

Espacio de Experimentación en Usos Creativos de la Tecnología Digital

Guía de proyectos, número 1

**Material versión borrador, por favor no distribuir**

*Camino Centenario 2565*

*Gonnet, Pcia de Buenos Aires*

*www.unipe.edu.ar*

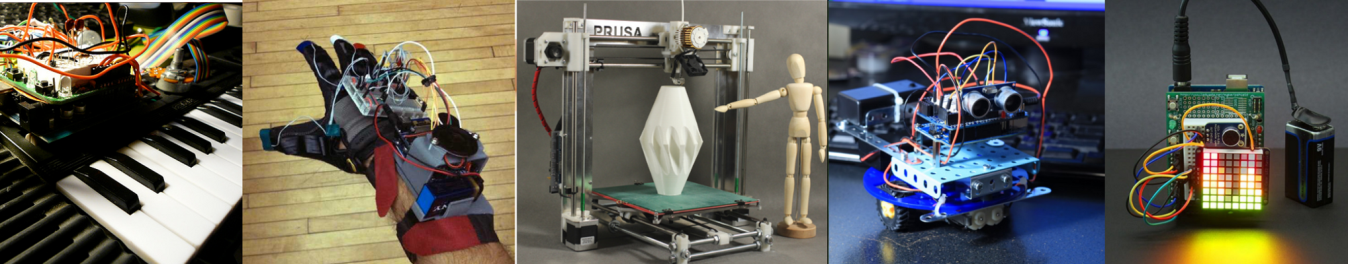
# ARDUINO.

***Antes de comenzar a experimentar con este poderoso y pequeño artefacto, se nos presenta la obvia pregunta ¿Qué es Arduino?***

***¿Qué es?***

En pocas palabras Arduino es una mini computadora de bajo costo que nos permite diseñar, construir y agregarle distintos componentes con mucha facilidad y sin necesidad de ser expertos.

Arduino es tan versátil que se puede utilizar para crear instrumentos musicales, juegos electrónicos, máquinas de pinball, arte interactiva, instrumentos de medición de todo tipo, estaciones meteorológicas, robots, casas inteligentes y en general, casi cualquier cosa que se te venga a la mente.



***¿Qué necesita para funcionar?***

Para funcionar Arduino necesita una fuente de energía (que puede ser suministrada por una batería de 9v o por el cable USB), un programa escrito en su memoria, y los componentes electrónicos y mecánicos que requiera nuestro proyecto.



***¿Qué se necesita para programar en Arduino?***

Lo único que se necesita para programar en Arduino es: un Arduino, un cable USB, y una computadora cualquiera (puede ser netbook, notebook, o una PC de escritorio). En la computadora además, se necesita instalar los drivers y el IDE Arduino, que es el programa que usaremos para crear nuestros proyectos, y es el que se encarga de cargar nuestros programas al Arduino.

***¿Cómo se le conectan los componentes?***

La placa Arduino cuenta con PINES de entrada y salida donde se conectan los cables de los componentes electrónicos. Posee 13 pines digitales y 6 analógicos, pero no te preocupes demasiado en entender que significa ahora eso, solo tené en cuenta que tenes disponibles ¡19 conexiones para expandir tu Arduino!



***¿Qué conocimientos previos necesito para seguir este cuadernillo?***

Este cuadernillo no parte de ninguna base por lo que no requiere conocimientos previos ni en programación ni en electrónica. Te guiará paso a paso en una serie de proyectos y experimentos que te ayudarán a entender y a manejarte en este apasionante y nuevo mundo de creaciones digitales y electrónicas de la mano de Arduino.

Equivocarse y volverlo a intentar es parte esencial del proceso de aprendizaje, por lo que no te preocupes si las cosas no salen bien desde la primera vez. Es muy importante mantener un espíritu curioso y lúdico en nuestros experimentos para poder disfrutarlos y sacarles el máximo provecho.

**PROYECTO 0 – *Antes de comenzar:*** *preparando el entorno.*

Antes de comenzar con el primer proyecto hay que preparar nuestro entorno de trabajo, y lo vamos a hacer en simples pasos.

