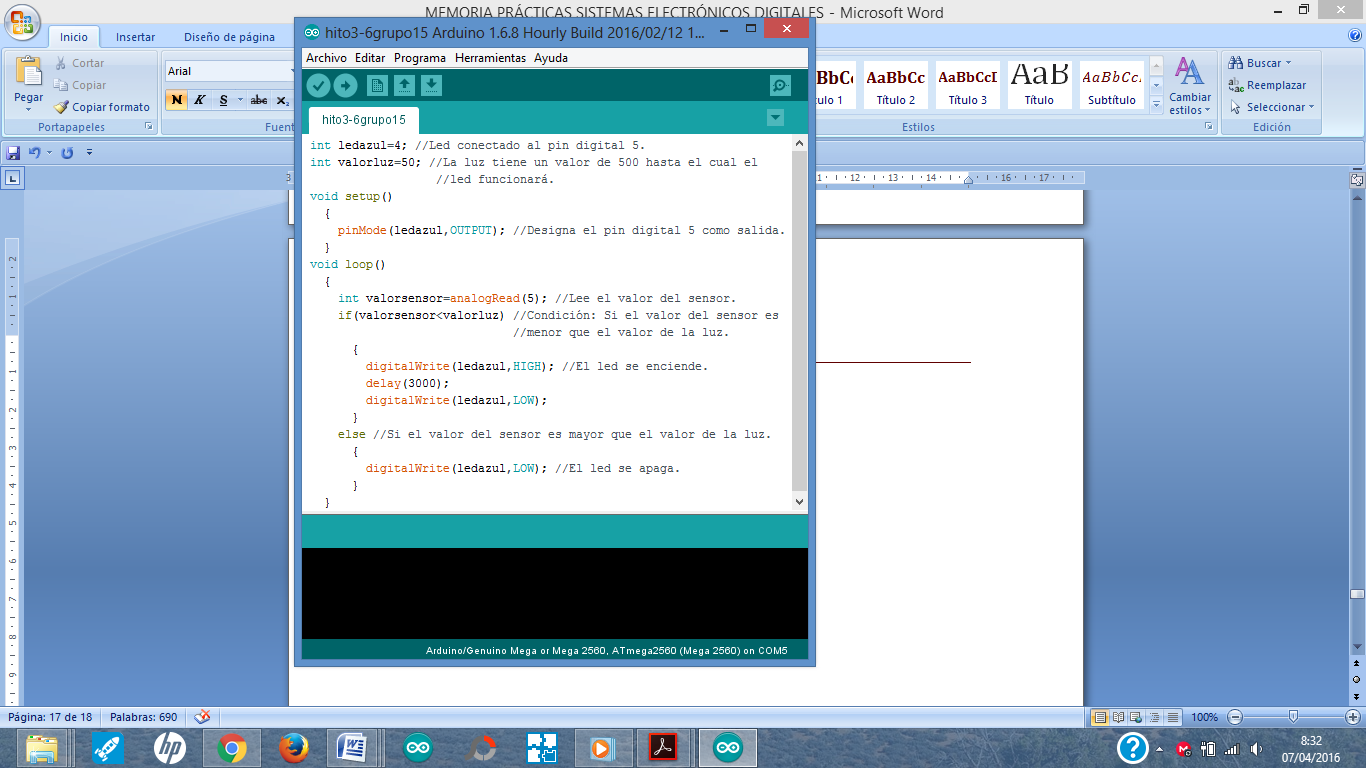
5. HITO 2.5

The following is an example to use the Sound Sensor - "Sound Sensor". Try to emit different tones of voice, more acute, serious, fine, and even music from your mobile so you can see that the data that appears on the screen varies depending on the type of sound.



