Übung 01 Modi

```
const int LedPin1 = 10;                          // Pin-Nummer für LED 1
const int LedPin2 = 11;                          // Pin-Nummer für LED 2
const int ButtonPin1 = 1;                        // Pin-Nummer für Schalter 1
const int ButtonPin2 = 2;                        // Pin-Nummer für Schalter 2
int buttonState1 = LOW;                          // Variable des aktuellen Schalterzustands 1
int buttonState2 = LOW;                          // Variable des aktuellen Schalterzustands 2
int previousButtonState1 = LOW;                  // Variable des vorherigen Schalterzustands 1
int previousButtonState2 = LOW;                  // Variable des vorherigen Schalterzustands 2
int mode = 0;                                    // Variable des voreingestellten Modus

void setup() {
pinMode(LedPin1, OUTPUT);                        // LED-Pin 1 als Ausgang
pinMode(LedPin2, OUTPUT);                        // LED-Pin 2 als Ausgang
pinMode(ButtonPin1, INPUT);                      // Schalter-Pin 1 als Eingang
pinMode(ButtonPin2, INPUT);                      // Schalter-Pin 2 als Eingang
}                                                // void setup() schliessen
void loop() {
int buttonState1 = digitalRead(ButtonPin1);      // Lese den aktuellen Zustand von Schalter 1
int buttonState2 = digitalRead(ButtonPin2);      // Lese den aktuellen Zustand von Schalter 2

if (buttonState1 == HIGH && previousButtonState1 == LOW) {mode = 1;}     // Wechsle den Modus
if (buttonState2 == HIGH && previousButtonState2 == LOW) {mode = 2;}     // Wechsle den Modus

if (mode == 1) {                                 // Nach Modus 1 LEDs schalten
digitalWrite(LedPin1, HIGH);                     // LED 1 einschalten
digitalWrite(LedPin2, LOW);                      // LED 2 ausschalten
}
else if (mode == 2) {                            // Nach Modus 2 LEDs schalten
digitalWrite(LedPin1, LOW);                      // LED 1 ausschalten
digitalWrite(LedPin2, HIGH);                     // LED 2 einschalten
}

previousButtonState1 = buttonState1;             // Aktualisiere den Zustand von Schalter 1
previousButtonState2 = buttonState2;             // Aktualisiere den Zustand von Schalter 2
}                                                // void loop() schliessen
```

Übung 02 Serial Monitor

```
const int ButtonPin1 = 2;                    // Pin-Nummer für den Schalter 1
const int ButtonPin2 = 3;                    // Pin-Nummer für den Schalter 2
const int ButtonPin3 = 4;                    // Pin-Nummer für den Schalter 3
const int PotentiometerPin = A0;             // Pin-Nummer für den Poti
int buttonState1 = LOW;                      // Variable des aktuellen Schalterzustands 1
int buttonState2 = LOW;                      // Variable des aktuellen Schalterzustands 2
int buttonState3 = LOW;                      // Variable des aktuellen Schalterzustands 3
potValue = 0;                                // Variable des initialen Potizustands

void setup() {
Serial.begin(9600);                          // Starte die serielle Kommunikation
pinMode(ButtonPin1, INPUT_PULLUP);           // Schalter-Pin 1 als Eingang
pinMode(ButtonPin2, INPUT_PULLUP);           // Schalter-Pin 2 als Eingang
pinMode(ButtonPin3, INPUT_PULLUP);           // Schalter-Pin 3 als Eingang
pinMode(PotentiometerPin, INPUT);            // Poti-Pin als Eingang
}                                            // void setup() schliessen

void loop() {
int buttonState1 = digitalRead(ButtonPin1);  // Lese den aktuellen Zustand von Schalter 1
int buttonState2 = digitalRead(ButtonPin2);  // Lese den aktuellen Zustand von Schalter 2
int buttonState3 = digitalRead(ButtonPin3);  // Lese den aktuellen Zustand von Schalter 3
int potValue = analogRead(PotentiometerPin); // Lese den aktuellen Wert des Potentiometers

Serial.println ("Schalter 1: ");             // Zustand von Schalter 1 als Serial Print
Serial.print (buttonState1);
Serial.println ("Schalter 2: ");             // Zustand von Schalter 2 als Serial Print
Serial.print(buttonState2);
Serial.println ("Schalter 3: ");             // Zustand von Schalter 3 als Serial Print
Serial.print(buttonState3);
Serial.println ("Potentiometerwert: ");      // Zustand vom Poti als Serial Print
Serial.print(potValue);
delay(100);                                  // Verzögerung zur Stabilität
}                                            // void loop() schliessen
```

