

# Taller Básico de Arduino (2015)

## Presentación del Taller Básico de Arduino



Arduino es una plataforma de prototipos de código abierto basado en hardware fácil de usar mediante el software. Las Placas Arduino son capaces de leer los valores del ambiente mediante sensores -un dedo en un botón, o un mensaje de Twitter - y lo convierten en una salida - la activación de un motor, encender un LED, publicar algo en línea. Usted puede decirle a su tablero qué hacer mediante el envío de un conjunto de instrucciones para el microcontrolador en el tablero. Para ello se utiliza el lenguaje de programación de Arduino (basado en Wiring), y el software de Arduino (IDE), basado en Processing.

## Organización de Equipos



Para aprovechar al máximo el taller de Arduino es recomendable que se organicen equipos de dos integrantes, de manera que cada uno pueda realizar los roles que se le asignará.

## Reglas y Normas del Taller



- Evitar introducir alimentos o bebidas al taller.
- Escuchar con atención las indicaciones de los instructores.
- Apoyar al compañero@ del equipo para realizar las actividades asignadas.
- Respeto en general al resto de los participantes e instructores.
- Preguntar si existe alguna duda con respecto a alguna conexión o componente a usar.
- Precaución al conectar y desconectar aparatos en las extensiones o multicontactos.

## Instalación de IDE de Arduino

La fundación Arduino tiene desarrollado su propio IDE para las diferentes plataformas más utilizadas (Linux, OSX y Windows), por ello se deberá descargar la versión de la plataforma en la que se vaya a trabajar con la placa desde la siguiente página <https://www.arduino.cc/en/Main/Software>



En este taller nos centraremos en la instalación del IDE para Windows por lo que se recomienda tener una cuenta con los permisos de administrador, para poder realizar la instalación de manera habitual (el famoso Next, Next). Finalizada la instalación procedemos a abrir el nuevo programa instalado y para ello se explicará brevemente las opciones elementales que conforman el entorno de desarrollo.

La siguiente lista menciona de manera sencilla los elementos fundamentales para usar el IDE de Arduino y se pueden identificar en la Imagen 1.

1. **Compilar:** Compila y aprueba su código. Detectará errores en la sintaxis (como la falta de punto y coma o paréntesis).
2. **Subir:** Envía tu código al Microcontrolador (en nuestro caso, placa Arduino). Al hacer clic en el botón, deberá ver las luces de su placa parpadear rápidamente.
3. **Nuevo:** Abre una nueva pestaña de la ventana de código.
4. **Abrir:** Permitirá abrir un sketch (un archivo de código, con extensión ".ino") existente.
5. **Guardar:** Esto guardará el sketch activo.
6. **Serial Monitor:** Se abrirá una ventana que muestra toda la información que se transmite de la placa, al puerto serial de la computadora al que está conectada y viceversa... Es muy útil para la depuración.
7. **Sketch Nombre:** Muestra el nombre del archivo que se está trabajando.
8. **Área de Código:** Esta es la zona en la que se redacta el código de su sketch.

9. **Área de mensajes:** Aquí es donde el IDE, si hubo errores, presenta un aviso de posibles errores en el código.
10. **Consola de Texto:** Área de mensajes de error completos. Al depurar, la consola de texto es muy útil.
11. **Placa y Puerto Serial:** Muestra la placa en uso y las selecciones de puerto serie.

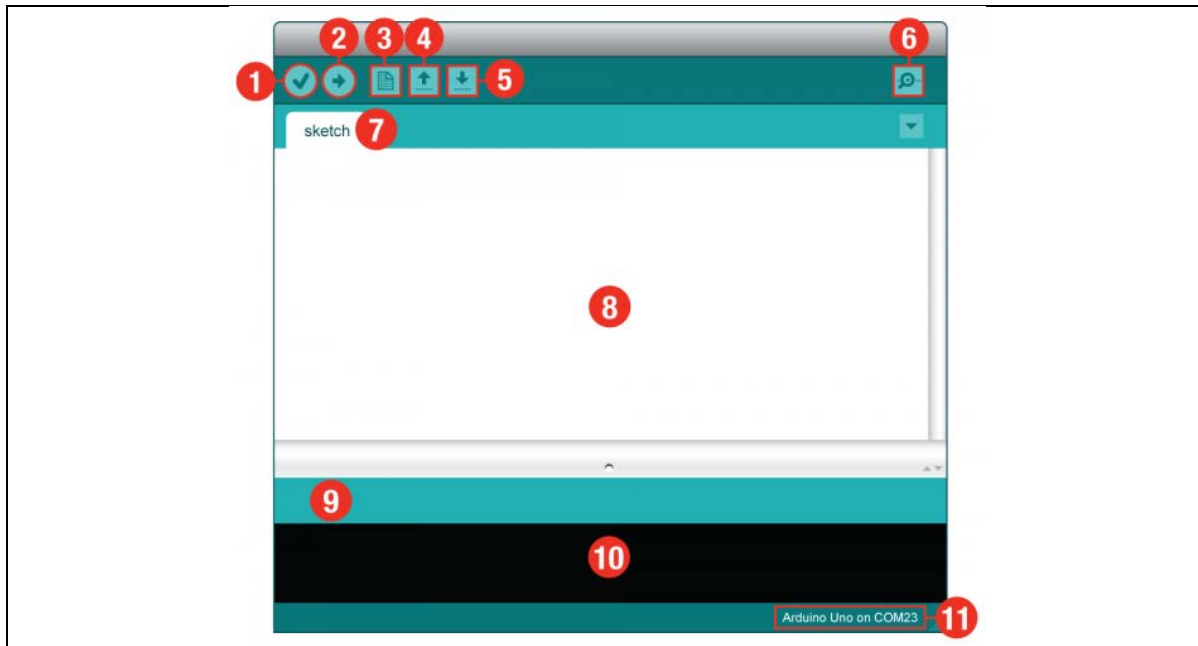


Imagen 1.- IDE Arduino

Fuente.- <https://learn.sparkfun.com/tutorials/sik-experiment-guide-for-arduino---v32>

