**Nombre :** Luis Alvarado Choque

**Carrera :** Ingenieria en Sistemas

**Nombre Docente :** Elias Eli Alvarez **Asignatura :** Sistemas de Control

**LABORATORIO 1**

**MATERIALES**

PC previa instalación del IDE de ARDUINO UNO

Cable USB

8 Resistencias de 220

8 Leds.

Dipswicht

Cables de conexión.

Pinzas y alicates

Protoboard

**EJERCICIO 1**

int pin;

void setup() {

for(pin=0; pin<=8; pin++){

pinMode(pin,OUTPUT);

}

}

void loop() {

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(100);

digitalWrite(i, LOW);

delay(100);

}

}

**EJERCICIO 2**

int pin;

void setup() {

for(pin=0; pin<=8; pin++){

pinMode(pin,OUTPUT);

}

}

void loop() {

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(500);

digitalWrite(i, LOW);

delay(500);

}

for(int j=7;j>=0;j--){

digitalWrite(j, HIGH);

delay(100);

digitalWrite(j, LOW);

delay(100);

}

}

**EJERCICIO 3**

void setup() {

pinMode(0, OUTPUT);

pinMode(1, OUTPUT);

pinMode(2, OUTPUT);

pinMode(3, OUTPUT);

pinMode(4, OUTPUT);

pinMode(5, OUTPUT);

pinMode(6, OUTPUT);

pinMode(7, OUTPUT);

}

void loop(){

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(100);

}

for(int y=7;y>=0;y--){

digitalWrite(y, LOW);

delay(100);

}

**}**

**EJERCICIO 4**

intcont = 1;

int pin;

intlimite = 7;

void setup() {

for(pin=0; pin<=7; pin++){

pinMode(pin,OUTPUT);

}

}

void loop() {

for(pin=0; pin<cont; pin++){

digitalWrite(pin,HIGH);

delay(500);

}

for(pin=cont; pin>=0; pin--){

digitalWrite(pin,LOW);

delay(500);

if(cont>limite){

for(pin=limite; pin>=0; pin--){

digitalWrite(pin,LOW);

delay(500);

}

cont = 1;

}

}

cont = cont+1;

}

**EJERCICIO 5**

void setup() {

pinMode(0, OUTPUT);

pinMode(1, OUTPUT);

pinMode(2, OUTPUT);

pinMode(3, OUTPUT);

pinMode(4, OUTPUT);

pinMode(5, OUTPUT);

pinMode(6, OUTPUT);

pinMode(7, OUTPUT);

pinMode(8, INPUT);

pinMode(9, INPUT);

}

void loop(){

int a = digitalRead(8);

int b = digitalRead(9);

int c=2\*a+b;

switch(c){

case 0:

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(100);

digitalWrite(i, LOW);

delay(100);

}

break;

case 1:

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(100);

digitalWrite(i, LOW);

delay(100);

}

for(int j=7;j>=0;j--){

digitalWrite(j, HIGH);

delay(100);

digitalWrite(j, LOW);

delay(100);

}

break;

case 2:

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(100);

}

for(int j=0;j<=7;j++){

digitalWrite(j, LOW);

delay(100);

}

break;

case 3:

for(inti=0;i<=7;i++){

digitalWrite(i, HIGH);

delay(100);

}

for(int y=7;y>=0;y--){

digitalWrite(y, LOW);

delay(100);