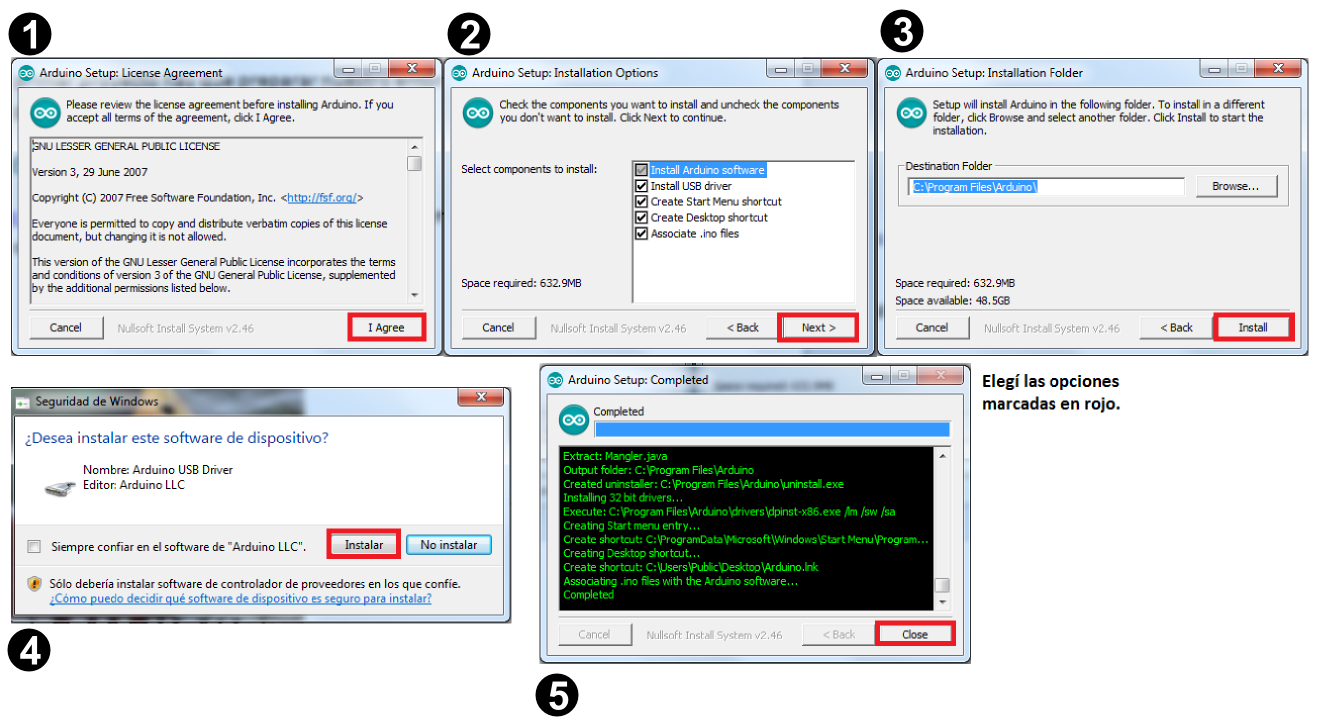
***1)- Descargá el entorno Arduino.***

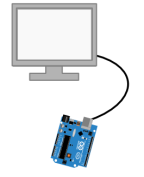
Ingresá a ésta página: <http://arduino.cc/en/Main/Software> y elegí el sistema operativo donde lo vas a utilizar. Por ejemplo si lo instalas en Windows:

******

***2) Instala el entorno Arduino.***

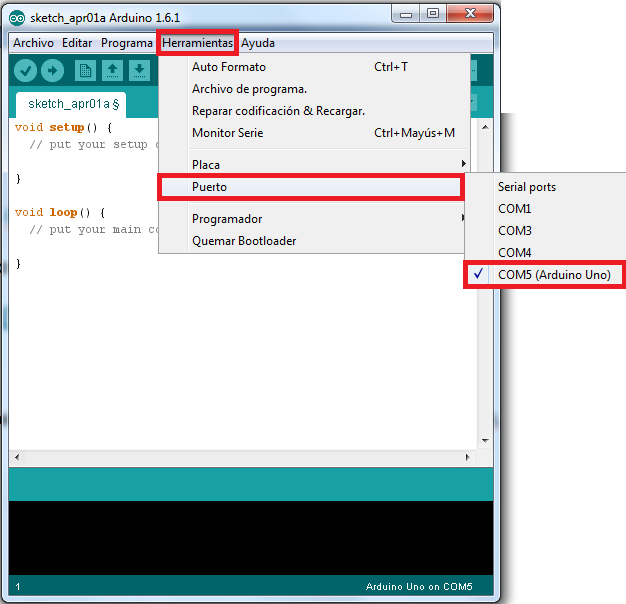
Las siguientes imágenes te van a servir para guiarte por el proceso de instalación. Es posible que en el momento que instales el IDE (Entorno de desarrollo) haya una versión nueva disponible, pero esencialmente los pasos serán los mismos.