### Explicación de Arduino

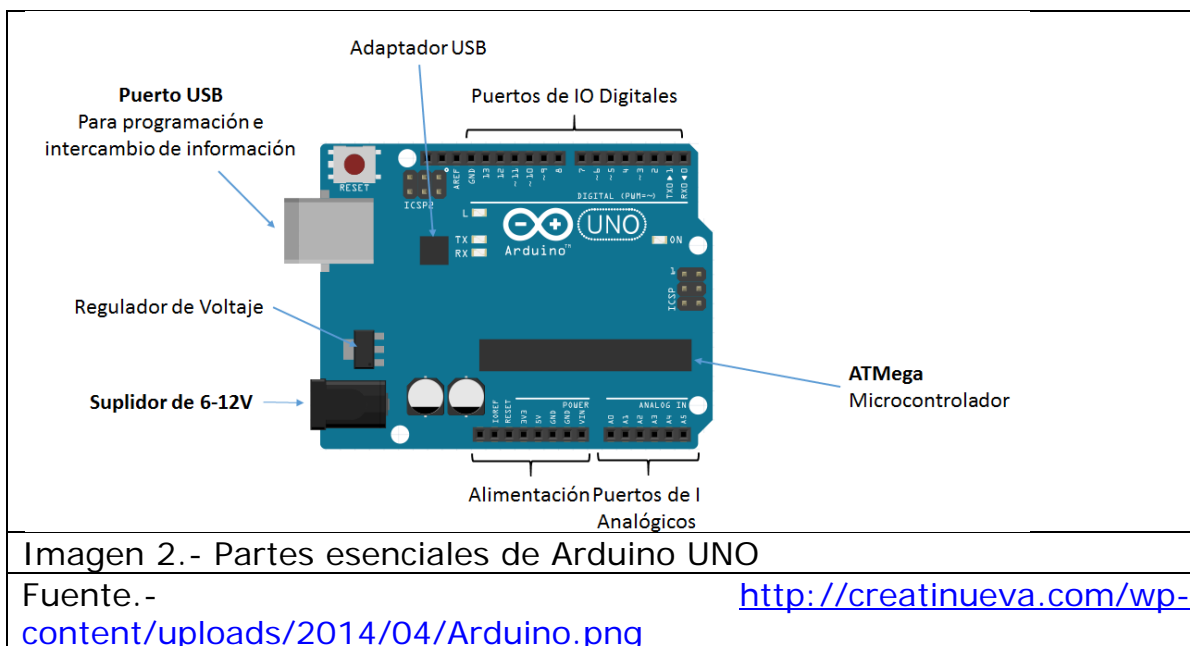
Arduino es una plataforma de prototipos electrónica de código abierto (open – source) basada en hardware y software flexibles y fáciles de usar. Está pensado e inspirado en artistas, diseñadores, y estudiantes de computación o robótica y para cualquier interesado en crear objetos o entornos interactivo, o simplemente por hobby.

Requiere de un lenguaje de programación para poder ser utilizado y, como su nombre lo dice, programado y configurarlo a nuestra necesidad, por lo que se puede decir que Arduino es una herramienta "completa" en cuanto a las herramientas principales nos referimos, ya que sólo debemos instalar y configurar con el lenguaje de programación de esta placa los componentes eléctricos que queramos para realizar el proyecto que tenemos en mente, haciéndola una herramienta no sólo de creación, sino también de aprendizaje en el ámbito del diseño de sistemas electrónicos-automáticos y, además, fácil de utilizar.

Las posibilidades de realizar proyectos basados en esta plataforma tienen como límite la imaginación de quien opera esta herramienta.

## Sus partes Esenciales

En la Imagen 2 se muestra de manera muy general las partes que conforman a una placa de Arduino.



## Voltaje de entrada, salida formas de suministrar energía

La mayor parte de los Arduinos funcionan a 5V. Sea cual sea el modelo, todos tienen un regulador de tensión, que básicamente es un componente que convierte el voltaje que con el que se alimenta la placa (lo recomendado es entre 7 y 12V) a 5V (o 3.3V), *tirando por tierra* el resto (en realidad no lo tira por tierra, es peor aun, **ese voltaje calienta tu placa Arduino**. Por ejemplo, alimentar Arduino con más de 20V lo destruiría).

Las tres mejores formas de alimentar el arduino van a depender del uso que se le vaya a dar, puede ser con el adaptador de corriente que se usa comúnmente para los celulares (Imagen 3), con 4 pilas AA que acumulen 6v (Imagen 4) o con pila de LiPo (Imagen 5).

De igual manera tendrás que analizar si el proyecto que estés desarrollando va a necesitar más energía, para evitar dejar al Arduino sin el suministro adecuado. Para ello se requiere de un poco más de conocimientos de Electrónica.



Imagen 3.- Adaptador de corriente para celulares.

Fuente.-

<http://miniimg.rightinthebox.com/images/384x384/201311/uvxwxx1385604572269.jpg>

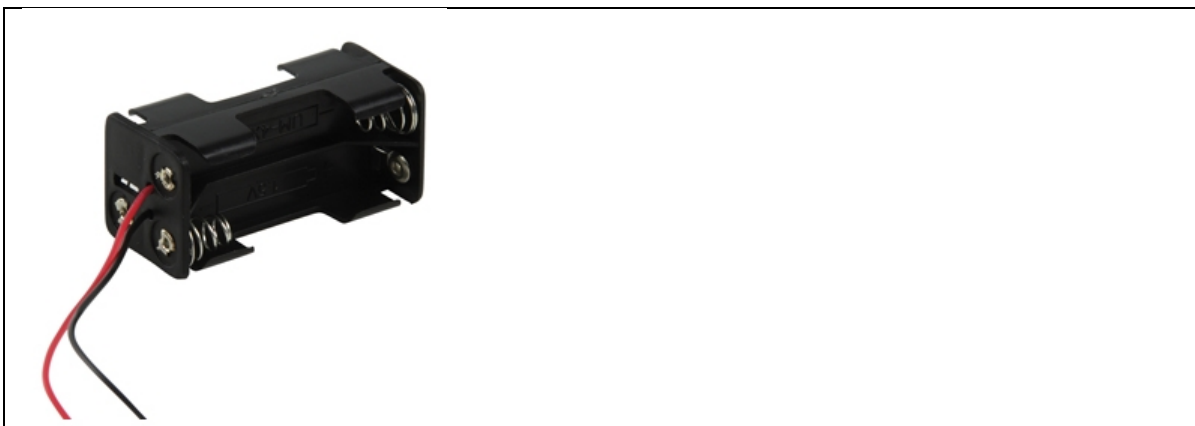


Imagen 4.- Pilas AA (acumulando 6v)