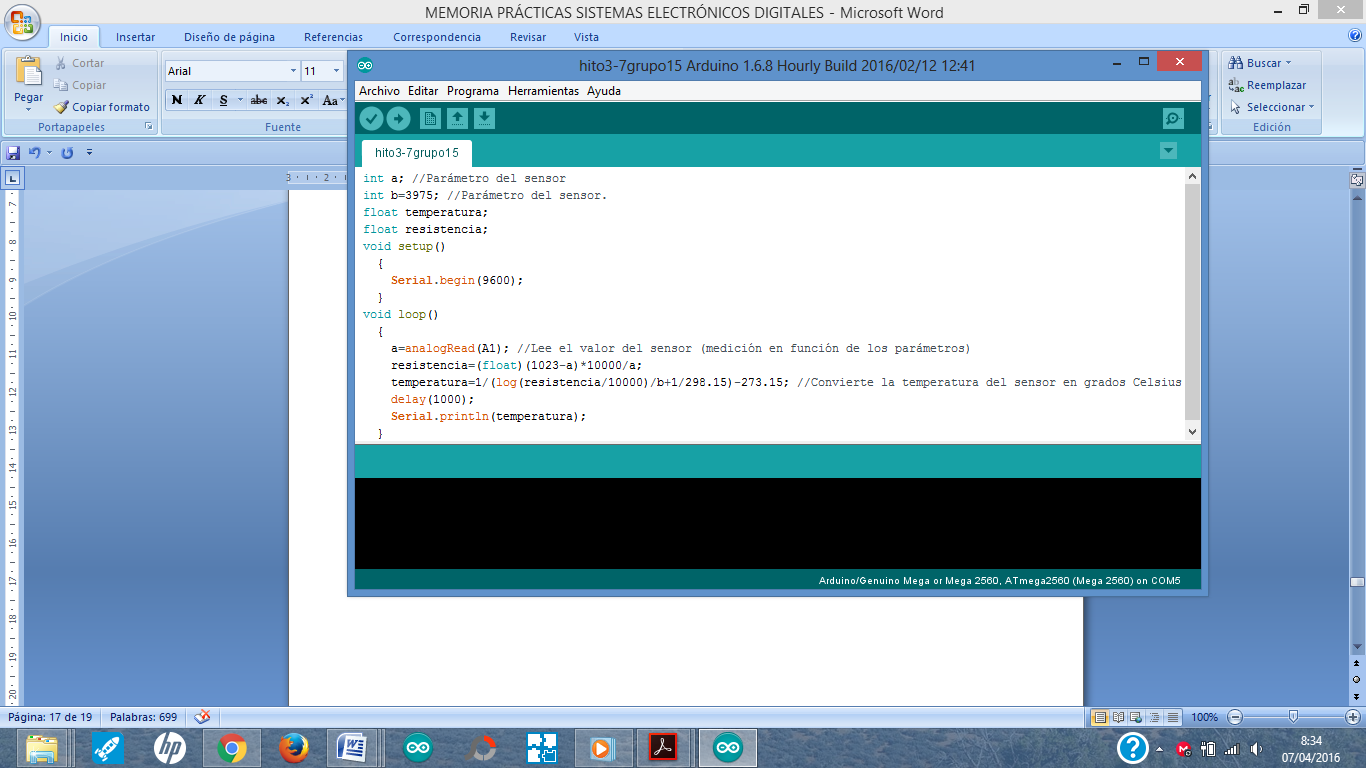
6. HITO 2.6

The following is an example to use the Light Sensor - "Light Sensor". Depending on the ambient light in the room will vary the value that comes through the screen, you can try turning on the light, lower the blinds, etc., to see the different values. You will see that depending on the light the value is different. Make a program that depending on the range that you see of values tells you if there is low light, medium light a lot of light.



