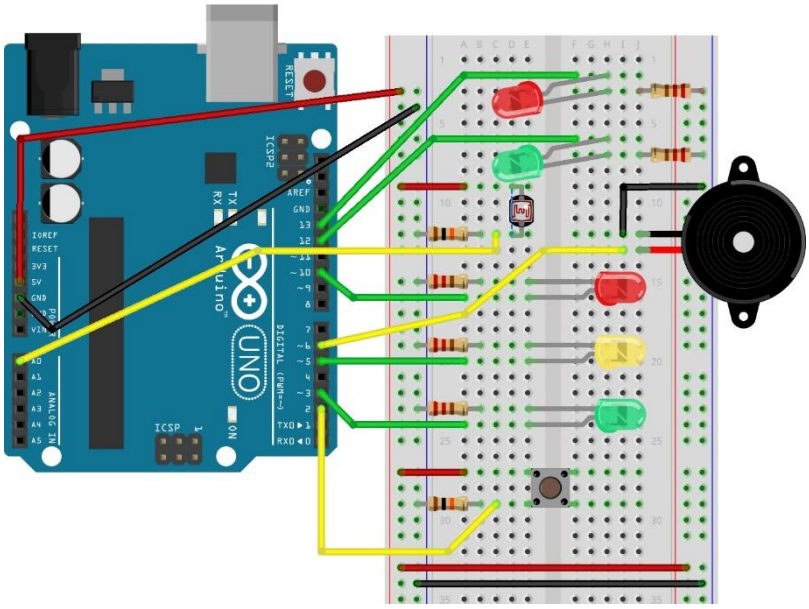
Arduino – Programmierpraxis: Messwerterfassung

Arbeitsanregungen zum Erfassen und Auswerten von Sensor-Messwerten

Insbesondere der Lösung von **Aufgabe 1)** wird die bereits mehrfach verwendete „Ampelschaltung“ (vgl. Abbildung) mit eingebautem Photowiderstand zu Grunde gelegt.



Aufgabe 1)

a) …vgl. Lösungen zu den **Aufgaben 4)** bis **6)** der Datei **ST_AA_08Arduino_Verzweigungen**:

```
//sensor management
int sensorValue;
int sensorLow = 1023;
int sensorHigh = 0;
//LEDs intended for signalling
int greenLedPin = 3;
int yellowLedPin = 5;
int redLedPin = 10;
//array to store sensor values
int dataArray[100];
int index;

void calibratePhotoResistor(){
  sensorValue = analogRead(A0);
  if(sensorValue > sensorHigh){
    sensorHigh = sensorValue;
  }
  if(sensorValue < sensorLow){
    sensorLow = sensorValue;
  }
}
```

```
void printIntArrayTable(int valArray[],int dim){
  int index;

  index = 0;
  Serial.println("");
  while(index < dim){
    if((index % 10) < 9){
      Serial.print(valArray[index]);
      Serial.print(" ");
    }
    else{
      Serial.println(valArray[index]);
    }
    index = index + 1;
  }
  Serial.println("");
}
```

```
void setup() {
  Serial.begin(9600);
  pinMode(yellowLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  digitalWrite(greenLedPin, LOW);
  digitalWrite(redLedPin, LOW);
  //calibrate PhotoResistor
  //lighting the yellow LED to
  //indicate sensor calibration
  digitalWrite(yellowLedPin, HIGH);
  while(millis() < 5000){
    calibratePhotoResistor();
  }
  digitalWrite(yellowLedPin, LOW);
  Serial.print("sensor low: ");
  Serial.println(sensorLow);
  Serial.print("sensor high: ");
  Serial.println(sensorHigh);
  index = 0;
}
```

```
void loop() {

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A0);
    dataArray[index] = sensorValue;
    index = index + 1;
    delay(100);
  }
  //data output
  else{
    digitalWrite(greenLedPin,LOW);
    printIntArrayTable(dataArray,100);

    Serial.println("\n");
    delay(10000);
    index = 0; //to trigger new measurement
    Serial.println(" ");
  }
}
```

b) Ergänzung der Lösung aus Aufgabenteil **a)** um die Frage-Programmblöcke:

```
long frageIntArrSum(int valArray[], int dim){
  int index;
  long antwort;

  index = 0;
  antwort = 0;
  while(index < dim){
    antwort = antwort + valArray[index];
    index = index + 1;
  }
  return antwort;
}

float frageIntArrMean(int valArray[], int dim){
  long sum;
  float antwort;

  antwort = 0;
  sum = frageIntArrSum(valArray,dim);
  antwort = sum/(float)dim;
  return antwort;
}
```