***3. Conectá tu arduino.***

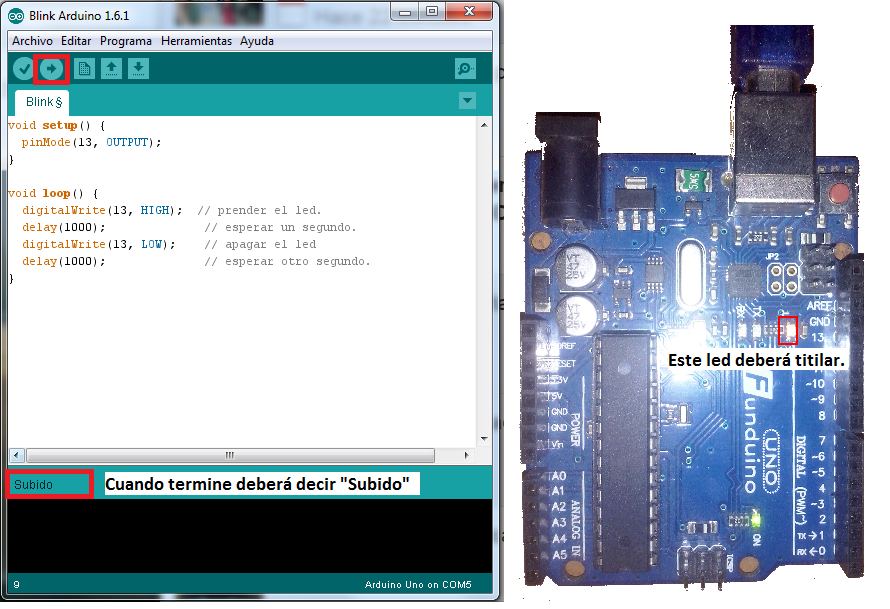
Usando el cable USB conecta tu arduino a la computadora.

***4. Ejecuta el IDE.***

Busca en el escritorio el acceso directo al IDE arduino  y dale doble click. Luego verifica que en Herramientas -> Puerto se encuentre seleccionado el puerto COM que tiene conectado el arduino. Aparece entre paréntesis Arduino Uno, como muestra la imagen.



|  |
| --- |
| Por último copia y pega este código. |
| **Código: PruebaCero** |
| void setup() {  pinMode(13, OUTPUT);  }  void loop() {  digitalWrite(13, HIGH); // prender el led.  delay(1000); // esperar un segundo.  digitalWrite(13, LOW); // apagar el led  delay(1000); // esperar otro segundo.  } |

Y cárgalo en el arduino haciendo click en el ícono como muestra la siguiente imagen.

Si todo fue bien, el led marcado en la imagen debería titilar. ¡Ahora si estás en condiciones de trabajar con arduino! Y porque no, empezar con los proyectos.

**PROYECTO 1 - *El semáforo:*** *Jugando con luces y colores.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | *Sobre este proyecto.* | | |
| Nuestro primer experimento no será nada más ni nada menos que un semáforo. Si bien no tendrá el tamaño de un semáforo real, va a funcionar exactamente igual que uno.  **Tags**: leds, digital, resistencias, semáforo, loop, setup. | | |
| *Elementos necesarios.* | | | | |
|  | **3 Leds (Rojo, amarillo, verde)**  Los leds son un tipo especial de diodo que cuando la corriente eléctrica circula por ellos emiten luz. | |  | **3 Resistencias 220Ω**  Las resistencias son un componente electrónico que actúa como una barrera y deja pasar solo cierta cantidad de electricidad (medida en Ohmios) |