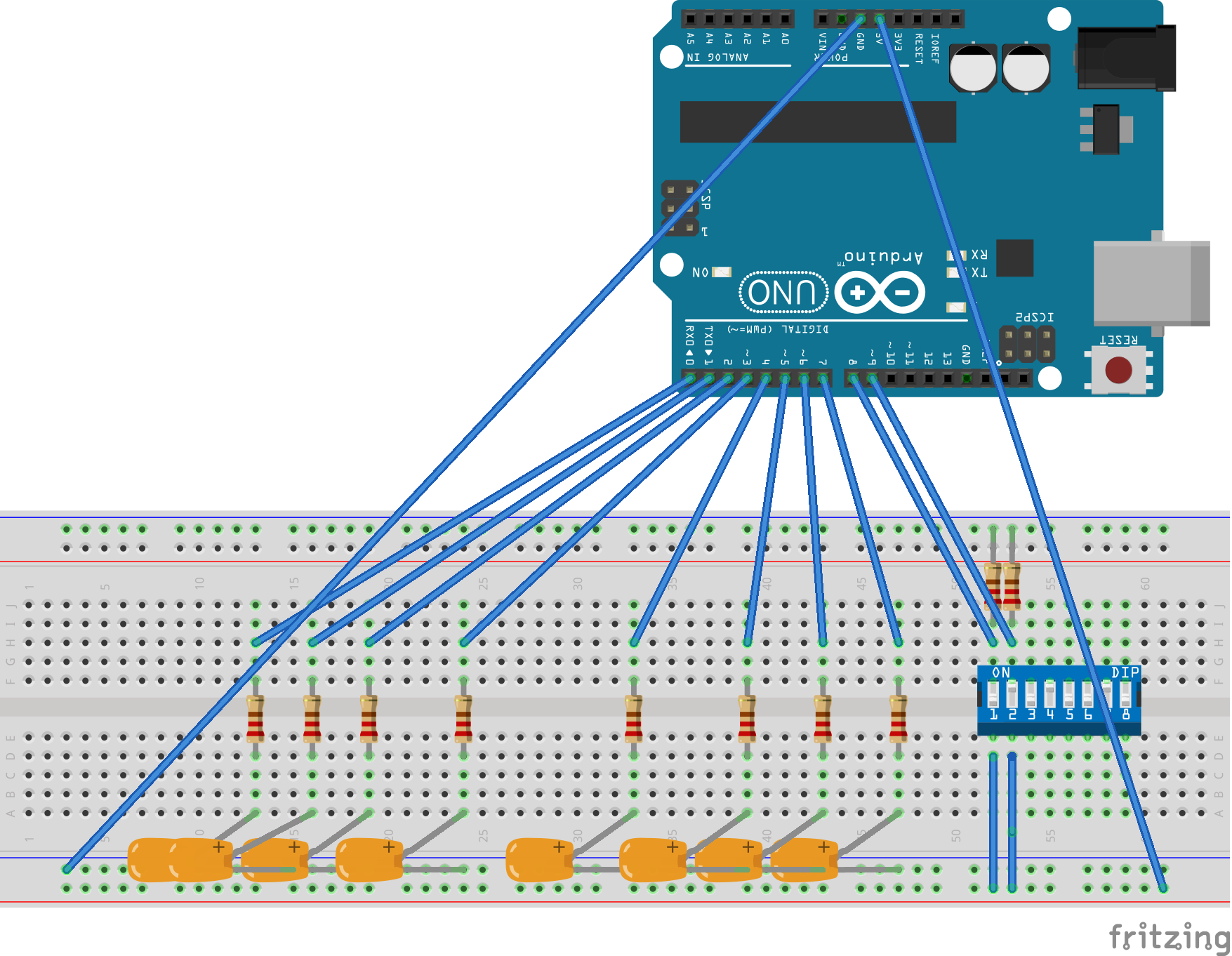
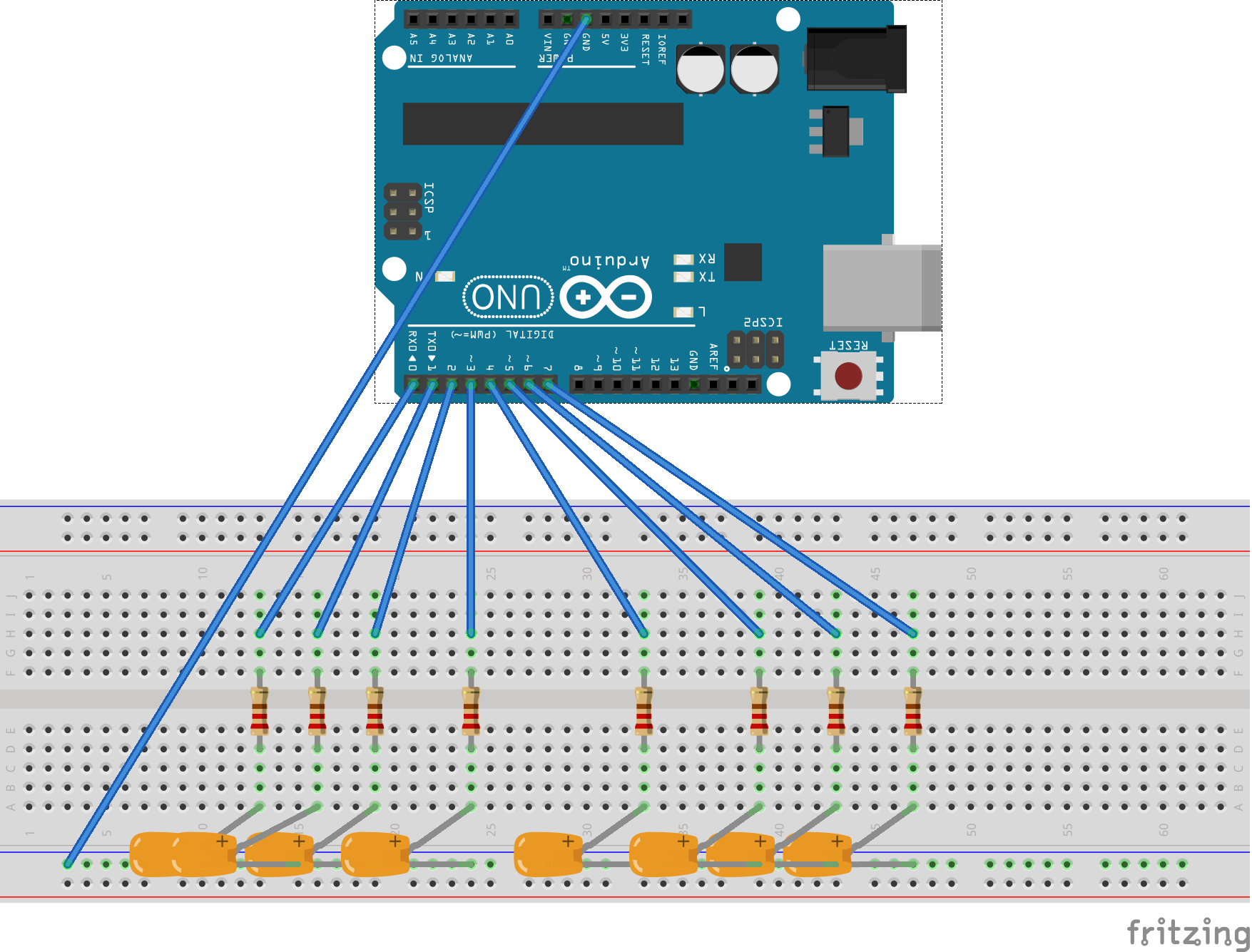
}