…und Modifikation des `loop`-Programmblocks:

```
void loop() {
  long sum;
  float mean;

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A0);
    dataArray[index] = sensorValue;
    index = index + 1;
    delay(100);
  }
  //data output
  else{
    digitalWrite(greenLedPin,LOW);
    sum = frageIntArrSum(dataArray,100);
    mean = frageIntArrMean(dataArray,100);
    printIntArrayTable(dataArray,100);
    Serial.println("\n");
    Serial.print("sum: ");
    Serial.println(sum);
    Serial.print("mean: ");
    Serial.println(mean);
    delay(10000);
    index = 0; //to trigger new measurement
    Serial.println(" ");
  }
}
```

Zusatz (1) – Modifikationen zur Steuerung der Ausgabe am seriellen Monitor über den Druckknopfschalter:

```
//sensor management
int sensorValue;
int sensorLow = 1023;
int sensorHigh = 0;
//LEDs intended for signalling
int greenLedPin = 3;
int yellowLedPin = 5;
int redLedPin = 10;
//push button management
int buttonPin = 2;
int switchState = 0;
//array to store sensor values
int dataArray[100];
int index;
```

```
void loop() {
  long sum;
  float mean;

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A0);
    dataArray[index] = sensorValue;
    index = index + 1;
    delay(100);
  }
  //data output
  else{
    digitalWrite(greenLedPin,LOW);
    switchState = digitalRead(buttonPin);
    if(switchState == 1){
      sum = frageIntArrSum(dataArray,100);
      ...
```

Zusatz (2) – Leuchtdioden als Anzeige der Größe gemessener Werte:

```
//sensor management
int sensorValue;
int sensorLow = 1023;
int sensorHigh = 0;
int sensorRange = 0;
//LEDs intended for signalling
int greenLedPin = 3;
int yellowLedPin = 5;
int redLedPin = 10;
...
```

```
void setup() {
  Serial.begin(9600);
  pinMode(yellowLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  digitalWrite(greenLedPin, LOW);
  digitalWrite(redLedPin, LOW);
  //calibrate PhotoResistor
  //lighting the yellow LED to
  //indicate sensor calibration
  digitalWrite(yellowLedPin, HIGH);
  while(millis() < 5000){
    calibratePhotoResistor();
  }
  digitalWrite(yellowLedPin, LOW);
  Serial.print("sensor low: ");
  Serial.println(sensorLow);
  Serial.print("sensor high: ");
  Serial.println(sensorHigh);
  sensorRange = sensorHigh - sensorLow;
  switchState = 0;
  index = 0;
}
```

```
void loop() {
  long sum;
  float mean;

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A0);
    if(sensorValue > sensorLow + 2*sensorRange/3){
      digitalWrite(greenLedPin,LOW);
      digitalWrite(redLedPin,HIGH);
    }
    else{
      if(sensorValue > sensorLow + sensorRange/3){
        digitalWrite(greenLedPin,LOW);
        digitalWrite(yellowLedPin,HIGH);
      }
    }
    dataArray[index] = sensorValue;
    index = index + 1;
    delay(100);
    digitalWrite(redLedPin,LOW);
    digitalWrite(yellowLedPin,LOW);
    digitalWrite(greenLedPin,HIGH);
  }
  //data output
  else{
  ...
```

```
      index = 0; //to trigger new measurement
      Serial.println(" ");
    }
  }
}
```

c) Ergänzung der Lösung aus Aufgabenteil **b)** um den Frage-Programmblock:

```
int frageIntArrMinVal(int valArray[], int dim){
  int index;
  int antwort;

  antwort = valArray[0];
  index = 1; // !!
  while(index < dim){
    if(valArray[index] < antwort){
      antwort = valArray[index];
    }
    index = index + 1;
  }
  return antwort;
}
```