7. HITO 2.7

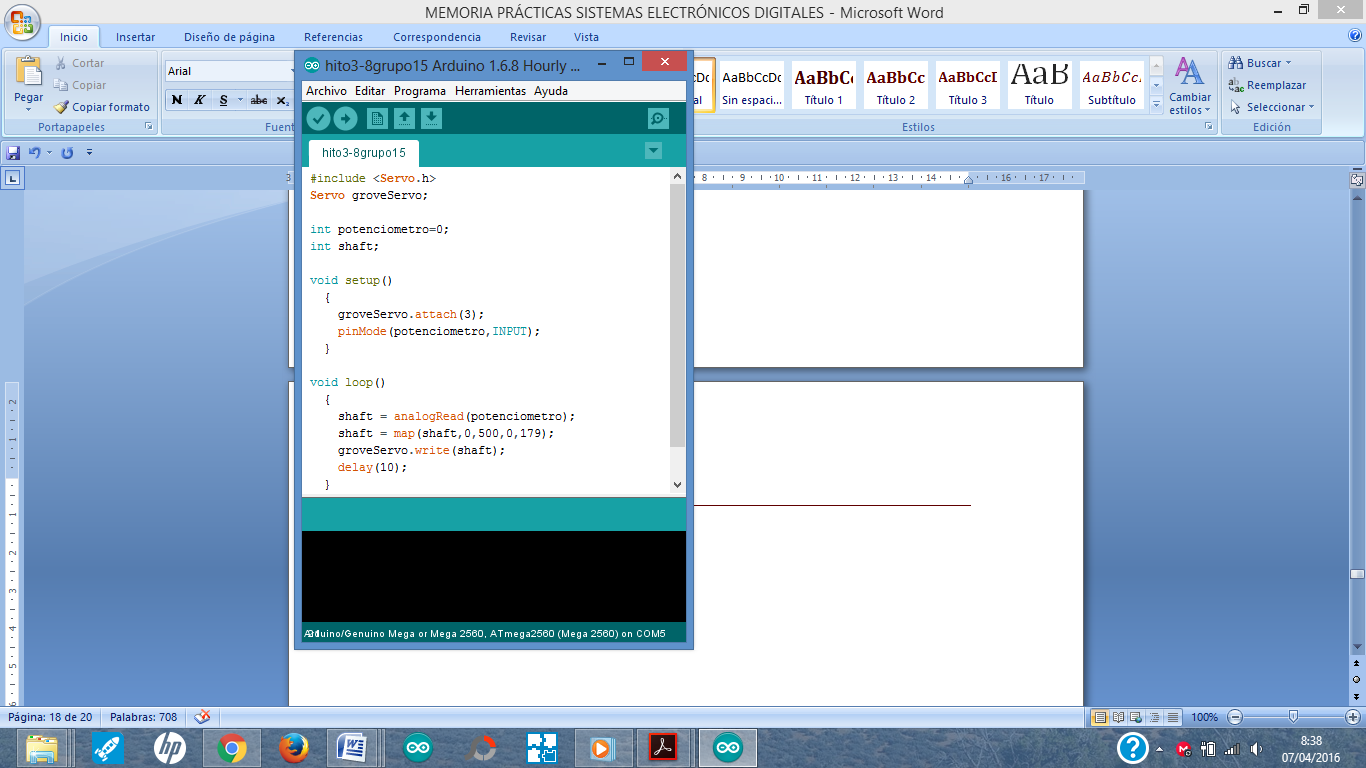
The following is an example to use the Temperature Sensor - "Temperature Senser". You should check the formula to display the degrees in Celsius or Fahrenheit. Change the program to indicate the value of the temperature in the two scales.



8. HITO 2.8

The following is an example for the use of a servo - "Servo". Basically this example will allow the device you connect that comes in the kit to be moved very little by little. We recommend that as you put in the example, connect the potentiometer to the plate of this example so that depending on the registered value, turn what you indicate with the potentiometer. You can also change the program so that there are different ranges in the rotation operation depending on what you read from this sensor.

Another example that you can do is that in addition to turning, with the sound device, you can emit sounds more or less attenuated depending on the values recorded in this milestone.



Stage 2

3.1. Traffic Ligth

Smart traffic light

Next, the block diagram of the system that you will have to implement so that it acts as a pedestrian traffic light is indicated. You must implement the code and the hardware part so that it meets the requirements of each of the states. Read carefully each section

Section 3.2 describes the different states that your smart semaphore should implement.

Then, in section 3.3 (SYSTEM DESIGN), the hardware components you will need and details of the implementation of each state will be described. Section 3.4 describes the high-level code that you must perform. Section 3.5 the example hardware assembly. And section 6 what you must deliver.

3.2. BLOCK DIAGRAM

The traffic light will start with state 1 and you will end up performing the steps described below.