Fuente.-

<http://mecatronica digital.com/mystore/item/133/bb/porta-pilas-4xaa-cubo>



Imagen 5.- Pilas de LiPo

Fuente.-

<http://www.educachip.com/wp-content/uploads/LiPo.jpg>

### **Breve concepto de Voltaje, Intensidad y Resistencia**

De manera muy formal se puede decir que Voltaje es una magnitud física que impulsa a los electrones a lo largo de un conductor en un circuito cerrado, es decir, que los electrones caminan por cualquier material que les permita ir de un punto a otro, su unidad es el **Voltaje**.

Con respecto a la Intensidad de corriente se refiere a la cantidad de electrones que pasa por un conductor en la unidad de tiempo, su unidad es el **Amperio**.

La resistencia eléctrica es la oposición que presenta un cuerpo al paso de una corriente eléctrica para circular a través de él, su unidad es **Ohm**.

Los conceptos anteriores se refieren a la Ley de Ohm y su forma de cálculo es muy sencilla mediante el triángulo que se muestra en la Imagen 6.

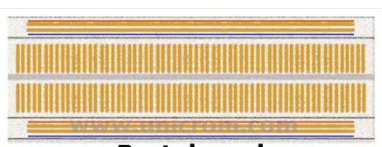


Imagen 6.- Ley de Ohm

Fuente.- <http://html.rincondelvago.com/000421003.jpg>

### Componentes básicos a usar

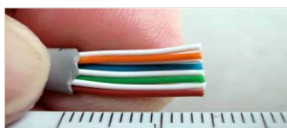
En este taller se verán solo alguno de los tantos componentes que existen para uso de la electrónica, la Imagen 7 muestra el componente y su nombre.



**Protoboard**



**LED**



**Cables para conexión  
(se puede usar los de  
Ethernet)**



**Resistor**



**Arduino Uno (con cable  
USB)**

Imagen 7.- Componentes a usar

Fuente.- Propia

## Uso del Protoboard

Tal como se ha mencionado, la protoboard es una placa que puede usarse para realizar los primeros experimentos con la electrónica, para ello se debe saber la forma en la que circulan los electrones. La Imagen 8 muestra la manera en que fluyen en los polos positivo (líneas Rojas), polos negativos (líneas Negras), en el centro existe un conjunto de orificios que funcionan de manera vertical, es decir que la electricidad se transmitirá solo en ese segmento de columna.

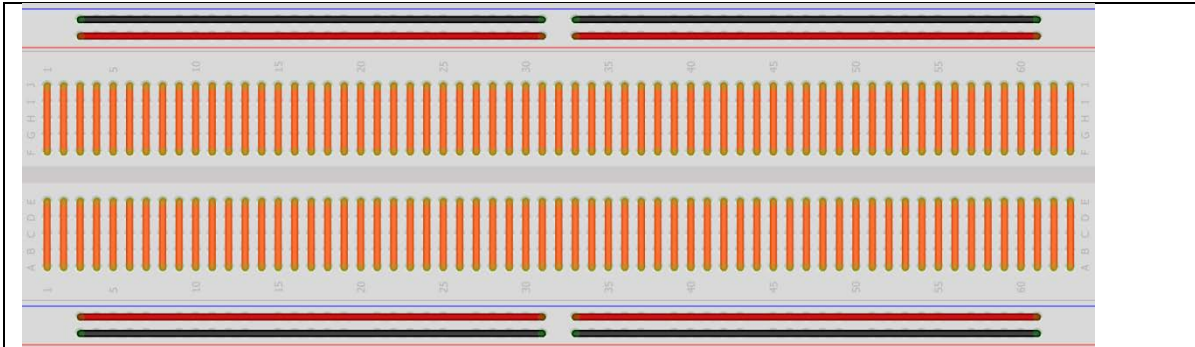


Imagen 8.- Funcionamiento del Protoboard

Fuente.- <http://compututorials.blogspot.mx/2011/08/como-funciona-el-protoboard.html>

## Conexión de LED, resistencia, cableado, Positivo y Negativo

### *Conectando el primer led (Práctica 1 [P2])*

Usar el protoboard es muy sencillo, para poder hacer un primer acercamiento se debe observar las conexiones que se muestran en la Imagen 9 y hacer la práctica previa con el protoboard.