…und Modifikation des `loop`-Programmblocks:

```
void loop() {
  long sum;
  float mean;
  int minVal;

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A0);

    …

  //data output
  else{
    digitalWrite(greenLedPin,LOW);
    switchState = digitalRead(buttonPin);
    if(switchState == 1){
      sum = frageIntArrSum(dataArray,100);
      mean = frageIntArrMean(dataArray,100);
      minVal = frageIntArrMinVal(dataArray,100);
      printIntArrayTable(dataArray,100);

       …
       Serial.print("mean: ");
       Serial.println(mean);
       Serial.print("minimum value: ");
       Serial.println(minVal);
       delay(10000);
       index = 0; //to trigger new measurement
       Serial.println(" ");
    }
  }
}
```

d) Ergänzung der Lösung aus Aufgabenteil **b)** um den Frage-Programmblock:

```
int frageIntArrMaxVal(int valArray[], int dim){
  int index;
  int antwort;

  antwort = valArray[0];
  index = 1; // !!
  while(index < dim){
    if(valArray[index] > antwort){
      antwort = valArray[index];
    }
    index = index + 1;
  }
  return antwort;
}
```

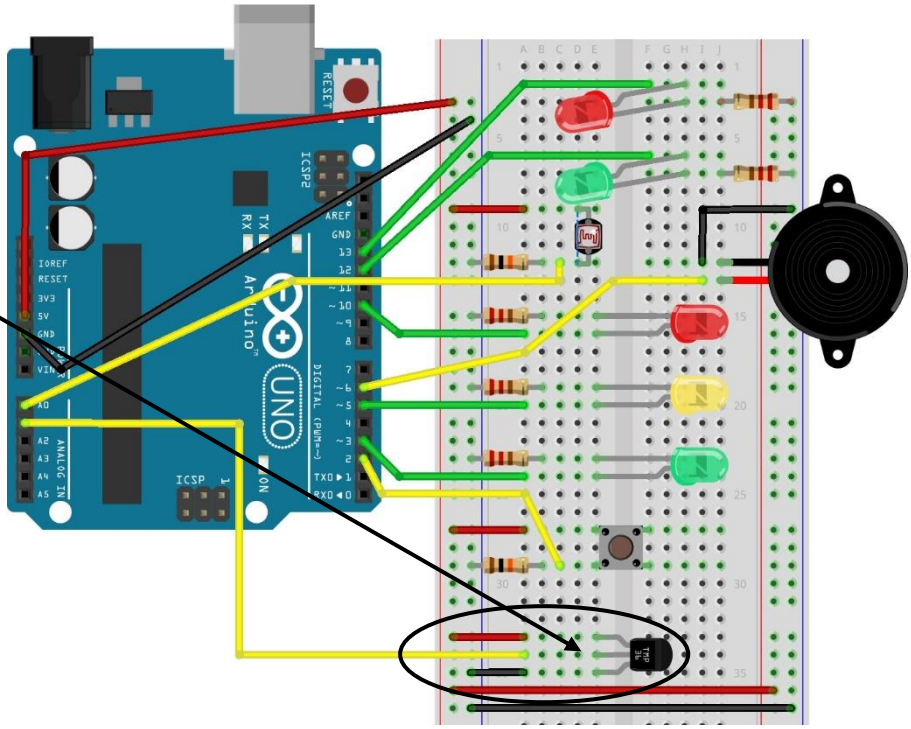
… und Modifikation des `loop`-Programmblocks:

```
void loop() {
  long sum;
  float mean;
  int minVal;
  int maxVal;

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A0);
    …
  //data output
  else{
    digitalWrite(greenLedPin,LOW);
    switchState = digitalRead(buttonPin);
    if(switchState == 1){
      sum = frageIntArrSum(dataArray,100);
      mean = frageIntArrMean(dataArray,100);
      minVal = frageIntArrMinVal(dataArray,100);
      maxVal = frageIntArrMaxVal(dataArray,100);
      …
      Serial.println(minVal);
      Serial.print("maximum value: ");
      Serial.println(maxVal);
      delay(10000);
      index = 0; //to trigger new measurement
      Serial.println(" ");
    }
  }
}
```

Aufgabe 2)

a) Mögliche Modifikation der „Ampelschaltung“ durch Hinzufügen des TMP36-Temperatursensors (mit Signalleitung am analogen Kontakt A1):



…und Anpassung des Programmcodes:

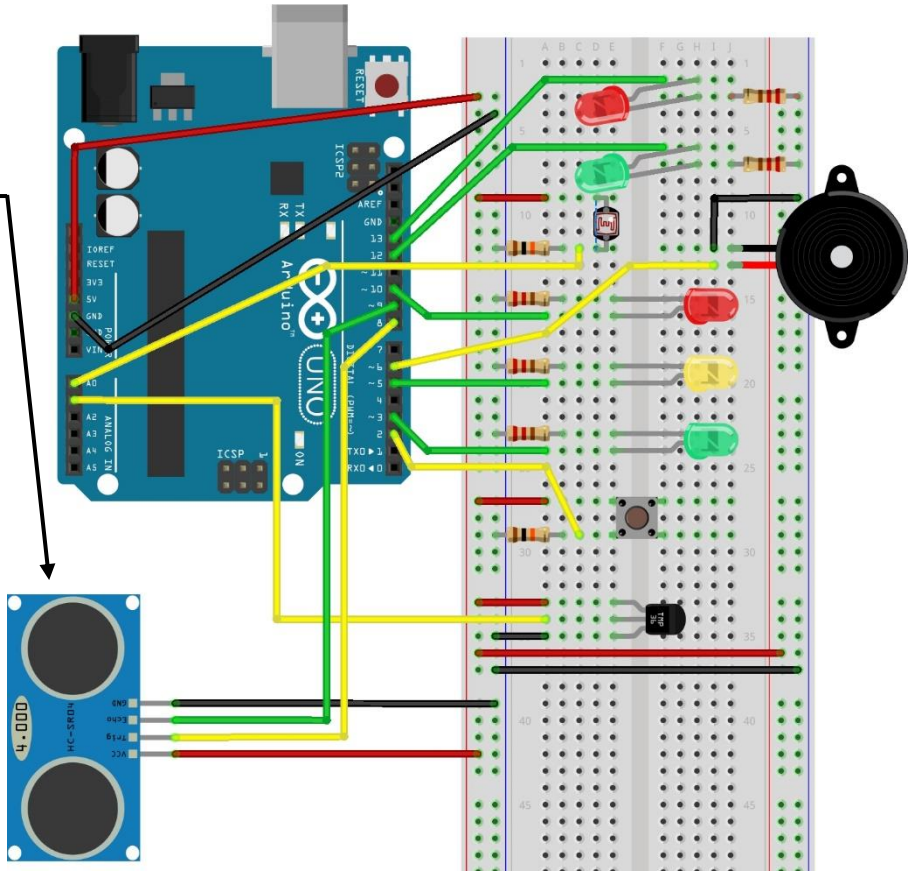
```
//sensor management
int sensorValue;
//LEDs intended for signalling
int greenLedPin = 3;
int yellowLedPin = 5;
int redLedPin = 10;
//push button management
int buttonPin = 2;
int switchState = 0;
//array to store sensor values
int dataArray[100];
int index;

void setup() {
  Serial.begin(9600);
  pinMode(yellowLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  digitalWrite(greenLedPin, LOW);
  digitalWrite(redLedPin, LOW);
  digitalWrite(yellowLedPin, LOW);
  switchState = 0;
  index = 0;
}
```

```
void loop() {
  float voltage;
  float tempC;
  long sum;
  float mean;
  int minVal;
  int maxVal;

  //collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A1);
    voltage = (sensorValue/1024.0)*5.0;
    tempC = (voltage - 0.5008)*100;
    if(tempC < 15){
      digitalWrite(greenLedPin,LOW);
      digitalWrite(yellowLedPin,HIGH);
    }
    else{
      if(tempC > 25){
        digitalWrite(greenLedPin,LOW);
        digitalWrite(redLedPin,HIGH);
      }
    }
    dataArray[index] = tempC;
    index = index + 1;
    delay(100);
    ...
```

b) Mögliche Modifikation der „Ampelschaltung“ durch Hinzufügen des HC-SR04-Ultraschallsensors (mit Trigger-Output am digitalen Kontakt 8 bzw. Echo-Input am digitalen Kontakt 9):



