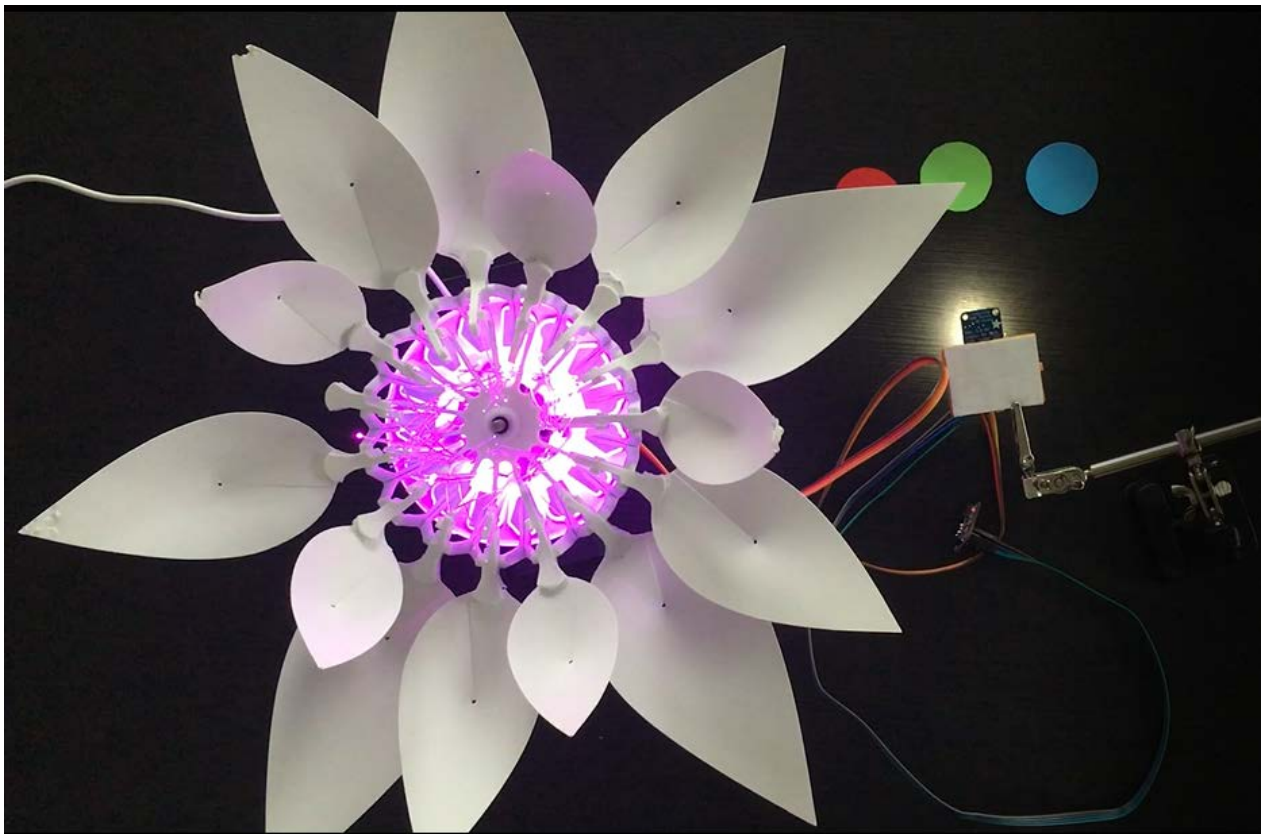
Schritt für Schritt

Die Vielfalt der Blume



Die Bestäubung ermöglicht den Transport von Pollen von den männlichen Fortpflanzungsorganen zum weiblichen Fortpflanzungsorgan, welches wiederum die Fortpflanzung der Pflanzen ermöglicht. Dieser Transport erfolgt entweder innerhalb der Blüten (Selbstbestäubung) oder durch Fremdbestäubung (die Pollen einer Blüte wird auf den Stempeln einer anderen Blüte übertragen). Im letzteren Fall können die Bestäubungsvektoren biotisch (Vögel, Insekten…) oder abiotisch (Wind, Wasser,…) sein. So kann die Fremdbestäubung dazu führen, dass sich Blumensorten kreuzen und so neue Arten z.B mit. einer neuen Pflanzenfarbe entstehen. In diesem Projekt wirst du die Vielfalt der Blume in Hinblick auf die Farbe der Blüte durch die Verbreitung der Samen durch den Wind nachbilden.