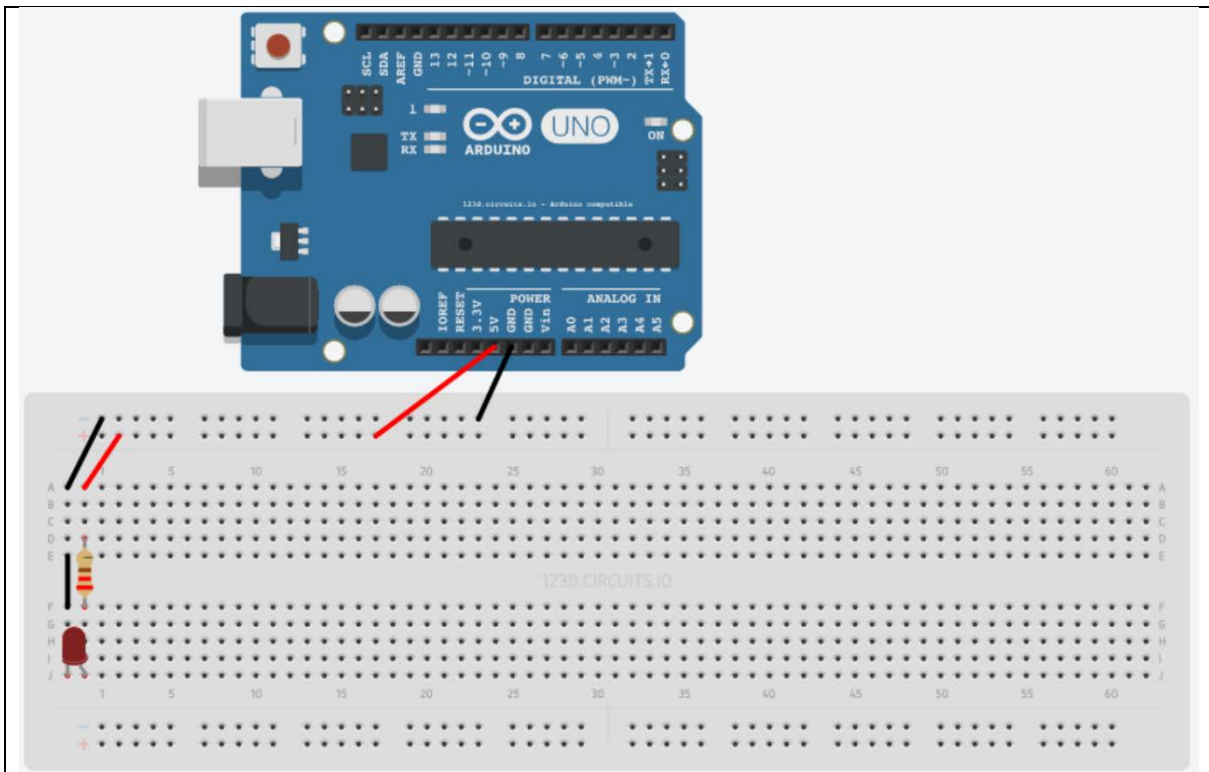


Imagen 9.- Primer acercamiento con el protoboard

Fuente.- Diseñado con 123d.circuits.io

## Conexión de Arduino a la Computadora

Anteriormente se mencionó de manera muy breve que se puede conectar el Arduino a la computadora, para ello solo hazlo tal como se muestra en la Imagen 10.

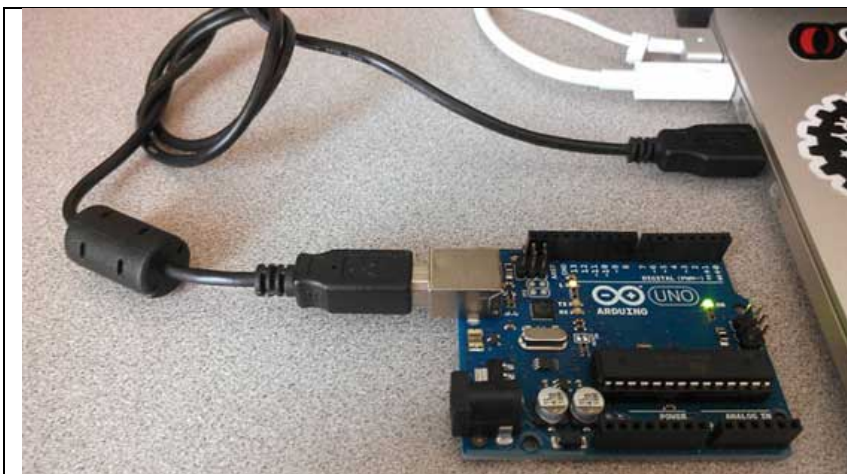
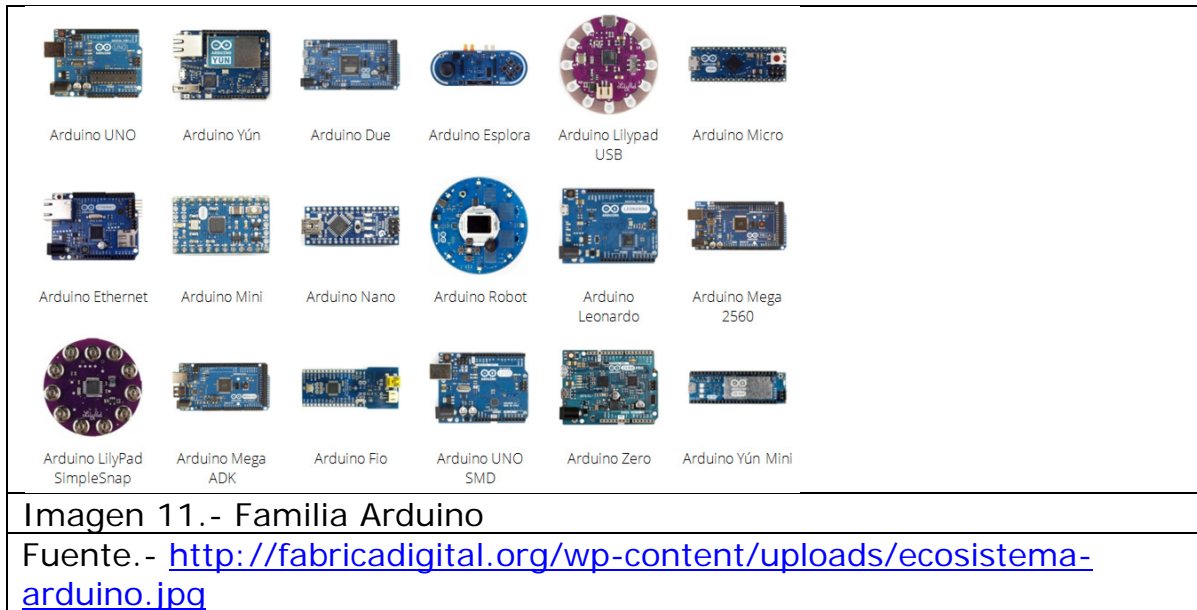


Imagen 10.- Conexión de Arduino a la Computadora usando cable USB

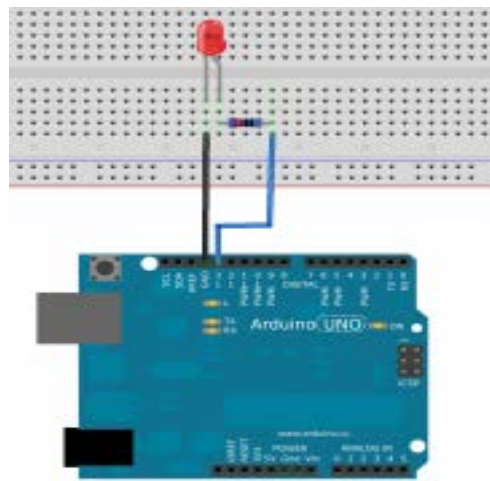
Fuente.- [http://blog.bricogeek.com/img\\_cms/2142-arduino-javascript-nodejs-firmata.jpg](http://blog.bricogeek.com/img_cms/2142-arduino-javascript-nodejs-firmata.jpg)



Hasta el momento se ha visto lo básico y muy esencial requerido para comprender el uso de una placa de Arduino, existen otras placas de Arduino y que están orientadas a un uso en especial o dependiendo de lo que se requiera hacer, la Imagen 11 se muestra parte de las placas que conforman la familia de Arduino.