Übung 03 Servomotor

```
#include <Servo.h>                                  // Die Library für den Servo einbinden
Servo myservo;                                      // Servo Objekt für die Servosteuerung

const int PotentiometerPin = A0;                    // Pin-Nummer für den Poti
const int Servomotor = 1;                           // Pin-Nummer für den Servomotor
int potValue = 0;                                   // Variable des initialen Potizustands

void setup() {
Serial.begin(9600);                                 // Starte die serielle Kommunikation
myservo.attach(Servomotor);                         // Servo als Ausgang
mode(PotentiometerPin, INPUT);                      // Poti-Pin als Eingang
}                                                   // void setup() schliessen

void loop() {
int potValue = analogRead(PotentiometerPin);        // Lese Poti aus, Zahl zwischen 0 and 1023
potValue = map(potState, 0, 1023, 0, 180);          // Potiwert konvertieren von 0 bis 180 Grad
myservo.write(potValue);                            // Servomotor ansteuern
delay(15);                                          // Verzögerung zur Stabilität
}                                                   // void loop() schliessen
```

Übung 04 Ultraschallsensor

```
const int Trigger = 2;                          // Pin-Nummer für Trigger
const int Echo = 4;                             // Pin-Nummer für Echo
int leangeX = 0;                                // Variable für die Laenge

void setup() {
Serial.begin(9600);                             // Starte die serielle Schnittstelle
pinMode(Trigger,OUTPUT);                        // Trigger als Output
pinMode(Echo,INPUT);                            // Echo als Input
}

void loop() {
digitalWrite(Trigger,LOW);                      // Trigger-Impuls zuerst ausschalten
delayMicroseconds(2);                           // Verzögerung
digitalWrite(Trigger,HIGH);                     // Tripper-Impuls einschalten

int leangeX =pulseIn(Echo,HIGH);                // Echo-Signal empfangen
leangeXincm = leangeX /29/2;                    // Umrechnen vom Signal in Zentimeter
Serial.println("Der Abstand betraegt ");        // Text über Serial Print ausgeben
Serial.print(leangeXincm);                      // Leange als Serial Print ausgeben
Serial.println(" cm");                          // Einheit als Serial Print ausgeben

if(leangeXincm <10){
Serial.println("Stopp!!!!");                    // Stopp-warnung ausgeben
}
else{
Serial.println("Okay");                         // Das Auto hat genug Abstand
}

delayMicroseconds(10);                          // Verzögerung zur Stabilisierung
}                                               // void loop() schliessen
```

Übung 05 Schrittmotor

```
const int StepPin = 3;                          // Pin-Nummer für die Drehgeschwindigkeit
const int DirPin = 4;                           // Pin-Nummer für die Drehrichtung
const int Time = 1000;                          // Wie lange soll der Motor sich drehen
const int Korrektur = 1;                        // Zeitkorrektur-Faktor (treell zu tgewollt)
int speed = 0;                                  // Initiale Drehgeschwindigkeit
int initial = 0;                                // Initiales Abfragen und Zeitvariable

void setup() {
Serial.begin(9600);                             // Starte die serielle Kommunikation
pinMode(StepPin, OUTPUT);                       // Anzahl Schritte als Ausgang
pinMode(DirPin, OUTPUT);                        // Richtung als Ausgang
}

void loop() {
while (!Serial.available());                    // Puffer auf Daten überprüfen

Serial.println("Wie schnell möchtest du drehen? (Grad pro Sekunde)");  // Geschwindigkeitsfrage
speed = Serial.parseInt() / 360;                // Lese die Geschwindigkeit und konvertiere
sie in Grad pro Sekunde
Serial.println("In welche Richtung möchtest du drehen? ('Rechts' für im Uhrzeigersinn und 'Links' für
im Gegenuhrzeigersinn)");                       // Drehrichtungsfrage
String direction = Serial.readString();         // Lese das eingegebene Wort als Zeichenkette
if (direction == "Rechts") {DirPin = HIGH; }    // Drehrichtung Rechts abspeichern
else if (direction == "Links") {DirPin = LOW; } // Drehrichtung Links abspeichern

Serial.print("Gewählte Geschwindigkeit: ");     // Einstellung über Serial Monitor ausgeben
Serial.println(speed * 360);                    // Einstellung über Serial Monitor ausgeben
Serial.print("Gewählte Richtung: ");            // Einstellung über Serial Monitor ausgeben
if (direction == "Rechts") Serial.println("Im Uhrzeigersinn"); }          // Drehrichtung Rechts
else if (direction == "Links") {Serial.println("Im Gegenuhrzeigersinn");}  // Drehrichtung Links

while (initial < (Time * Korrektur)) {
digitalWrite(StepPin, HIGH);
delayMicroseconds((500 * Korrektur * (1/speed)));   // Geschwindigkeit über die Pause einstellen
digitalWrite(StepPin, LOW);
delayMicroseconds((500 * Korrektur * (1/speed)));   // Geschwindigkeit über die Pause einstellen
initial++;                                      // initial um eins erhöhen
}
initial = 0; /                                  // initial zurücksetzen
}                                               // void loop() schliessen
```