break;

}

}

**B) DIAGRAMA EN FRITZING**



**2.** La empresa de automatización TEC-EMI te contrata para hacer un montaje de un LED Intermitente en una placa Arduino UNO, con el único requisito de que el LED debe estar ubicado en el Pin 5. Realizar el montaje y la respectiva programación de 2 segundos prendido y 1 segundo apagado

voidsetup()

{

pinMode(5,OUTPUT);

}

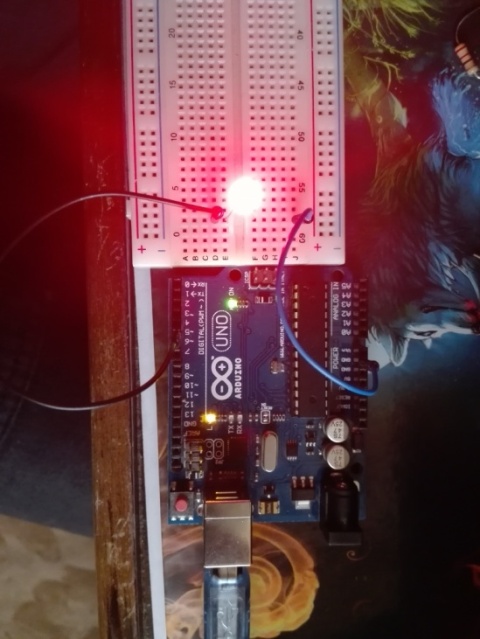
void loop()