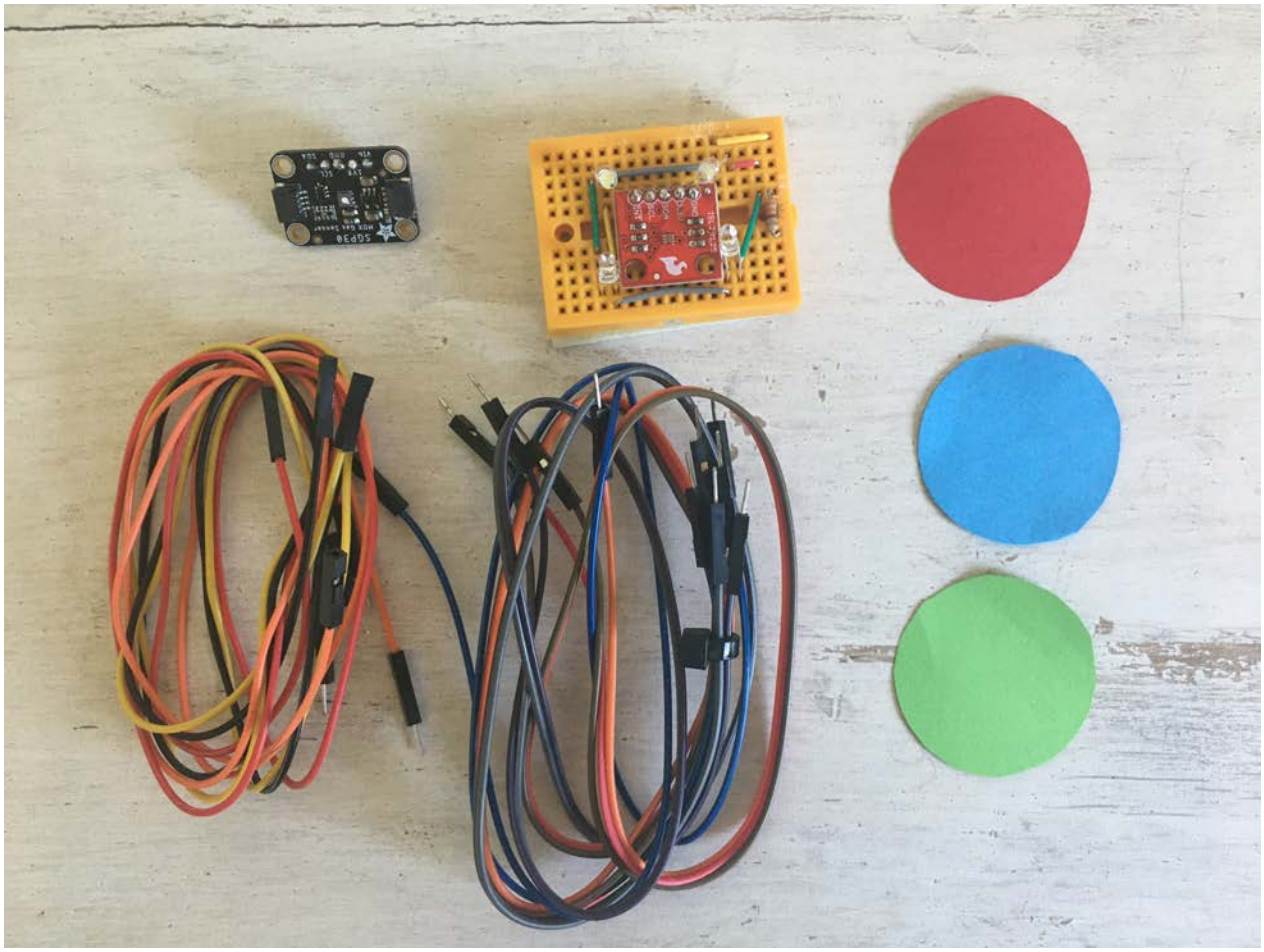
Du brauchst ein rotes, grünes und blaues Papier. Jedes Papier stellt einen Samen einer Art von Blume mit Blütenblättern dar. Die Blume öffnet sich (50 %), wenn eine Farbe erkannt wird und die LEDs leuchten entsprechend der Farbe. Wenn eine zweite Farbe erkannt wird, musst du den CO2-Sensor anhauchen, was den Wind nachbildet. Die Blume öffnet sich (100 %) und leuchtet in der gleichen Farbe wie die Mischung der beiden Farben.

