Listing1:

* Funktionsrahmen wird erzeugt
* x<0, also wird f erneut aber mit x = 42592 aufgerufen und es wir ein neuer Funktionsrahmen über dem ersten erzeugt
* Funktion wird mit x= 4259 aufgerufen, neuer Funktionsrahmen über den beiden anderen
* Funktion wird mit x= 425 aufgerufen, neuer Funktionsrahmen über den anderen
* Funktion wird mit x= 42aufgerufen, neuer Funktionsrahmen über den anderen
* Funktion wird mit x= 4 aufgerufen, neuer Funktionsrahmen über den anderen
* Funktion wird mit x= 0 aufgerufen, neuer Funktionsrahmen über den anderen
* Funkton mit x= 0 gibt 0 zurück, Funktionsrahmen wird entfernt
* Nächste Funktion gibt 0+1 zurück, ihr funktionsrahmen wird entfernt
* Nächste Funktion gibt 1+1 zurück, ihr funktionsrahmen wird entfernt
* Nächste Funktion gibt 2+1 zurück, ihr funktionsrahmen wird entfernt
* Nächste Funktion gibt 3+1 zurück, ihr funktionsrahmen wird entfernt
* Nächste Funktion gibt 3+1 zurück, ihr funktionsrahmen wird entfernt
* Die unterste Funktion gibt 5+1 = 5 zurück, auch ihr funktionsrahmen wird entfernt

|  |
| --- |
| * f(0)-----------------0 |
| * f(4)-----------------0+1 |
| * f(42)---------------1+1 |
| * f(425)-------------2+1 |
| * f(4259)-----------3+1 |
| * f(42592)---------4+1 |
| * f(-42592)--------5+1 |

* Das Programm scheint die Länge einer Zahl zu bestimmen.
* Bei 0 wird aufgrund der Rekursion eine 0 zurückgegeben obwohl eine 1 sinnvoller wäre
* Verbesserte Version:

int f(int x){

int l=1;

if (x < 0)

return f(-x) + 1;

while(x>9){

l++;

x/=10;

}

return l;

}

Listing2:

* Funktionsrahmen mit lokalen Variablen wird erzeugt
* So lange wie str[links] ein Leerzeichen ist, wird die die Variable links hochgezählt
  + string[0] wird abgerufen und ist ‚ ‘
  + Usw. bis string[3] schließlich ein ‚T‘ ist
* Der Rahmen der ersten while schleife wird geschlossen
* Das gleiche wird nun von rechts aus gemacht. Dabei wird das \0 ignoriert
* Schließlich wird der Wert von links und rechts zurückgegeben und der Funktionsrahmen entfernt
* Das Programm scheint die Leerzeichen zu zählen. Dabei werden nur Leerzeichen vor- und hinter Buchstabenfolgen erfasst und nicht innerhalb.

Verbesserung:

int g(char\* str) {

int len = strlen(str);

int n = 0;

for (int I = 0; I <= len – 1;i++)

{

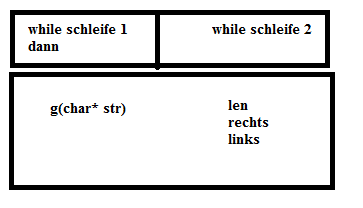
if(str[i] == ‘ ‘)

n++;

}

return n;

}



Listing 3:

* Funktionsrahmen mit Variablen wird erzeugt
* While Schleife mit eigenen Variablen wird erzeugt
* While Schleife mit eigenen Variablen wird erzeugt
* While Schleife läuft 3 mal durch und addiert 3\*3 zu x, X = 9
* Äußere while Schleife läuft 3 mal durch und multipliziert ergebnis immer mit x: ergebnis = 9,81,729
* Ergebnis wird zurückgegeben und Funktionsrahmen geschlossen

Die Funktion rechnet (z^2)^e

Verbessert:

int h( int z, int e) {

int zquadrat = z\*z;

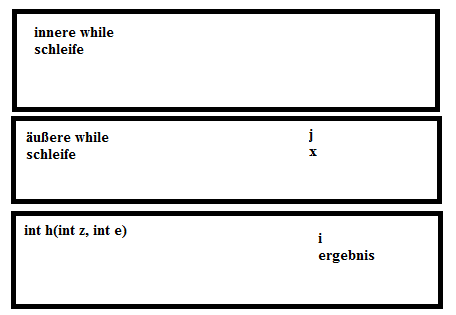
int ergebnis = zquadrat;

for (int i = 1; i < e;i++)

ergebnis \*= zquadrat;

return ergebnis;

}



Aufgabe2:

#include <LiquidCrystal.h>

LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

byte leer[8] = {

B11111,

B10001,

B10001,

B10001,

B10001,

B10001,

B10001,

B11111 };

void balkenDiagramm(int zeile,int spalte, int wert,int minimum,int maximum);

void aufgabeA();

void aufgabeB();

void aufgabeC();

void aufgabeD();

void aufgabeE();

double getDist();

long fetchPulse();

void sendTrigger();

void setup() {

pinMode(A0, INPUT);

pinMode(4, INPUT\_PULLUP);

pinMode(2, OUTPUT);

pinMode(3, INPUT);

lcd.begin(16, 2);

lcd.createChar(0,leer);

}

void loop() {

//aufgabeA();

//aufgabeB();

//aufgabeC();

//aufgabeD();

aufgabeE();

//Aufgabe f ist in FunktionBlakendiagramm integriert

}

void aufgabeE()