{

digitalWrite(5,HIGH);

delay(2000);

digitalWrite(5,LOW);

delay(1000);

}

**3.** La multinacional francesa Flante experta en maquinaria industrial te ha contratado para que automatices una máquina cortadora de papel. La condición principal es que el operario de la máquina cuando vaya a realizar el corte siempre mantenga las dos manos ocupadas, esta es una regla de seguridad industrial para evitar accidentes. El operario debe oprimir los dos pulsadores uno con cada mano y la cuchilla cortadora debe bajar y hacer el corte. El siguiente montaje simula el control de la máquina, los dos pulsadores (S1 y S2) y el LED rojo simula la cuchilla cortadora.

voidsetup()

{

pinMode(0, OUTPUT);

pinMode(8,INPUT);

pinMode(9,INPUT);

}

boolean status;

void loop()

{

int a=digitalRead(8);

int b=digitalRead(9);

int c=2\*a+b;

switch(c){

case 0:

digitalWrite(0,LOW);

break;

case 1:

digitalWrite(0,LOW);

break;

case 2:

digitalWrite(0,LOW);

break;

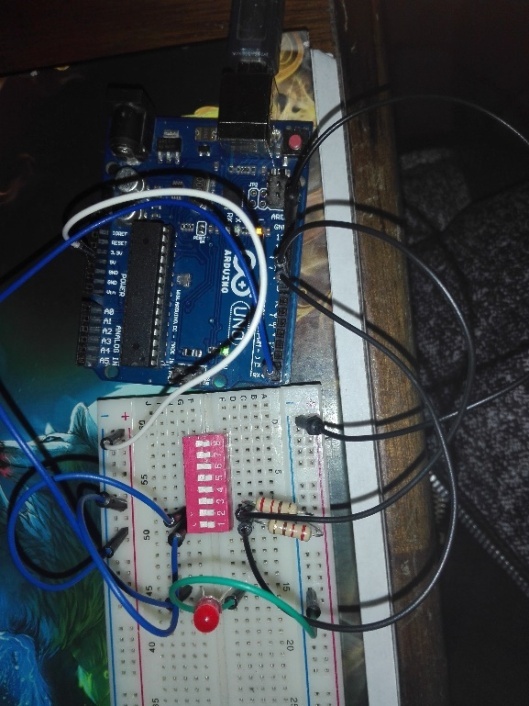
case 3:

digitalWrite(0,HIGH);

break;

}

}

**4.** La empresa TEC-EMI quiere que realices un sistema de iluminaciónLED para las escaleras de una casa. La condición es que si conmutas aS1(swicht1) el LED Rojo se enciende y si luego conmutas S2(swicht2)el LED Rojo se apaga. Es decir que los conmutadores son opuestos siS1 enciende el LED, S2 debería apagarlo y si S2 enciende el LED, S1debería apagarlo.

voidsetup()

{

pinMode(0, OUTPUT);

pinMode(8,INPUT);

pinMode(9,INPUT);

}

boolean status;

void loop()

{

int a=digitalRead(8);

int b=digitalRead(9);

int c=2\*a+b;

switch(c){

case 0:

digitalWrite(0,LOW);

break;

case 1:

if(status==1)

{

digitalWrite(0,HIGH);

status=0;

}else

{

digitalWrite(0,LOW);

status=1;

}

break;

case 2:

if(status==0)

{

digitalWrite(0,HIGH);

status=1;

}else{

digitalWrite(0,LOW);

status=0;