**STAGE 1**

**CARS ARE GOING**

SET THE SYSTEM

**PEDESTRIAN PUSH BOTTON**

**STAGE 2**

**PEDESTRIAN WAIT**

**VEHICLES REDUCE SPEED**

Intermittent

**STAGE 3**

**PEDESTRIAN WAIT**

**VEHICLES STOP**

Fixed Yellow

**STAGE 4**

**MUSIC AND PEDESTRIAN START WALKING**

**CAR STOP**

**END OF CYCLE**

RESET THE PROCESS

**VEHICLES START TO GO**

**PEDESTRIAN CAN PULSE AGAIN**

3.3. DESIGN THE SYSTEM

The smart traffic light consists of a button, a buzzer and four LEDs (red, yellow, green and blue).

The designed traffic light follows the following sequence. Initially, only the green LED is on, giving way to cars (STATE 1).

When pressing the button the system will activate, after two seconds the green will turn off and the blue will light indicating that the pedestrian must wait. Simultaneously, the yellow LED flashes (flashing) that indicates the vehicles that reduce their speed (STATE 2).

Subsequently, the blue will remain lit indicating 'Pedestrian wait' while the yellow will stop blinking and stay 'Yellow fixed' for four seconds. The latter will serve to indicate to the vehicles that they must stop because it is about to turn red (STATE 3).

Finally, the yellow and blue LEDs will turn off, then the red LED will light indicating that the vehicles must be stopped and the melody sounds indicating that pedestrians can cross. Finally, the red LED will turn off (STATUS 4).

On the theme of melody. A sound is nothing more than a vibration of the air that our ears can grasp. A sound that has a certain tone depends on the frequency at which the air vibrates. The musical notes are vibrations of certain frequencies. Of course, in the creation of music, many other complex factors are involved, such as, for example, the timbre.

To create music with Arduino we must define a series of notes, which in English are defined by letters, as well as the time we want each note to last. We can also include spaces between the notes, to put silences in our melody.

Therefore, as you will see in what you have to implement in the next section, the melody function simply runs through all the notes and calls PlayNote to play them with the appropriate duration. The PlayNote function assigns the appropriate frequency to the input note we want to play, and calls PlayTone. The PlayNote function plays the right note, vibrating the Buzzer for the duration of the note, with the input frequency that defines the specific note.

When this process ends, it returns to its initial state, where only the green LED will be on, waiting for the user to press the button to restart the system.

3.4. PSEUDOCODE

Next, we indicate the variables by way of example and the pseudocode that you should follow for the elaboration of the traffic light. If you are going to make any additional changes you must document it in the delivery of this block.

**VARIABLES**

**TYPE int** ledrojo, ledamarillo, ledverde, ledazul, boton, sonido, length, beats[], tempo, posicionboton.

**Character** notes[].

**Configure function**

**Function setup**

Start

Configure Serial ← 9600;

Pin digital button ← INPUT;

Pin digital ledrojo ← OUTPUT;

Pin digital ledamarillo ← OUTPUT;

Pin digital ledverde ← OUTPUT;

Pin digital ledazul ← OUTPUT;

Pin digital sound ← OUTPUT;

End

**PROCESOS Y FUNCIÓN CÍCLICA (LOOP)**

**(you can search another example of tone in google for Arduino)**

**Procedure** playTone (**entero** tone, duration)

Start

From i ← 0 to i < duration \* 1000L with tone \* 2 do

sound ← HIGH;

delaymicroseconds (tone);

sound ← LOW;

delaymicroseconds (tone);

End for;

End

**Procedure** playNote (**caracter** note, **int** duration)

Start

names[] ← { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };

tones[] ← { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

From i ← 0 to i < 8 with 1 step

If (names[i] = note) then

playTone (tones[i], duration);

End if;

End for;

End

**Procedure** suenamelodia

Start

From i ← 0 to i < length with 1 step

If (notes[i] = ' ') then

delay (beats[i] \* tempo);

else

playNote (notes[i], beats[i] \* tempo);

End if;

delay (tempo / 2);

End from;

End

**Function loop**

Start

ledverde ← HIGH;

posicionboton ← read button;

If posicionboton = 1 then

wait (2000);

ledverde ← LOW;

ledazul ← HIGH;

for From i ← 0 to i<11 with 1 step

ledamarillo ← HIGH;

delay (1000);

ledamarillo ← LOW;

delay (1000);