### *Parpadeo de LED (Práctica 2 [P2])*



**El código: su compilación, respaldo en computadora y almacenado en Arduino.**

Antes de comenzar es necesario que se tenga abierto el IDE de Arduino, después conectar el Arduino a la computadora, hacer clic en la opción

herramientas (tools) del IDE y seleccionar la placa que se usará (en este caso es la UNO R3), además de el puerto por el cual se comunicará el IDE con la placa (en Windows dice COM X, la X varia de acuerdo al número que asigne el S.O., ejemplo COM1, COM2)

## Compilación


Todo código que se quiera probar en la placa, se recomienda que primero se compile, el compilador se encarga de verificar los errores de sintaxis que existan en todo el código, para ello se debe hacer clic en el

botón de una Paloma .

## Respaldo en la computadora

Para guardar el sketch o código se debe hacer clic en la opción Archivo y luego en Guardar Como..., ahí se podrá seleccionar la ruta que se desee. Se recomienda que una vez guardado el código se utilice la combinación de teclas Ctrl + S.

## Almacenado en Arduino

Después de Compilar y guardar los cambios realizados, se puede realizar la subida del código a la placa, para ello se hace clic en el botón que tiene como ícono una flecha .

## Destreza P1: Armado y Desarmado (2 rondas)

Para el primer reto se debe tener los componentes que se indican a continuación: 1 LED, 1 Protoboard, 4 Cables de conexión, 1 Arduino UNO con su cable USB.

Posteriormente se debe copiar el siguiente código en un nuevo Sketch del IDE Arduino.

```
/*  
  Parpadeo  
  Enciende un LED por un segundo y después apaga el mismo, así repetidamente.  
  */  
// Pin 13 tiene un LED conectado.  
// asigna a la variable led el valor 13  
int led = 13;  
// la rutina de setup corre una vez o cuando se presiona reset  
void setup() {
```

```

// inicializa el pin 13 como pin de salida
pinMode(led, OUTPUT);
}
// la rutina loop corre constantemente
void loop() {
  digitalWrite(led, HIGH); // enciende el LED (HIGH es el nivel de voltaje)
  delay(1000);             // espera un segundo
  digitalWrite(led, LOW);  // apaga el LED poniendo el voltaje a LOW
  delay(1000);             // espera por un segundo
}

```

Una vez terminado de copiar el código se realiza la compilación, guardado del código en la computadora y la subida del código a la placa Arduino.

Terminado el proceso anterior, se debe desconectar la Placa de Arduino y cada integrante deberá realizar la conexión que se muestra en la Imagen 12, en la que para verificar que esté correcta la conexión se debe conectar el Arduino a la Computadora.

Para lo anterior un integrante mencionará el componente que requiere conectar y el otro lo proporcionará, para ello cada uno tendrá un máximo de 90 segundos. Después de que cada integrante haya realizado la conexión al menos una vez se hará la primera competencia de conexión desde cero.