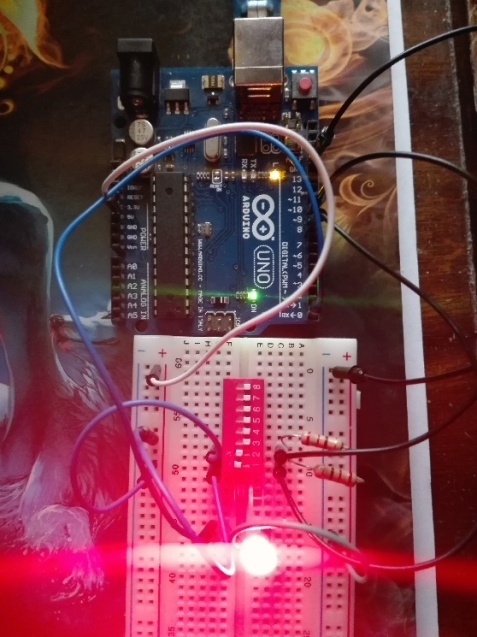
}

break;

case 3:

digitalWrite(0,LOW);

break;

}

}

**5. INVESTIGUE SOBRE LAS FUNCIONES ESPECIALES DEL ARDUINO UNO.**

El lenguaje del Arduino incluye una serie de funciones que están incluidas directamente en la librería básica (“Core”) y que podemos usar sin más. Estas funciones se pueden agrupar en diferentes tipos:

* Funciones de entrada/salida digital (I/O: Input/Output).
* Funciones de entrada/salida analógicas.
* Funciones avanzadas de entrada/salida.
* Funciones de entrada/salida del ArduinoDue.
* Funciones temporales.
* Funciones matemáticas.
* Funciones trigonométricas.
* Funciones aleatorias.
* Funciones de manipulación de bits y bytes.
* Funciones de gestión de interrupciones.
* Funciones de gestión de comunicaciones.
* Funciones de gestión del puerto USB (Leonardo y Due solamente).
* Funciones de conversión de tipos.

**6. DESCRIBA LAS CARACTERÍSTICAS DE POR LO MENOS CINCO DIFERENTES TIPOS DE**

**Arduino UNO**

Es la plataforma más extendida y la primera que salió al mercado, por ello nos podemos basar en esta para hacer la comparativa con el resto de placas. Todas las características de esta placa estarán implementadas en casi todas las placas restantes, a excepción de algunas que ya veremos. Se basa en un microcontrolador Atmel ATmega320 de 8 bits a 16Mhz que funciona a 5v. 32KB son correspondientes a la memoria flash (0,5KB reservados para el bootloader), 2KB de SRAM y 1KB de EEPROM. En cuanto a memoria es una de las placas más limitadas, pero no por ello resulta insuficiente para casi todos los proyectos que rondan la red. Las salidas pueden trabajar a voltajes superiores, de entre 6 y 20v pero se recomienda una tensión de trabajo de entre 7 y 12v. Contiene 14 pines digitales, 6 de ellos se pueden emplear como [PWM](https://es.wikipedia.org/wiki/Modulaci%C3%B3n_por_ancho_de_pulsos). En cuanto a pines analógicos se cuenta con hasta 6. Estos pines pueden trabajar con intensidades de corriente de hasta 40mA.

**ArduinoTRE**

Primera placa Arduino fabricada en Estados Unidos. Integra un procesador Texas Instrument Sitara AM335x de 1Ghz basado en ARMCortex A8 con 512MB de DDR3L, lo que le da hasta 100 veces más rendimiento comparado con otras placas como Leonardo y Uno. Esto abre las puertas a más aplicaciones avanzadas y soporte para sistemas basados en Linux. Por un lado sigue contando con el microcontrolador Atmel ATMega32u4 de 16Mhz y 32KB de flash y 2.5KB de SRAM, junto al 1KB de EEPROM. Tiene 14 pines digitales, 7 PWM, 6 analógicos multiplexados, y su parte SBC cuenta con HDMI, USB, microSD, soporte para conector LCD, etc… Todo compatible con los escudos de Arduino y con Arduino IDE.