|  |  |  |
| --- | --- | --- |
| Un semáforo no es más que una señal lumínica que comunica a los conductores que es lo que deberían hacer: si deberían esperar o si deberían avanzar. Para crear nuestro mini semáforo por lo tanto vamos a necesitar luces que indiquen el estado del mismo, y además algo que maneje los tiempos en que las luces van a estar prendidas. Ese algo que va a controlar las luces, va a ser nuestro pequeño programa en el Arduino, y las luces que utilizaremos van a ser leds. | | |
| **Sobre los leds.**  Conociendo un poco más |  | Al conectar los leds tendrás que tener un especial cuidado ya que las patitas del mismo deben estar correctamente conectadas tal como muestra la figura de la derecha.  La patita que corresponde a la mueca o marca que tiene el led en uno de sus lados corresponde al polo negativo y va conetado al pin ground en nuestro Arduino. |
| led.pngLos **leds** pertene-cen a un grupo de componentes electrónicos llamados diodos, que se comportan diferente dependiendo de la dirección en que circula la corriente eléctrica que los atraviesa. Por eso es importante conectarlos bien ya que se pueden dañar. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | |
| ***Esquema Semáforo versión 1.*** | | Siguiendo el siguiente esquema vamos a armar la primera prueba. Conectaremos la salida digital 7 a una resistencia de 220ohm y esta resistencia a su vez a la pata positiva del led verde. Luego desde la pata negativa (o cátodo) lo conectaremos al Arduino al pin etiquetado como ground para cerrar el circuito.  El uso de la resistencia es necesario ya que la placa arduino envía más corriente eléctrica de la que el led puede manejar (esto es si se encuentra encendido por un tiempo prolongado). | |
| Ahora sí una vez que este circuito básico está armado, estamos listos para darle las primeras instrucciones a nuestro Arduino. Para eso en el entorno Arduino crea un nuevo Sketch y escribe lo siguiente: | | |
| **Código: PruebaLeds\_01** | | |
| // PruebaLeds\_01  void setup() {  pinMode(7,OUTPUT); // le decimos a arduino que usaremos este pin como salida.  digitalWrite(7,HIGH); // enviar la señal HIGH al pin 8 (encender el led)  }  void loop() {  // por ahora no vamos a hacer nada acá.  } | | |

Compílalo cárgalo y probalo. ¿Qué es lo que pasó?

|  |  |  |
| --- | --- | --- |
|  | **Si todo fue bien** | El led se encendió y no volvió a apagarse. |
|  | **Si algo fue mal** | El led no se encendió. |
|  | **Cosas a revisar si no funcionó como debía…** | * Revisar que el conexionado sea el correcto. * Revisar que no se haya soltado ningún cable. * Revisar que el código esté correcto y no falten “;” ni llaves por cerrar. (Click en el bóton  dentro del entorno Arduino para verificar el código) * Revisar que el código haya sido cargado al Arduino y que éste se encuentre conectado al puerto USB. * Revisa que el led esté correctamente conectado (el ánodo al pin 7 y el cátod al pin Ground (Tierra)) |

¡Perfecto! No es el mejor semáforo del mundo ya que deja que todo el mundo pase sin control, pero bueno es recién nuestra primera prueba nada más. Vamos a repasar con detenimiento que es lo que sucedió:



**Sobre los pines, las señales digitales y analógicas…**

El Arduino dispone de varios pines que permiten conectar salidas y entradas digitales y salidas y entradas analógicas. Ya veremos la diferencia entre ellas más adelante, por ahora basta decir que las señales digitales solo pueden tener dos valores posibles, mientras que las analógicas pueden tener muchos valores.

Los pines digitales son los que se encuentran etiquetados del 0 al 13, cómo se ve en la figura siguiente.



Que sean digitales significa que en estos pines solo puede haber dos valores posibles, un voltaje “Alto” (HIGH) o un voltaje bajo (“LOW”). Estos pines se pueden usar tanto como para escribir, es decir para producir una salida (prender un led en nuestro caso), o bien para leer una entrada (la que puede proveer un sensor).

Para prender el led podemos utilizamos la línea de código.

|  |
| --- |
| **Fragmento de código.** |
| digitalWrite(7,HIGH); // enviar la señal HIGH al pin 8 (encender el led) |

Y para apagarlo ésta otra:

|  |
| --- |
| **Fragmento de código.** |
| digitalWrite(7,LOW); // enviar la señal LOW al pin 8 (apagar el led) |

Con esta nueva información ya estamos preparados para prender y apagar el led, para eso podríamos utilizar un código como el siguiente.