### ***Competencia 1:***

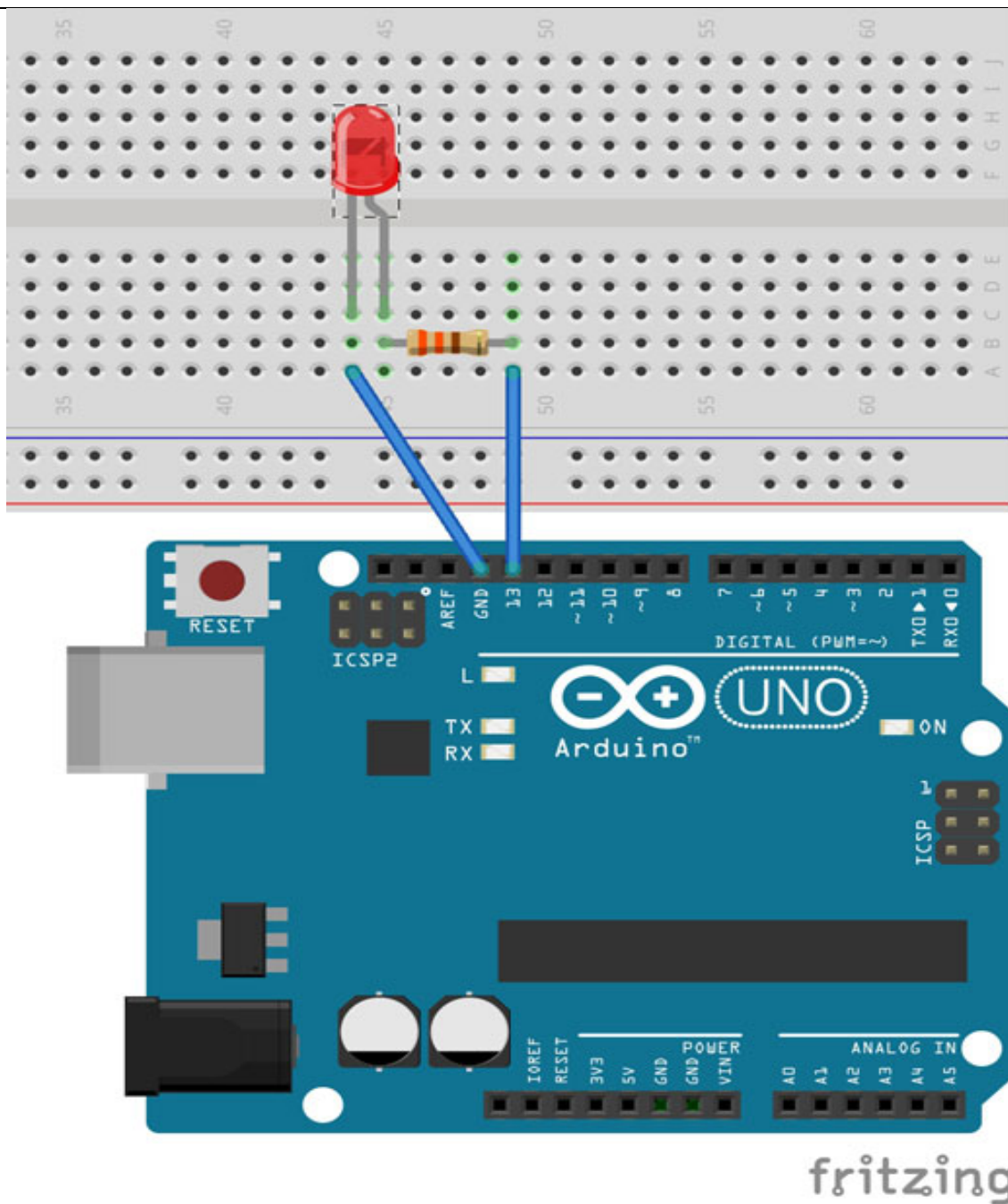


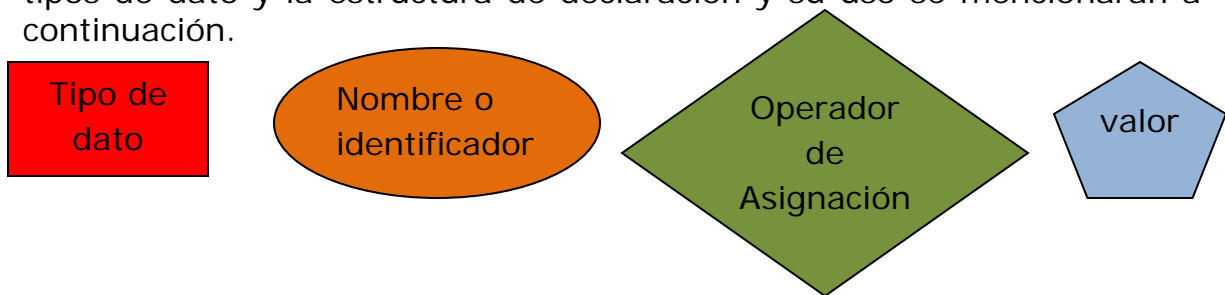
Imagen 12.- Parpadeo de LED (Práctica 1)

Fuente.- <http://princetronics.com/wp-content/uploads/2014/01/led-diagram-arduino.jpg>

## Explicación de Tipos de dato (int, float, char, string, boole) y su declaración (constantes y variables)

Así como en cualquier lenguaje humano, existen reglas para poder redactar y escribir correctamente una frase u oración, de la misma manera en los lenguajes de programación existe lo que se llama Sintaxis. Ello a grandes rasgos permite que la computadora "comprenda" las instrucciones que se requiera brindar

De la misma manera los humanos manejan Números, Letra, Letras. En los lenguajes de programación existe lo mismo pero se les conoce como tipos de dato y la estructura de declaración y su uso se mencionarán a continuación.



int solo es para números enteros, ejemplo: `int ledRojo = 1;`

float número con punto decimal, ejemplo: `float distSens1 = 12.36;`

char SOLO UN CARÁCTER y su valor debe ir entre comillas simples, ejemplo: `char vocal1 = 'A';`

string secuencia de caracteres y su valor debe ir entre comillas dobles, ejemplo: `string mensaje1 = "Led Rojo Encendido";`

boole De acuerdo al lenguaje puede ser 0 ó 1, TRUE o FALSE, ejemplo: `boole encendido = TRUE;`

## Estructura del código (cabeceras, variables, constantes, void setup(), void loop())

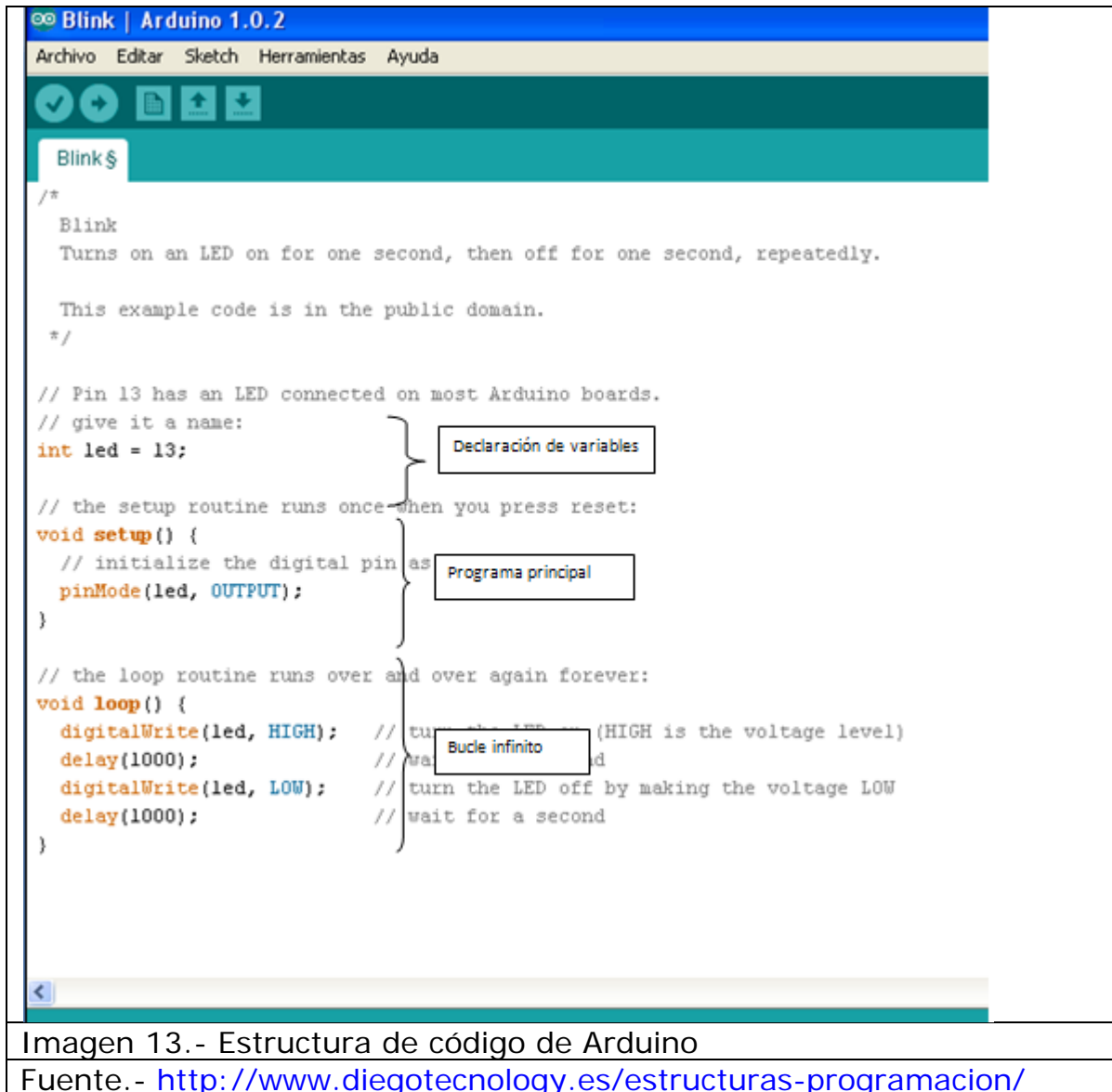
La Imagen 13, muestra la estructura básica de un programa para Arduino.