End for;

ledamarillo ← HIGH;

delay (4000);

ledamarillo ← LOW;

ledazul ← LOW;

ledrojo ← HIGH;

suenamelodia ();

ledrojo ← LOW;

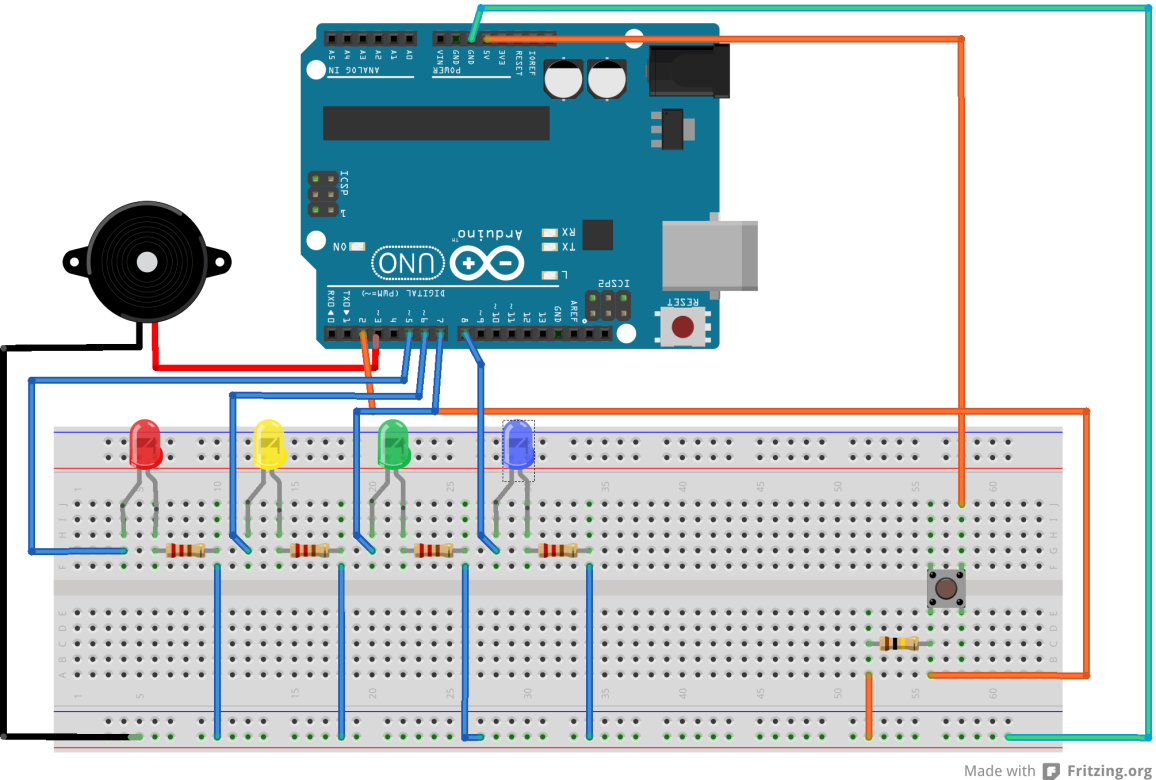
End if;