Zielsetzung

- Du kannst eine LED schalten.
- Du kannst einen Farbsensor integrieren.
- Du kannst einen Kohlenstoffdioxidsensor integrieren.
- Du kannst einen Schrittmotor ansteuern.
- Du verstehst Bedingungen.
- Du kannst mit globalen und lokalen Variablen umgehen.
- Du verstehst logische Operatoren.
- Du verstehst das Schreiben und Aufrufen von Funktionen.

Material

- 1 Bionic Flower
- 1 Farbsensor
- 1 Kohlenstoffdioxidsensor
- Jumper Kabel
- *DIVERSITY_OF_THE_FLOWER_Code_Challenge.ino* (download on github)



Aufgabe 1: Steuer die LEDs an

Einige Variablen oder Funktionen (z. B. *RGB_Calibration ()* sind auskommentierte Codezeilen, diese werden für Aufgabe 2 hilfreich sein.

Ändere die Farbe der LED's. Die Bionic Flower besteht aus 5 eingebauten LEDs. Die Farbe jeder LED ist durch einen RGB-Code gegeben. Die LEDs sind an GPIO 16 angeschlossen.

Verdrahtungsplan:

LEDs	ESP32
LEDs	GPIO 16

Code:

1. Öffne die Datei: *DIVERSITY_OF_THE_FLOWER_Code_Challenge.ino*.
2. *Bibliothek*

   Füg die Bibliothek zur Steuerung der LEDs hinzu.
3. *Globale Variablen*
   ** Definiere den GPIO der LEDs und gib ihm den Variablennamen "LED_PIN".
   ** Erstelle das Objekt für die LEDs.
   ** Erstelle 6 Funktionen zum Aufleuchten der LEDs in den Farben Blau, Rot, Grün, Magenta (Mischung zwischen Blau und Rot), Cyan (Mischung zwischen Blau und Grün) und Gelb (Mischung zwischen Rot und Grün). Denk daran, eine Funktion zum Ausschalten der LEDs (schwarze Farbe) hinzuzufügen.

   Hier findest du Hilfe zu den Farben: RGB-Code Website-Link : https://www.w3schools.com/colors/colors_picker.asp
4. *setup()*
   ** Initialisiere die LEDs.
   ** Schalte die LEDs aus.
5. *loop()* (Schleife)