…und Anpassung des Programmcodes:

```
//sensor management
int triggerPin = 8;
int echoPin =9;
//LEDs intended for signalling
int greenLedPin = 3;
int yellowLedPin = 5;
int redLedPin = 10;
//push button management
int buttonPin = 2;
int switchState = 0;
//array to store sensor values
int dataArray[100];
int index;
```

```
void setup() {
  Serial.begin(9600);
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(yellowLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  digitalWrite(greenLedPin, LOW);
  digitalWrite(redLedPin, LOW);
  digitalWrite(yellowLedPin, LOW);
  switchState = 0;
   ıdex = 0;
```

```
void loop() {
  long timePassed;
  float distance;
  long sum;
  float mean;
  int minVal;
  int maxVal;

//collecting data
  if(index < 100){
  //lighting the green LED
  //to indicate cata collection
  digitalWrite(greenLedPin,HIGH);
  digitalWrite(triggerPin,LOW);
  delay(5);
  digitalWrite(triggerPin,HIGH);
  delay(10);
  digitalWrite(triggerPin,LOW);
  timePassed = pulseIn(echoPin,HIGH);
  distance = timePassed*0.03432/2;
    if(distance < 100){
      digitalWrite(greenLedPin,LOW);
      digitalWrite(yellowLedPin,HIGH);
    }
    else{
      if(distance > 200){
        digitalWrite(greenLedPin,LOW);
        digitalWrite(redLedPin,HIGH);
      }
    }
    if(distance <= 255){
      dataArray[index] = distance;
      index = index + 1;
      delay(100);
    }
    digitalWrite(redLedPin,LOW);
    digitalWrite(yellowLedPin,LOW);
    digitalWrite(greenLedPin,HIGH);
  }
  //data output
  else{
    digitalWrite(greenLedPin,LOW);
    switchState = digitalRead(buttonPin);
    if(switchState == 1){
      sum = frageIntArrSum(dataArray,100);
      mean = frageIntArrMean(dataArray,100);
      ...
```

c) Anpassung des Programmcodes durch Deklaration der für die Temperaturmessung benötigten Variablen (vgl. Lösung zu Aufgabenteil **a)**), Neucodierung der Berechnungsformel für den Wert der Variablen distance:

```
//collecting data
  if(index < 100){
    //lighting the green LED
    //to indicate cata collection
    digitalWrite(greenLedPin,HIGH);
    sensorValue = analogRead(A1);
    voltage = (sensorValue/1024.0)*5.0;
    tempC = (voltage - 0.5008)*100;
    digitalWrite(triggerPin,LOW);
    delay(5);
    digitalWrite(triggerPin,HIGH);
    delay(10);
    digitalWrite(triggerPin,LOW);
    timePassed = pulseIn(echoPin,HIGH);
    distance = timePassed*(0.03315 + 0.6*tempC)/2;
    if(distance < 100){
    ...
```

Arbeitsanregungen zum Codieren und Verwenden von Programmbibliotheken

Aufgabe 3)

a) Vervollständigtes Codegerüst der Programbibliothek **morse.cpp**:

```
#include "Arduino.h"
#include "morse.h"

void shortSignal(int pin){
  digitalWrite(pin,HIGH);
  delay(200);
  digitalWrite(pin,LOW);
  delay(200);
}

void longSignal(int pin){
  digitalWrite(pin,HIGH);
  delay(600);
  digitalWrite(pin,LOW);
  delay(200);
}

void letterD(int pin){
  int counter = 1;

  longSignal(pin);
  while(counter <= 2){
    shortSignal(pin);
    counter = counter + 1;
  }
}
```

```
void letterA(int pin){
  shortSignal(pin);
  longSignal(pin);
}

void letterS(int pin){
  int counter = 1;

  while(counter <= 3){
    shortSignal(pin);
    counter = counter + 1;
  }
}

void letterP(int pin){
  int counter = 1;
  shortSignal(pin);
  while(counter <= 2){
    longSignal(pin);
    counter = counter + 1;
  }
  shortSignal(pin);
}

void letterT(int pin){
  longSignal(pin);
}
```

…samt zugehöriger Header-Datei und deren Einbindung in ein Programmierprojekt:

```
#ifndef morse_h
#define morse_h

#include "Arduino.h"

// function prototypes

 void letterA(int pin);
 void letterD(int pin);
 void letterP(int pin);
 void letterS(int pin);
 void letterT(int pin);

#endif
```

```
//using a single LED to morse Das passt
//version with library

#include <morse.h>

int ledPin = 10;
int charPin = 5;
int wordPin = 3;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(charPin, OUTPUT);
  pinMode(wordPin, OUTPUT);
  digitalWrite(ledPin,LOW);
  digitalWrite(charPin,LOW);
  digitalWrite(wordPin,LOW);
}

void loop() {
```