|  |
| --- |
| **Código: PruebaLeds\_02** |
| // PruebaLeds\_02  void setup() {  pinMode(7,OUTPUT); // le decimos a arduino que usaremos este pin como salida.  digitalWrite(7,HIGH); // enviar la señal HIGH al pin 8 (encender el led)  digitalWrite(7,LOW); // apagar el led  }  void loop() {  // por ahora no vamos a hacer nada acá.  } |

Compílalo cárgalo y probalo. ¿Qué es lo que pasó?

|  |  |  |
| --- | --- | --- |
|  | **Si todo fue bien** | El led se encendió e inmediatamente se apago (es probable que ni siquiera lo hayas visto encenido). |
|  | **Si algo fue mal** | El led sigue encendido. |
|  | **Cosas a revisar si no funcionó como debía…** | * Revisar que el código haya sido cargado al Arduino y que éste se encuentre conectado al puerto USB. |

Como el Arduino es tan rápido (bueno en realidad muy lento comparado con una PC de escritorio o incluso con un celular) en cuanto el Arduino escribió HIGH en el pin 7, inmediatamente escribió LOW en el mismo pin haciendo que el led se prenda por una pequeñísima fracción imperceptible de tiempo y se apague.

Lo que necesitamos hacer es que el Arduino espere un momento antes de apagar el led, para dejarnos apreciar lo que sucede. Para indicarle al Arduino que espere un tiempo antes de ejecutar la siguiente acción, podemos usar el siguiente código

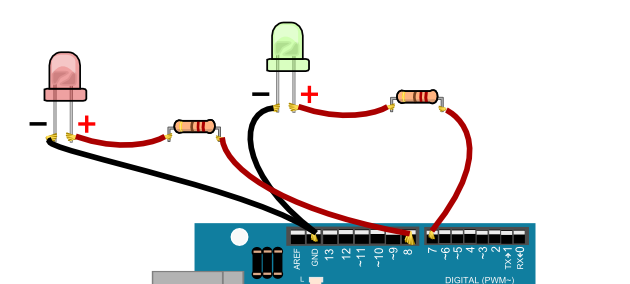
|  |
| --- |
| **Fragmento de código.** |
| delay(1000); // esperar 1000 milisegundos. |

Vamos a ver el ejemplo completo de como indicar al Arduino que deje al led encendido un tiempo para poder verlo. Copia, pega y carga este código en tu Arduino.

|  |
| --- |
| **Código: PruebaLeds\_03** |
| // PruebaLeds\_03  void setup() {  pinMode(7,OUTPUT); // le decimos a arduino que usaremos este pin como salida.  digitalWrite(7,HIGH); // enviar la señal HIGH al pin 8 (encender el led)  delay(5000); // Esperar 5 segundos  digitalWrite(7,LOW); // apagar el led  delay(1000); // Esperar 1 segundo  digitalWrite(7,HIGH); // volver a encenderlo  }  void loop() {  // por ahora no vamos a hacer nada acá.  } |

|  |  |  |
| --- | --- | --- |
|  | **Si todo fue bien** | El led se encendió, pasaron 5 segundos y se apagó. Luego paso un segundo y volvió a encenderse. |
|  | **Si algo fue mal** | El led no prendió. |
|  | **Cosas a revisar si no funcionó como debía…** | * Revisar que el código haya sido cargado al Arduino y que éste se encuentre conectado al puerto USB. |

Muy bien, y ahora sí, para completar nuestro semáforo sólo debemos agregar un led más, siguiendo el siguiente esquema:



Ahora sí tenemos dos leds para empezar a jugar. Vamos a hacer que primero prenda el verde mientras el rojo esté a pagado, y luego al revés, mirando el siguiente código:

|  |
| --- |
| **Código: PruebaLeds\_05** |
| // PruebaLeds\_05  void setup() {  pinMode(7,OUTPUT); // usamos el pin 7 como salida (led verde)  pinMode(8,OUTPUT); // usamos el pin 8 como salida (led rojo)  digitalWrite(8,HIGH); // enviar la señal HIGH para prender el led rojo  digitalWrite(7, LOW); // enviar la señal LOW para asegurarnos de que el verde este apagado.  delay(5000); // Esperar 5 segundos  digitalWrite(8,LOW); // apagamos el led rojo  digitalWrite(7, LOW); // encendemos el verde  delay(5000); // Esperar 5 segundos  // hacemos todo una vez más  digitalWrite(8,HIGH); // enviar la señal HIGH para prender el led rojo  digitalWrite(7, LOW); // enviar la señal LOW para asegurarnos de que el verde este apagado.  delay(5000); // Esperar 5 segundos  digitalWrite(8,LOW); // apagamos el led rojo  digitalWrite(7, LOW); // encendemos el verde  }  void loop() {  // por ahora no vamos a hacer nada acá.  } |