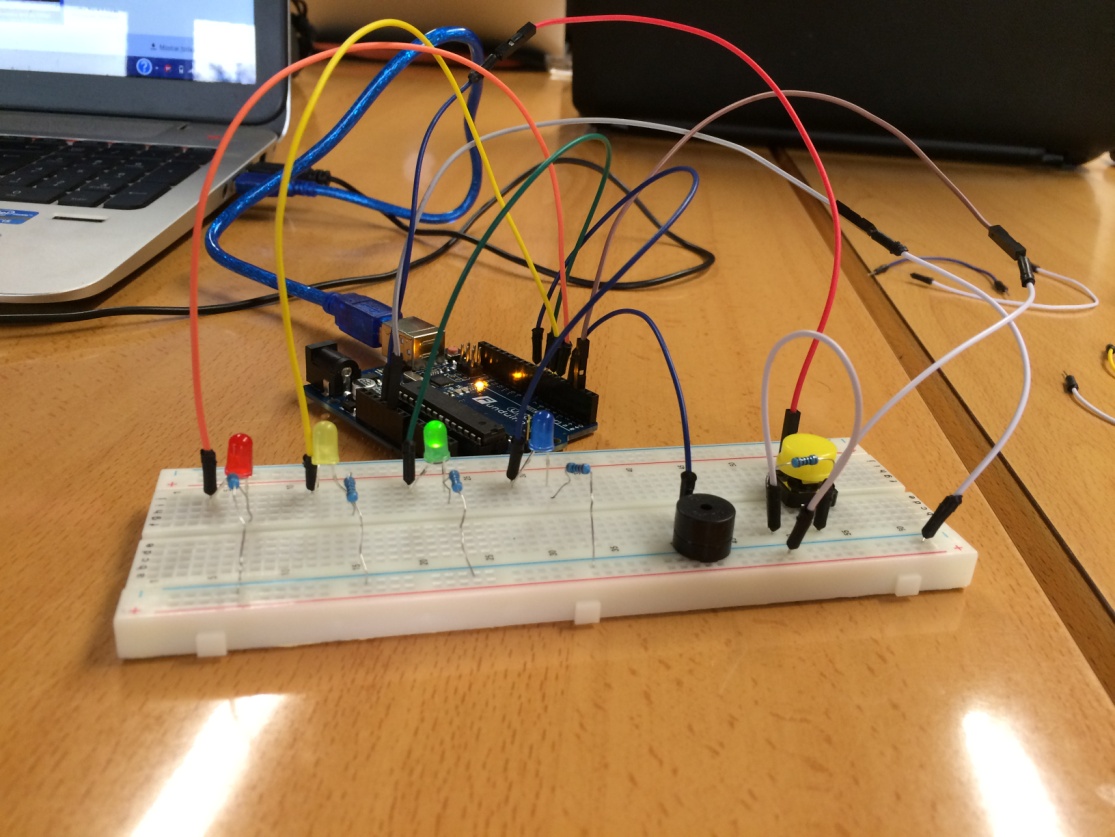
End

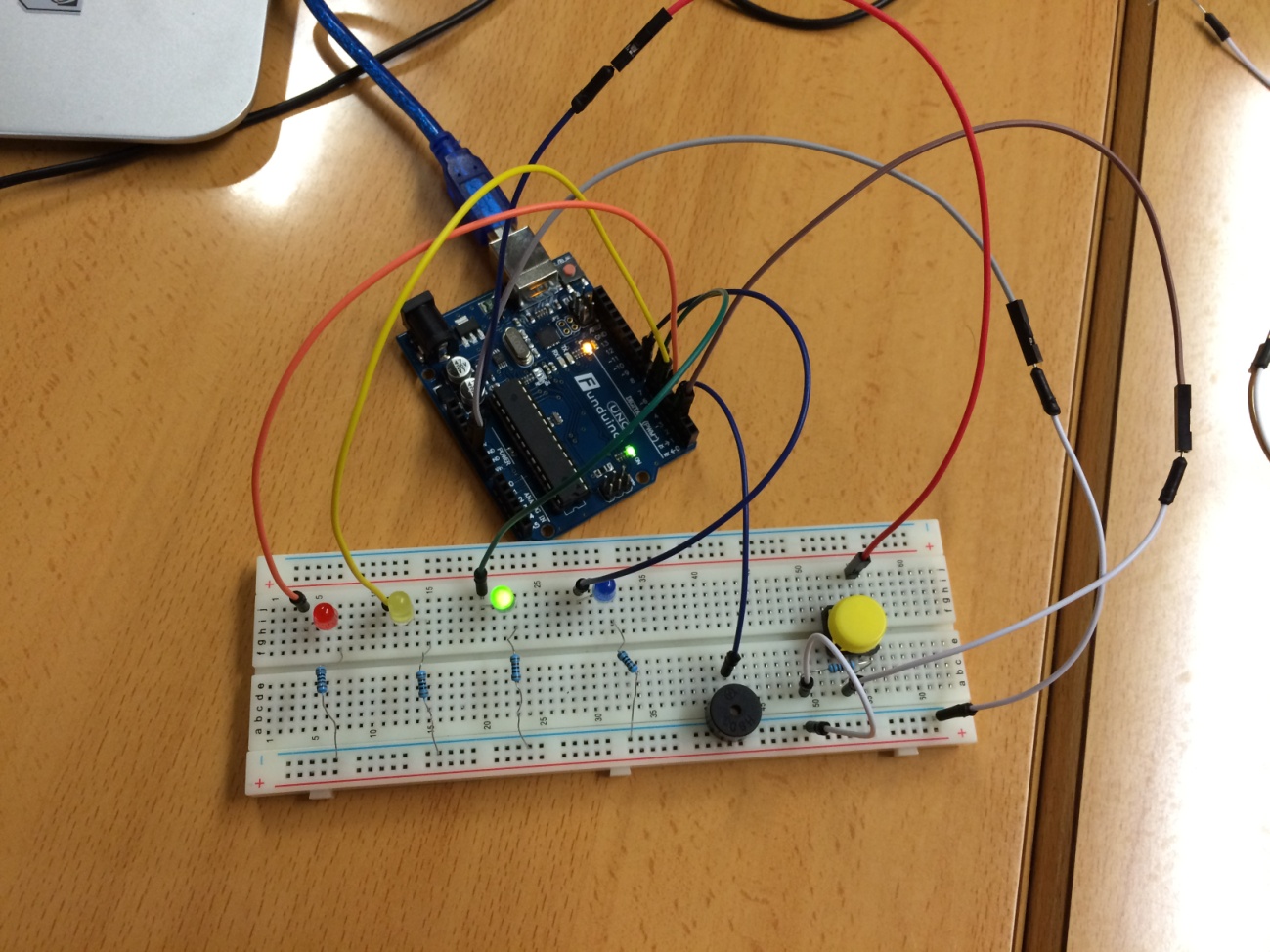
3.5. EMPEZAMOS EL MONTAJE

The first thing you should do after having understood the previous steps is to start the hardware assembly.

Below, we present an example of the Digital Traffic Light assembly according to the specifications proposed up to 3.4. If you make changes to make a different semaphore, you must also document it with images of the assembly in the delivery of the campus (see point 3.6 we explain the delivery). Although the image shows an Arduino Uno you can adapt it to your practice board without problems.







3.6. END STAGE – FOR TEACHER

Para **que la entrega del semáforo esté aprobada** hay que mostrar el funcionamiento del semáforo digital siguiendo las mismas instrucciones que te hemos planteado, o planteando tu propio semáforo digital al profesor en la fecha establecida por el mismo.

Puedes incorporar nuevas funcionalidades como que sólo se ponga en verde si se detecta presencia de peatones (sensor magnético), que dure más o menos el semáforo en función del tiempo, etc.

Si realizas las mismas instrucciones de esta memoria en la entrega sólo debes hacer un .zip con el siguiente contenido y formato:

* Entrega2-NumeroGrupoX-sed.zip,