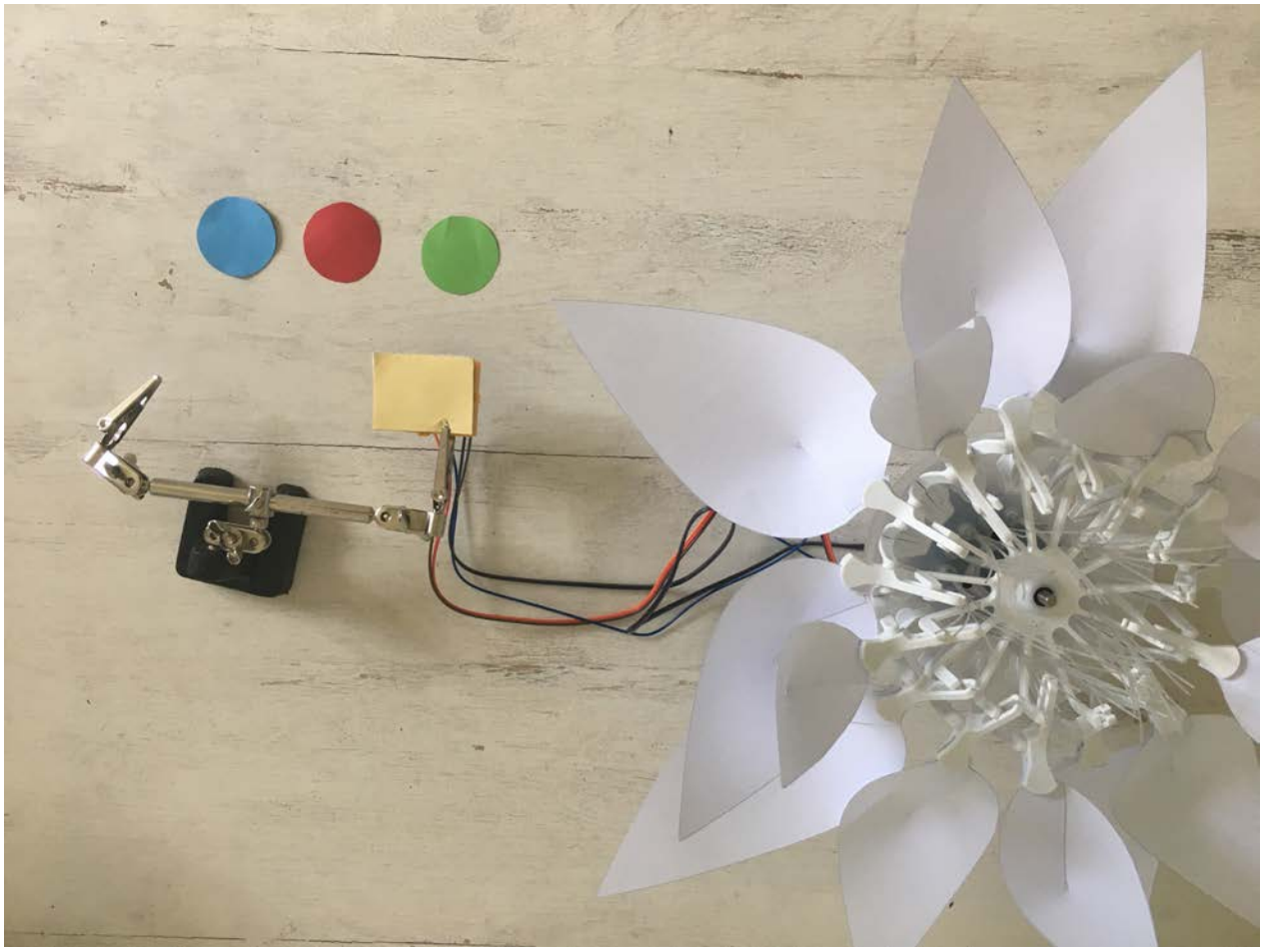
   Erzeuge eine Farbabfolge:
   ** Schalte die LEDs in Blau ein.
   ** Warte 500 ms.
   ** Schalte die LEDs in Rot ein
   ** Warte 500 ms.
   ** Schalten die LEDs in Magenta ein.
   ** Warte 500 ms.
   ** Schalte die LEDs in Grün ein.
   ** Warte 500 ms.
   ** Schalte die LEDs in Cyan ein.
   ** Warte 500 ms.
   ** Schalte die gelben LEDs ein.
   ** Warte 500 ms.

Aufgabe 2: Integriere den Farbsensor

Jede Farbe ist durch ihren RGB-Code, eine Farbkomponente von 0 bis 255, definiert. (https://www.w3schools.com/colors/colors_picker.asp).Nach einer Kalibrierung misst der Farbsensor den Wert der roten, grünen und blauen Farbkomponenten.
Der Farbsensor kommuniziert über das I2C-Protokoll, er nutzt also die Pins SCL und SDA. Die I2C-Adresse des Farbsensors ist 0x44.