**Arduino/Genuino 101**

Se trata de una placa que se conoce como Arduino 101 en América y Genuino 101 fuera de Estados Unidos. Esta nueva placa ha sido presentada en el OpeningConference at Maker Faire de Roma, y su precio ronda los 30$ (27€). Genuino 101 sigue la misma filosofía de las placas oficiales de Arduino, pero llama la atención su módulo Intel Curie, un módulo de dimensiones reducidas y bajo consumo potenciados por el SoC Intel Quark de 32 bits. Se trata de un SoC que contiene un microcontrolador x86 (una oportunidad única de programar en una plataforma x86, alejándose de los ATMega y los ARM), 80KB de SRAM (24KB disponible para sketches), 384 KB de memoria flash, DSP, Bluetooth, sensores acelerómetros y giroscopio, etc… Por el resto de características, como conexiones y tamaño, es igual a Arduino UNO y compatibles con sus shields.

**Arduino Zero**

En aspecto es similar a Arduino UNO, pero esta placa esconde sorpresas con respecto a la plataforma UNO. En vez del microcontrolador AtmelATmega basado en arquitectura AVR de 8 bits, el Zero contiene un potente Atmel SAMD21 MCU de 48Mhz con un coreARMCortex M0 de 32 bits. Con 256 KB de memoria flash, 32 KB de SRAM y una EEPROM de más de 16KB por emulación. El voltaje en el que opera es de 3v3/5v (7mA) y contiene 14 pines E/S digitales, de los cuales 12 son PWM y UART. En el terreno analógico se dispone de 6 entradas para un canal ADC de 12 bits y una salida analógica para DAC de 10 bits. En definitiva, esta placa va destinada para los que Arduino UNO se les quede corto y necesitan algo más de potencia de procesamiento.

**Arduino Zero Pro**

La Zero Pro es una versión mejorada del modelo Zero normal. Ideal para proyectos que demanden una mayor potencia de cómputo, ya que integra un potente microcontrolador de 32-bit como es el Cortex M0+ basado en ARM. Éste corre a 48Mhz de reloj, y va integrado en un Atmel SAMD21 MCU. Por el resto de características es similar a la Zero.

**7. INVESTIGUE SOBRE LOS LENGUAJES DE PROGRAMACIÓN EXISTENTES PARA ARDUINO.**

**Net**

Muchos son los que tienen experiencia con el conjunto de lenguajes del framework de Microsoft .NET y es que  Microsoft, esta apostando por la creación de wearables con su propia versión del sdk para Intel Galileo(La cual estuvo regalando a algunos para comenzar a desarrollar). Si no tenemos esta versión del intel galileo, podemos utilizar una pequeña placa compatible con Arduino UNO llamada netduino. La cual nos permite programar nuestro arduino utilizando el entorno de desarrollo Visual C#(o Visual Studio) con el lenguaje de programación C#.

**Python**

Aunque no se trata de realizar la programación directamente en nuestra placa, se puede utilizar la librería pyfirmdata para poder controlar el arduino de forma remota(además también podemos hacerlo como hemos hecho en otros casos como con los artículos de [Internet de las cosas](http://booleanbite.com/web/internet-de-las-cosas-1-conexion-raspberry-arduino/)). Por un lado, necesitaremos poner un programa especifico en nuestro arduino y así podemos mandar las ordenes desde nuestro equipo usando python. Este programa puede encontrarse en los ejemplos de arduino ejemplos->firmada->StandartFirmData. una vez que pasamos este programa, ya podemos usar python.

**Node.js**

Para los amantes de JavaScript, tenemos una versión del anterior ejemplo pero usando una librería cuyo nombre nos puede dar nostalgia; (supongo que alguno se lo imaginará al ver la imagen del artículo) la librería jhony-five para nodeJS (Basado el nombre en el carismático robot de la película Cortocircuito)  nos permite programar nuestro arduino de igual manera que lo hecho hecho con python.

**Java**

En este caso como en los demás, no podemos ejecutar directamente el código Java Pero si podemos mandarlo por el puerto USB. para ello, usamos la librería RXTXcommons y la librería Arduino  para Java.

**CONCLUSIÓN**

Realizando la programación en arduino se vio que es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware libre, flexibles y fáciles de usar, también introduciéndonos al diseño e implementación de programas de aplicación básicos, utilizando placas ARDUINO, específicamente el ARDUINO UNO, realizando cada uno de los ejercicios propuestos en el laboratorio.