* Contenido: Debes entregar un fichero .docx que tenga las siguientes secciones denominadas del siguiente modo y que tengan lo siguiente:
  + Sección 1-Semáforo: La primera página con el nombre y apellidos de los miembros del grupo.
  + Sección 2-Semáforo: Tabla de Verdad que represente el funcionamiento de tu código y puertas lógicas asociadas.
  + Sección 3-Semáforo : El código de tu fichero de Arduino copiado dentro del mismo documento.

Si realizas un semáforo diferente al de esta memoria en la entrega debes hacer un .zip con el siguiente contenido y formato:

* NumeroGrupoX-sed.zip,
* Contenido: Debes entregar un fichero .docx que tenga las siguientes secciones:
  + Sección 1-Semáforo: La primera página con el nombre y apellidos de los miembros del grupo.
  + Sección 2-Semáforo: Tabla de Verdad que represente el funcionamiento de tu código.
  + Sección 3-Semáforo: Diagrama de estados / bloque para explicar el funcionamiento de tu bloque y la descripción que consideres para explicar cómo funciona.
  + Sección 4-Semáforo: El código de tu fichero de Arduino copiado dentro del mismo documento.
  + [optativo] Sección 5-Semáforo: Un enlace a un video que demuestre su funcionamiento (o bien, fotos de su funcionamiento en ese fichero).