In dieser Aufgabe leuchtet die Blume in der Farbe, die der Farbsensor erkennt (rot, blau oder grün).

- Die Funktion *BW_Calibration()* erlaubt es, den maximalen RGB-Wert durch Messung eines weißen Objekts und den minimalen RGB-Wert mit einem schwarzen Objekt zu messen. Mit diesen Werten können dann die RGB-Werte aller Farben berechnet werden.
- Die Funktion *RGB_Calibration()* ermöglicht die Messung der RGB-Komponenten für die Farben Rot, Grün und Blau. Diese Werte werden dann in der Funktion *Read_RGB()* verwendet, um festzustellen, ob eine dieser Farben von dem Sensor erfasst wird.
- Die Funktion *Read_RGB()* ermöglicht das Auslesen der Werte der RGB-Komponenten. Sie ermittelt, ob sich die rote Farbe unter dem Sensor befindet (Red=true, wenn dies der Fall ist), ob sich die grüne Farbe unter dem Sensor befindet (GREEN=true, wenn dies der Fall ist) oder ob sich die blaue Farbe unter dem Sensor befindet (BLUE=true, wenn dies der Fall ist).



Verdrahtungsplan:

Farbsensor	ESP32
SCL	GPIO 5
SDA	GPIO 4
(+)	5 V
(-)	GND

Code:

1. *Bibliothek*

   Füg die Bibliothek für die I2C-Kommunikation und für den Farbsensor hinzu.

```
// color sensor's library
#include "SparkFunISL29125.h"
```

2. *Globale Variablen*
   ** Definiere die Pins der I2C-Kommunikation.
   ** Entferne den Kommentarbereich der Codezeilen für den Farbsensor
   ** Definiere die I2C-Adresse des Sensors
   ** Erstelle das Objekt für den Farbsensor.

```
// Color sensor
SFE_ISL29125 RGB_sensor;
```

** Entferne die Kommentarsymbole für die *BW_Calibration()* Funktion, der *RGB_Calibration()* Funktion und der *Read_RGB()* Funktion. Nimm dir Zeit die Funktionen zu lesen und zu verstehen.

3. *setup()*
   ** Starte die I2c-kommunikation
   ** Initialisiere den Farbsensor.

```
//Color sensor initialisation
 RGB_sensor.init();
```

** Kalibriere den Sensor zuerst mit den schwarz und weißen Referenzobjekten (wie Papierblätter) und dann mit den rot, grün und blauen.
** Warte 3s.

4. *loop()* (Schleife)
   ** Schreibe eine while-Schleife:
   o Solange keine Farbe erkannt wird (kein Rot, kein Blau, kein Grün), lies den Farbsensor weiter aus.
   ** Wenn du aus der while-Schleife heraus bist, ist eine Farbe erkannt worden, also kannst du die Blume in der detektierten Farbe beleuchten.
   ** Warte 5s.
   ** Schalte dann die LEDs aus.
   ** Setzt die boolesche Variable (ROT,BLAU und GRÜN) auf false.

Aufgabe 3: Integriere den Schrittmotor

Benutze den Schrittmotor zum Öffnen und Schließen der Bionic Flower.
Verwenden Sie den Schrittmotor, um die Bionic Flower zu öffnen oder zu schließen.