|  |  |  |
| --- | --- | --- |
|  | **Si todo fue bien** | El led verde empezó apagado, y el rojo encendido. Luego cinco segundos después, el verde se prendió y el rojo se apago. Cinco segundos después volvió a suceder lo mismo. |
|  | **Si algo fue mal** | Los leds no se prendieron en el orden correcto o no se prendieron en lo absoluto. |
|  | **Cosas a revisar si no funcionó como debía…** | * Revisar que el código haya sido cargado al Arduino y que éste se encuentre conectado al puerto USB. * Revisar el conexionado, y comprobar que ambos leds tienen conexión al pin “ground” y que el ánodo y cátodo de cada uno estén correctamente conectados. |

Ahora la pregunta que puede surgir es… ¿Cómo hacemos para que el semáforo siga funcionando sin tener que agregar el código para hacer simplemente lo mismo?

Como podes ver, todos los códigos que envíamos a arduino tienen dos estructuras, una llamada “setup” y otra llamada “void”:

|  |
| --- |
| **Fragmento de código.** |
| void setup() {}  void loop() {} |

Estas estructuras son llamadas funciones, y dentro de ellas se escribe código (luego volveremos sobre el concepto de función). Lo que escribimos dentro de la función setup() se ejectuta solo una vez cuando el Arduino se enciende. Mientras que la función loop() se ejecuta una y otra vez mientras el arduino tiene corriente.

Finalmente para que nuestro semáforo quede terminado, este sería un posible código final:

|  |
| --- |
| **Código: Semáforo** |
| // Semaforo  void setup() {  // lo que esta dentro de setup se ejecuta solo una vez.  // solo necesitamos decirle una vez al arduino como usaremos los pines 7 y 8  pinMode(7,OUTPUT); // verde  pinMode(8,OUTPUT); // rojo  // comenzamos con ambos apagados.  digitalWrite(7,LOW); //  digitalWrite(8, LOW); //  }  void loop() {  // este codigo se va a ejecutar una y otra vez  digitalWrite(7,LOW); // apagamos el verde  digitalWrite(8,HIGH); // encendemos el rojo  delay(2000); // esperamos 2 segundos  digitalWrite(7,HIGH); // encendemos el verde  digitalWrite(8,LOW); // apagamos el rojo  delay(2000); // esperamos 2 segundos  // una vez que termina la fucnión loop, se vuelve a ejecutar desde el comienzo.  } |

|  |  |  |
| --- | --- | --- |
|  | **Si todo fue bien** | Nuestro semáforo funciona. Primero se enciende el rojo, luego se apaga y se prende el verde, y ¡Así para siempre! (o hasta que le cortes la corriente) |
|  | **Si algo fue mal** | Los leds no se prendieron en el orden correcto o no se prendieron en lo absoluto. |
|  | **Cosas a revisar si no funcionó como debía…** | * Revisar que el código haya sido cargado al Arduino y que éste se encuentre conectado al puerto USB. * Revisar el conexionado, y comprobar que ambos leds tienen conexión al pin “ground” y que el ánodo y cátodo de cada uno estén correctamente conectados. |

Vamos a repasar que es lo que sucedió cuando cargamos este código al Arduino.



¡Muy bien ahora si hemos terminado con el semáforo! Antes de terminar de dar por terminado este proyecto a continuación hay una serie de experimentos que podes probar.

|  |  |
| --- | --- |
| *¡Hora de experimentos y ejercicios!* | |
| **Ahora que ya sabes un poco más sobre Arduino porque no intentás…** | |
|  | 1- Has que el semáforo pase más tiempo en rojo que en verde. |
|  | 2- Has que el led verde empiece a titilar antes de apagarse y encender el rojo. Tal como los semáforos actuales. |
|  | 3- Agrega un led amarillo que se encienda entre medio del rojo y el verde. Ahora sí estaría completo. |
|  | 4- ¿Se te ocurre como hacer para que el semáforo al encenderse encienda los tres led por unos segundos y luego funcione normalmente? (*pista*: recordá la diferencia entre setup y loop) |