Las cabeceras son los archivos externos llamados librerías, que son otros archivos que tienen código para realizar alguna tarea en específico.

Las variables son las declaraciones cuyo valor pueden variar en cualquier momento, las constantes son declaraciones cuyo valor NO SE PUEDE MODIFICAR.

La estructura void Setup() es la parte del código donde se realiza la forma en la que inicializará el programa y funciones que se guarden en el Arduino.

La estructura void loop() es la parte del código que realiza una o más acciones de manera infinita.



### Acerca de la función delay()

La función delay() sirve para realizar una pausa en milisegundos entre un fragmento de código y otro, 1000 milisegundos equivalen a 1 segundo. Ejemplo: accion1; delay(1000); acción2;



## Acerca de las sentencias if y switch

Gran parte de las instrucciones que la computadora necesita para realizar alguna tarea, se basan en la forma en que se comunican los humanos, de igual manera están las oraciones que limitan una respuesta de acuerdo a los valores que se proporcionen.

Como ejemplo está la sentencia **if**, la cual evalúa entre uno o más valores para ejecutar alguna acción. Imagina la siguiente frase:

Si (llueve)

```
{  
    Uso Impermeable;  
}
```

Algo similar podemos mencionar a la computadora para que evalúe instrucciones y valores que se le envíen. La estructura es muy similar a la anterior

if(valor a evaluar)

```
{  
    Acción a realizar;  
}
```

Muy similar funciona la estructura **switch**, que de igual manera evalúa el valor y de acuerdo al valor ejecuta la acción correspondiente. Imagina lo siguiente:

Si te preguntan la estación del año en la que se encuentra una persona dirá una respuesta y puede ser alguna de las siguientes: Primavera, Verano, Otoño e Invierno, es decir son un número limitado de respuestas que se puede dar. En código sería así:

```
string respuesta = "Verano";
```

```
switch(respuesta)
```

```
{  
    case "Primavera":  
        Acción01;  
        Break;
```

```
case "Verano":  
    Acción02;  
    Break;  
case "Otoño":  
    Acción03;  
    Break;  
case "Invierno":  
    Acción04;  
    Break;  
}
```

La diferencia entre cuando usar **if** y cuando usar **switch**, recae en saber si existen un número limitado de opciones o no, ***si la cantidad de opciones es limitada entonces se puede usar switch, si no se sabe entonces conviene usar if.***

Con Arduino es muy sencillo utilizar esas estructuras de control para hacer que se encienda o apague lo que se desee, siempre y cuando esté conectado al Arduino y esté programado en su código.

Es posible hacer que al ingresar una letra se encienda un LED en específico o se apague, o se enciendan muchos LED o se apaguen muchos LED.

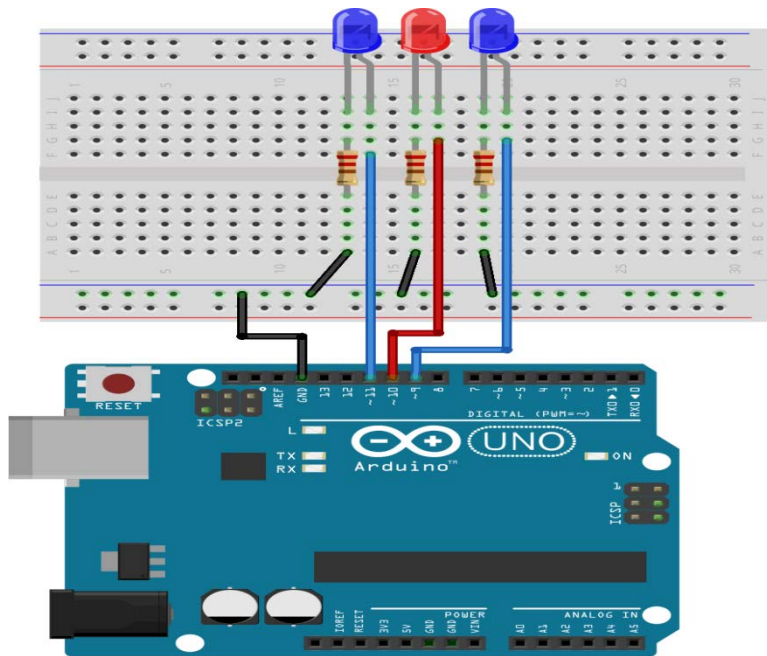
Debido a lo anterior es necesario enfatizar que la computadora SÍ DISINGUE ENTRE MAYÚSCULAS y minúsculas, es decir, para la computadora **Uno** es TOTALMENTE DIFERENTE DE **uno**.