b) Erweiterte Datei **morse_ext.h** bzw. Ausschnitt aus der erweiterten Datei **morse_ext.cpp** – diese müssen in einem Unterordner namens **Morse_ext** gespeichert werden:

```
#ifndef morse_ext_h
#define morse_ext_h

#include "Arduino.h"

// function prototypes

 void letterA(int pin);
 void letterB(int pin);
 void letterC(int pin);
 void letterD(int pin);
 void letterE(int pin);
 void letterF(int pin);
 void letterG(int pin);
 void letterH(int pin);
 void letterI(int pin);
 void letterJ(int pin);
 void letterK(int pin);
 void letterL(int pin);
 void letterM(int pin);
 void letterN(int pin);
 void letterO(int pin);
 void letterP(int pin);
 void letterQ(int pin);
 void letterR(int pin);
 void letterS(int pin);
 void letterT(int pin);
 void letterU(int pin);
 void letterV(int pin);
 void letterW(int pin);
 void letterX(int pin);
 void letterY(int pin);
 void letterZ(int pin);

#endif
```

```
#include "Arduino.h"
#include "morse_ext.h"

void shortSignal(int pin){
  digitalWrite(pin,HIGH);
  delay(200);
  digitalWrite(pin,LOW);
  delay(200);
}

void longSignal(int pin){
  digitalWrite(pin,HIGH);
  delay(600);
  digitalWrite(pin,LOW);
  delay(200);
}

void letterA(int pin){
  shortSignal(pin);
  longSignal(pin);
}

void letterB(int pin){
  int counter = 1;

  longSignal(pin);
  while(counter <= 3){
    shortSignal(pin);
  }
}

void letterC(int pin){
  int counter = 1;

  while(counter <= 2){
    longSignal(pin);
    shortSignal(pin);
```

...

Aufgabe 4)

Code der Header-Datei **iArrStatistics.h**:

```
#ifndef iArrStatistics_h
#define iArrStatistics_h

#include "Arduino.h"

// function prototypes

 long frageIntArrSum(int intArray[], int dim);
 int frageIntArrMinVal(int intArray[], int dim);
 int frageIntArrMaxVal(int intArray[], int dim);
 float frageIntArrMean(int intArray[], int dim);
 float frageIntArrStdDev(int intArray[], int dim);

#endif
```

...vervollständigtes Codegerüst der Programbibliothek **iArrStatistics.cpp**:

```
#include "Arduino.h"
#include "iArrStatistics.h"

long frageIntArrSum(int intArray[], int dim)
{
  int index;
  long result;

  index = 0;
  result = 0;
  while(index < dim){
    result = result + intArray[index];
    index = index + 1;
  }
  return result;
}

int frageIntArrMinVal(int intArray[], int dim)
{
  int index;
  int mini;

  mini = intArray[0];
  index = 1;
  while(index < dim){
    if(intArray[index] < mini){
      mini = intArray[index];
    }
    index = index + 1;
  }
  return mini;
}
```

```
int frageIntArrMaxVal(int intArray[], int dim)
{
  int index;
  int maxi;

  maxi = intArray[0];
  index = 1;
  while(index < dim){
    if(intArray[index] > maxi){
      maxi= intArray[index];
    }
    index = index + 1;
  }
  return maxi;
}

float frageIntArrMean(int intArray[], int dim)
{
  float result;

  result = frageIntArrSum(intArray,dim)/(float) dim;
  return result;
}
```

```
float frageIntArrStdDev(int intArray[], int dim)
{
  int index;
  float m;
  float result;

  m = frageIntArrMean(intArray,dim);
  result = 0;
  index = 0;
  while(index < dim){
    result = result + (intArray[index] - m) * (intArray[index] - m);
    index = index + 1;
  }
  result = result/dim;
  result = sqrt(result);
  return result;
}
```

und Einbindung der Header-Datei in das Programmierprojekt zur Distanzmessung mit dem Ultraschallsensor (Auszug):

```
#include <iArrStatistics.h>

long timePassed;
float distance;
int triggerPin = 7;
int echoPin = 6;
int buttonPin = 2;
int switchstate = 0;
int dataArray[100];
int index;

void printIntArrayTable(int valArray[],int dim){
  int index;

  index = 0;
  Serial.println("");
  while(index < dim){
  ...
```

```
void setup() {
  Serial.begin(9600);
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buttonPin, INPUT);
  digitalWrite(buttonPin,LOW);
  switchstate = 0;
  index = 0;
}

void loop() {
  long s;
  float m;
  int mini;
  int maxi;
  float sD;

  //collecting data
  ...
```

```
...
}
else{
  switchstate = digitalRead(buttonPin);
  if(switchstate == 1){
    printIntArrayTable(dataArray,100);
    s=frageIntArrSum(dataArray,100);
    m=frageIntArrMean(dataArray,100);
    mini = frageIntArrMinVal(dataArray,100);
    maxi = frageIntArrMaxVal(dataArray,100);
    sD=frageIntArrStdDev(dataArray,100);

    Serial.print("sum: ");
    Serial.print(s);
...
```