Bei dieser Aufgabe leuchtet bei Erkennung einer Farbe die Blume in dieser Farbe und die Blume öffnet sich zur Hälfte (50 %)

Code:

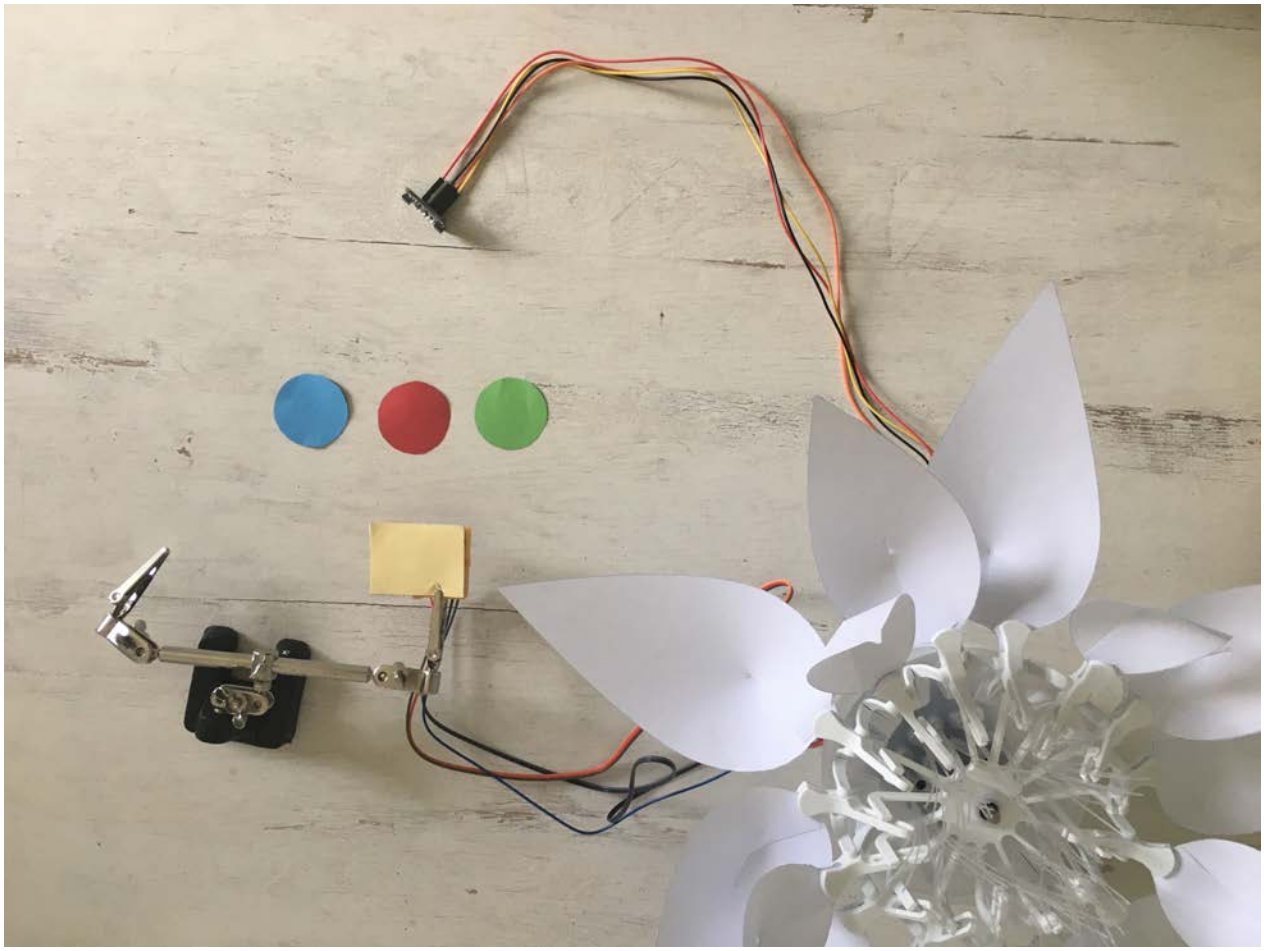
1. *Bibliothek*

   Füg die Bibliothek zur Steuerung des Motors der Bionic Flower hinzu.
2. *Globale Variablen*
   ** Definier den GPIO und die Variable für den Motor.
   ** Erstelle das Objekt für den Motor.
   ** Füge die Funktion *motor_calibration()* hinzu.
3. *setup()*
   ** Initialisiere den Motor.
   ** Kalibriere den Motor.
4. *loop()* (Schleife)
   ** Schreibe eine while-Schleife:
   ** Solange keine Farbe erkannt wird (kein Rot, kein Blau, kein Grün), lies den Farbsensor weiter aus.
   ** Wenn du aus der while-Schleife heraus bist, ist eine Farbe erkannt worden, also kannst du die Blume in der detektierten Farbe beleuchten.
   ** Öffne die Blume (50%).
   ** Warte 5s.
   ** Schließe die Blume.
   ** Schalte die LEDs aus.
   ** Setze die boolesche Variable (ROT,BLAU und GRÜN) auf false.