Eso quiere decir que si tiene programado que al ingresar la letra **A** realice alguna acción y se ingresa la letra **a**, es muy posible que haga nada, claro a menos que se tenga programado que también con ese valor haga algo el Arduino.

Se debe ser lo más claro y específico posible cuando se ingresen las instrucciones a utilizar con Arduino.

Con Arduino se puede utilizar el monitor serial, que es una ventana desde la que se puede interactuar ingresando alguna letra o número y programarlo para que realice alguna acción.

## Encender LEDS con Software (Práctica 3 [P3])



**El código: explicación.**

### Destreza P2: Armado y Desarmado (2 rondas)

Para el segundo reto se debe tener los componentes que se indican a continuación: 3 LED, 1 Protoboard, 8 Cables de conexión, 1 Arduino UNO con su cable USB.

Posteriormente se debe copiar el siguiente código en un nuevo Sketch del IDE Arduino.

```
int led1 = 9;
int led2 = 10;
int led3 = 11;
char letra = 'x';

// la rutina de setup corre una vez o cuando se presiona reset
void setup() {
  // inicializa los pines 9, 10 y 11, como pines de salida.
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  digitalWrite(led1, LOW); //inicializa en apagado el led.
  digitalWrite(led2, LOW); //inicializa en apagado el led.
  digitalWrite(led3, LOW); //inicializa en apagado el led.
  Serial.begin(9600);
}
// la rutina loop corre constantemente
void loop() {
  if(Serial.available()){
```

```

letra = Serial.read();
if(letra == 'x'){
  delay(1000); // solo hace una espera de 1 segundo.
}
if(letra == 'A'){
  digitalWrite(led1, HIGH); // enciende el LED (HIGH es el nivel de voltaje)
  digitalWrite(led2, LOW); // Apaga el led 2
  digitalWrite(led3, LOW); // Apaga el led 3
  delay(1000);
}
if(letra == 'B'){
  digitalWrite(led2, HIGH); // enciende el LED (HIGH es el nivel de voltaje)
  digitalWrite(led1, LOW); // Apaga el led 1
  digitalWrite(led3, LOW); // Apaga el led 3
  delay(1000);
}
if(letra == 'C'){
  digitalWrite(led3, HIGH); // enciende el LED (HIGH es el nivel de voltaje)
  digitalWrite(led1, LOW); // Apaga el led 1
  digitalWrite(led2, LOW); // Apaga el led 2
  delay(1000);
}
}
}

```

Una vez terminado de copiar el código se realiza la compilación, guardado del código en la computadora y la subida del código a la placa Arduino.

Terminado el proceso anterior, se debe desconectar la Placa de Arduino y cada integrante deberá realizar la conexión que se muestra en la Imagen 14, en la que para verificar que esté correcta la conexión se debe conectar el Arduino a la Computadora.

Para lo anterior un integrante mencionará el componente que requiere conectar y el otro lo proporcionará, para ello cada uno tendrá un máximo de 120 segundos. Después de que cada integrante haya realizado la conexión al menos una vez se hará la primera competencia de conexión desde cero.

### ***Competencia 2: Encender LEDS con Software (todo desde cero)***



Fuente.- <http://www.arduino.utfsm.cl/wp-content/uploads/2014/05/FritzingImage2.png>

## *Competencias*

Las competencias 1 y 2, tienen dos rondas: en la primera, un participante será hardware y el otro software, en la segunda intercambian roles.

Las competencias 3 y 4 son de una sola ronda.

Se asignan puntajes en cada ronda, a los cuatro primeros equipos en concluir el reto.

Los puntajes se asignan según la siguiente tabla.

1er lugar: 40.

2do lugar: 30.

3er lugar: 20.

4to lugar: 10.

Al finalizar la ronda de cada competencia, ambos integrantes del equipo levantan las manos en señal de que han concluido, para saber el orden en el que van concluyendo.

### Competencia 1:

Realizar en el menor tiempo lo siguiente:

1. Desarmar el circuito "Parpadeo de led", reunir todos los componentes y entregárselo al de software.
2. El de hardware debe armar nuevamente el circuito, bajo las siguientes condiciones: debe solicitar cada componente al de software de uno en uno para realizar el ensamblado.
3. Una vez finalizado el ensamblado del circuito, debe conectarlo a la computadora para tener la fuente de alimentación.
4. Finalmente, ambos integrantes del equipo levantan las manos en señal de que han concluido.

### Competencia 2:

Realizar en el menor tiempo lo siguiente:

1. Desarmar el circuito "Encender LEDS con Software", reunir todos los componentes y entregárselo al de software.



2. El de hardware debe armar nuevamente el circuito, bajo las siguientes condiciones: debe solicitar cada componente al de software de uno en uno para realizar el ensamblado.
3. Una vez finalizado el ensamblado del circuito, debe conectarlo a la computadora para tener la fuente de alimentación.

Finalmente, ambos integrantes del equipo levantan las manos en señal de que han concluido.

### Competencia 3:

Realizar en el menor tiempo lo siguiente:

1. Ensamblar el circuito "Encender LEDS con Software".
2. Se utilizará el código de la práctica "Encender LEDS con Software".
3. Al ingresar la letra mencionada en cada inciso, realizar la funcionalidad que corresponda:
  - a) "D" se apagan todos los leds.
  - b) "E" encienden los tres leds.
  - c) "F" el led1 permanece apagado, led2 permanece encendido y led3 parpadea tres veces con intervalos de 2 segundos, al finalizar el último parpadeo se apagan todos.

### Competencia 4:

Realizar en el menor tiempo lo siguiente:

1. Ensamblar el circuito "Encender LEDS con Software".
2. Se les proporciona una pieza de cartón para dibujar un rostro, hacer las perforaciones necesarias al cartón para ingresar el led, de tal manera que cada leds ilumine una parte del rostro.
3. Se les da 20 minutos como máximo para esta actividad.
4. Ganan esta competencia, los 4 equipos que realicen más funciones de gestos.