```
/* SEGUIDOR DE LINEA UPNA */

//Este ejemplo es un programa para arduino que utiliza la librería de Pololu que
//aparece a continuación, y que está diseñada para seguidores de línea y
//sensores QTR-1RC y QTR-8RC.


#include <PololuQTRSensors.h>


//Este ejemplo se ha diseñado para su uso con ocho sensores de QTR-1RC o un
//módulo de ocho sensores de QTR-8RC. Estos sensores de reflectancia se debe
//conectar a las entradas digitales 3 a 10, aubque en este ejemplo como solo
//utilizamos 6 sensores y el pin de control del emisor no lo activamos
//conectaremos a los pines 2 a 7. El pin de control del emisor, opcionalmente se
//puede conectar al pin digital de 2, o puede dejarse que desconectado y cambiar
//el EMITTER_PIN # definido por QTR_NO_EMITTER_PIN.

//En la fase de instalación de este ejemplo se calibra el sensor durante diez
//segundos y se enciende un LED de control, mientras que la calibración que está
//produciendo. Durante esta fase, se debe exponer cada sensor de reflectancia a
//las lecturas más claras y más oscuras con las que se vaya ha encontrar. Por
//ejemplo, si usted está haciendo un seguidor de línea, usted debe deslizarse a
//través de los sensores de la línea durante la fase de calibrado de forma que
//cada sensor puede obtener una lectura de cuanto oscura es la linea y cun claro
//es el suelo. Una calibración incorrecta dará lugar a lecturas pobres.
//Si se desea omitir la fase de calibración, se puede obtener las lecturas del
//sensor (tiempos de pulso desde 0 hasta 2500 us) llamando a la función
//qtra.read (sensorValues) en lugar de qtra.readLine (sensorValues).

//El bucle principal del ejemplo lee los valores de los sensores calibrados y
//los utiliza para estimar la posición de una línea. Se puede probar esto con un
//pedazo de cinta adhesiva  de 3 / 4 "cinta aislante negra pegada sobre un
//pedazo de papel blanco y deslizar el sensor a través de ella. Imprime los
//valores del sensor al monitor en serie, según los números del 0 (máxima
//reflectancia) a 9 (mínimo reflectancia), seguido de la ubicación aproximada de
//la línea como un número del 0 al 5000. 1000: la línea está directamente debajo
//del sensor 1, 2000 significa directamente debajo del sensor 2, etc. 0
//significa que la línea está directamente debajo del sensor 0 o fue visto por
//ultima vez por el sensor 0 antes de que se pierdan. 5000: la línea está
//directamente debajo del sensor de 5 o fue visto por última vez por el sensor
//de 5 antes de que se pierdan.



//Definimos el valor de varias constantes que utiliza el programa
#define NUM_SENSORS   6     // número de sensores utilizados
#define TIMEOUT       2500  // espera 2500us para apagar los sensores
#define QTR_NO_EMITTER_PIN  // el emisor no está controlado

//Los sensores del 1 al 6 están conectados a los pines digitales 2 a 7
PololuQTRSensorsRC qtrrc((unsigned char[]) {2, 3, 4, 5, 6, 7},  NUM_SENSORS, TIMEOUT);
unsigned int sensorValues[NUM_SENSORS];  //inicializamos una variable vacía

//Los pines de control de los motores van a ser 10 y 11 para la dirección,
//y 12 y 13 para la velocidad. Los pines 10 y 12 serán de un motro y 11 y 13
//del otro.