Aufgabe 4: Integriere den Kohlenstoffdioxidsensor (CO2)

Der CO2-Sensor liefert eine ungefähre Messung des CO2-Gehalts in der Luft. Der Farbsensor kommuniziert über das I2C-Protokoll, er nutzt also die Pins SCL und SDA. Die I2C-Adresse des Farbsensors ist 0x58.

In dieser Aufgabe wirst du den CO2-Wert in der Luft messen. Wenn eine Farbe erkannt wird, leuchtet die Blüte in dieser Farbe auf und die Blüte öffnet sich teilweise (50%). Wenn dann ein Atemzug erkannt wird (der den Wind darstellt), schließt sich die Blume.



Verdrahtungsplan:

Kohlenstoffdioxidsensor	ESP32
SCL	GPIO 5
SDA	GPIO 4
(+)	5 V
(-)	GND

Code:

1. *Bibliothek*

   Füg die Bibliothek zur Integration des CO2-Sensors hinzu.

```
// CO2 sensor's library
#include "Adafruit_SGP30.h"
```

2. *Globale Variablen*
   ** Definiere die I2C-Adresse des Sensors.
   ** Definieren eine globale Variable zum Speichern des CO2-Niveaus
   ** Definiere eine globale Variable für einen CO2-Schwellenwert zur Erkennung eines Atems (ca. 700 ppm).
   *** Erstelle das Objekt für den CO2-Sensor.

```
//CO2 sensor
Adafruit_SGP30 sgp;
```

3. *setup()*
   ** Initialisiere den CO2-Sensor.

```
// CO2 sensor initialisation
sgp.begin();
```

4. *loop()* (Schleife)
   ** Schreibe eine while-Schleife:
   ** Solange keine Farbe erkannt wird (kein Rot, kein Blau, kein Grün), lies den Farbsensor weiter aus.
   ** Wenn du aus der while-Schleife heraus bist, ist eine Farbe erkannt worden, also kannst du die Blume in der detektierten Farbe beleuchten.
   ** Öffne die Blume (50%).
   ** Lies den CO2-Wert aus.

```
//Start the transmission with the CO2 sensor
 Wire.beginTransmission(ADD_CO2);
 //Read the CO2 value
 if (! sgp.IAQmeasure()) {return;}
 int CO2_value = sgp.eCO2;
 Serial.print("eCO2 "); Serial.println(CO2_value);
 // End the transmission with the CO2 sensor
 Wire.endTransmission();
```

** Schreibe eine while-Schleife
** solange kein Atemzug erkannt wird, lies den CO2-Wert aus.
** Wenn du aus der while-Schleife heraus bist, ist ein Atemzug erkannt worden, und die Blume schließt sich.
** Schalte die LEDs aus.
** Setze die boolesche Variable (ROT,BLAU und GRÜN) auf false.

Aufgabe 5: Das ganze Szenario

Erstelle nun den endgültigen Code, um das Szenario nachzubilden :
** Zuerst wird eine Farbe erfasst (rot, blau oder grün) und die Blume leuchtet in dieser Farbe und ist zu 50% geöffnet.
** Eine zweite Farbe wird erfasst (rot, blau oder grün) und ein Atemzug wird registriert, und die Blume leuchtet in einer Mischfarbe (Magenta, Cyan oder Gelb), je nach den 2 Farben, die zuvor erfasst wurden. Die Blume ist nun vollständig geöffnet (100 %).
** Nach 5s schließt sich die Blume und schaltet die LEDs aus.