int dirA = 11;
int dirB = 10;
int speedA = 13;
int speedB = 12;
```

```
//Comienza la fase de instalación en la que se va ha realizar la calibración

void setup()
{

  // definimos los pines de control de los motores como salida
  pinMode (dirA, OUTPUT);
  pinMode (dirB, OUTPUT);
  pinMode (speedA, OUTPUT);
  pinMode (speedB, OUTPUT);

  delay(500);                    // comenzamos con 500ms de espera

  int i;                    // inicilizamos el contador i
  pinMode(13, OUTPUT);              // ponemos el pin del LED como salida
  digitalWrite(13, HIGH);   // encendemos el LED para iniciar la calibracion

  for (i = 0; i < 400; i++) // hace que la calibración dure unos 10 segundos
  {
    qtrrc.calibrate();      // todos los sensores realizan 10 lecturas de 2500us
  }
  digitalWrite(13, LOW);   // apagamos el LED para acabar el modo de calibracion

  //escribe en pantalla la media de los mínimos valores medidos para cada sensor
  Serial.begin(9600);  // velocidad de transmisión entre la placa y el ordenador
  for (i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(qtrrc.calibratedMinimumOn[i]);
    Serial.print(' ');
  }
  Serial.println();

  //escribe en pantalla la media de los máximos valores medidos para cada sensor
  for (i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(qtrrc.calibratedMaximumOn[i]);
    Serial.print(' ');
  }

  Serial.println();
  Serial.println();
  delay(1000);

}

//Una vez realizada la calibración comienza el bucle principal donde pretende
//que sigua una línea negra sobre un fondo blanco

void loop()
{
  // lee los valores de calibrado de los sensores y obtiene una medida de la
  // posición de línea 0 a 5000, donde 0 significa que directamente debajo del
  // sensor 0 o la línea se perdió pasado el sensor 0, 1000 significa que
  // directamente debajo del sensor 1, 2000 significa que directamente debajo
  // del sensor 2, etc.
  // Nota: los valores devueltos serán incorrectos si los sensores no han sido
  // adecuadamente calibrados durante la fase de calibración.  Para obtener los
  // valores en bruto del sensor llamar a la función: qtra.read(sensorValues);
```

```
  unsigned int position = qtrrc.readLine(sensorValues);

  //qtrrc.read(sensorValues);  // en caso de querer leer los valores en bruto

  // escribir los valores de los sensores como un número del 0 al 9, donde 0
  // significa máxima reflectancia y 9 mínima reflectancia, seguido de la
  // posición de la línea
  unsigned char i;
  for (i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(sensorValues[i] * 10 / 1001);
    Serial.print(' ');
  }
  Serial.print("    ");
  Serial.println(position);

  // Una vez conocemos la posición de la línea con respecto a los sensores
  // podemos decidir que acciones debe realiazar para ir siguiendo la línea

  // En caso de que la línea se encuentre entre los sensores 2 y 3 queremos que
  // siga hacia adelante, de manera que le damos la misma dirección y velocidad
  // a ambos motores
  if (position > 2400&&position < 2600)
  {
    Serial.print(" arranca");
    analogWrite (dirA, 255);
    analogWrite (dirB, 255);
    digitalWrite (speedA, LOW);
    digitalWrite (speedB, LOW);
  }

  // En caso de que la línea se encuentre a la derecha del sensor 2 queremos
  // que gire hacia la izquierda y vuelva a la línea, así que le damos la misma
  // dirección a los dos motores pero velocidades distintas. Le damos menos
  // velocidad a la rueda interior del giro
  if (position < 2400)
  {
    Serial.print(" izquierda");
    analogWrite (dirA, 255);
    analogWrite (dirB, 50);
    digitalWrite (speedA, LOW);
    digitalWrite (speedB, LOW);
  }


  // En caso de que la línea se encuentre a la izquierda del sensor 3 queremos
  // que gire hacia la derecha y vuelva a la línea, así que le damos la misma
  // dirección a los dos motores pero velocidades distintas. Le damos menos
  // velocidad a la rueda interior del giro
  if (position > 2600)
  {
     Serial.println(" derecha");
     analogWrite (dirA, 50);
     analogWrite (dirB, 255);
     digitalWrite (speedA, LOW);
     digitalWrite (speedB, LOW);
  }
}
```