{

int maximumL,minimumL,maximum,minimum,wert;

boolean funktion = false;

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Kalibrierung");

lcd.setCursor(0,1);

lcd.print("Ultraschall");

delay(1500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("\176Wert 1");

lcd.setCursor(1,1);

lcd.print("Wert 2");

while(digitalRead(4)){

}

lcd.setCursor(0,1);

lcd.print("\176");

minimum = getDist();

delay(200);

while(!digitalRead(4));

while (digitalRead(4)){

}

maximum = getDist();

delay(200);

while(!digitalRead(4));

if (minimum > maximum)

{

int buffer = maximum;

maximum = minimum;

minimum = buffer;

}

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Kalibrierung");

lcd.setCursor(0,1);

lcd.print("Lichtsensor");

delay(1500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("\176Wert 1");

lcd.setCursor(1,1);

lcd.print("Wert 2");

while(digitalRead(4)){

}

lcd.setCursor(0,1);

lcd.print("\176");

minimumL = analogRead(0);

delay(200);

while(!digitalRead(4));

while (digitalRead(4)){

}

maximumL = analogRead(0);

delay(200);

while(!digitalRead(4));

if (minimumL > maximumL)

{

int buffer = maximumL;

maximumL = minimumL;

minimumL = buffer;

}

lcd.clear();

uint32\_t zeit;

while(1)

{

lcd.setCursor(0,0);

if (!digitalRead(4) && millis() - zeit > 200)

{

funktion = !funktion;

zeit = millis();

}

if (!funktion)

{

wert = getDist();

lcd.print("Ultraschall:");

balkenDiagramm(1,0,wert,minimum,maximum);

}

else

{

lcd.print("Licht: ");

wert = analogRead(0);

balkenDiagramm(1,0,wert,minimumL,maximumL);

}

lcd.setCursor(13,0);

lcd.print(wert);

lcd.print(" ");

}

}

void aufgabeD()

{

int maximum,minimum,wert;

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Kalibrierung");

delay(1000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("N\341hern und");

lcd.setCursor(0,1);

lcd.print("Knopf Dr\365cken");

delay(1500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Dann entfernen und");

lcd.setCursor(0,1);

lcd.print("Knopf Dr\365cken");

delay(1500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("\176Wert 1");

lcd.setCursor(1,1);

lcd.print("Wert 2");

while(digitalRead(4)){

}

lcd.setCursor(0,1);

lcd.print("\176");

minimum = getDist();

while(!digitalRead(4));

while (digitalRead(4)){

}

maximum = getDist();

while(!digitalRead(4));

if (minimum > maximum)

{

int buffer = maximum;

maximum = minimum;

minimum = buffer;

}

lcd.clear();

while(1)

{

wert = getDist();

lcd.setCursor(0,0);

lcd.print(wert);

lcd.print(" ");

balkenDiagramm(1,0,wert,minimum,maximum);

}

}

void aufgabeC()

{

int maximum,minimum,wert;

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Kalibrierung");

delay(1000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Abdunkeln +");

lcd.setCursor(0,1);

lcd.print("Knopf Dr\365cken");

delay(1500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Dann erhellen +");

lcd.setCursor(0,1);

lcd.print("Knopf Dr\365cken");

delay(1500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("\176Wert 1");

lcd.setCursor(1,1);

lcd.print("Wert 2");

while(digitalRead(4)){

}

lcd.setCursor(0,1);

lcd.print("\176");

minimum = analogRead(0);

while(!digitalRead(4));

while (digitalRead(4)){

}

maximum = analogRead(0);

while(!digitalRead(4));

if (minimum > maximum)

{

int buffer = maximum;

maximum = minimum;

minimum = buffer;

}

lcd.clear();

while(1)

{

wert = analogRead(0);

lcd.setCursor(0,0);

lcd.print(wert);

lcd.print(" ");

balkenDiagramm(1,0,wert,minimum,maximum);

}

}

void aufgabeB()

{

int maximum,minimum,wert;

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Kalibrierung");

delay(1000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Abdunkeln und");

lcd.setCursor(0,1);

lcd.print("Knopf Dr\365cken");

while(digitalRead(4)){

}

minimum = analogRead(0);

while(!digitalRead(4));

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Erhellen und");

lcd.setCursor(0,1);

lcd.print("Knopf Dr\365cken");

while (digitalRead(4)){

}

maximum = analogRead(0);

while(!digitalRead(4));

lcd.clear();

while(1)

{

wert = analogRead(0);

lcd.setCursor(0,0);

lcd.print(wert);

lcd.print(" ");

balkenDiagramm(1,0,wert,minimum,maximum);

}

}

void aufgabeA()

{

int wert = analogRead(0);

lcd.setCursor(0,0);

lcd.print(wert);

lcd.print(" ");

balkenDiagramm(1,0,wert,0,1023);

}

void balkenDiagramm(int zeile, int spalte, int wert, int minimum, int maximum)

{

lcd.setCursor(spalte,zeile);

int intervalllaenge =(maximum-minimum)/7;

int anzahlRauten = wert/intervalllaenge;

if(anzahlRauten > 6)

anzahlRauten = 6;

for(int i = 0; i < anzahlRauten;i++)

{

lcd.print("\377"); //volle Kästchen gibt es bereits auf dem display

}

for(int i = 0 ; i < 6 - anzahlRauten ; i++)

{

//lcd.write(0); seit Arduino 1.0 nicht mehr so funktionsfähig

lcd.print((char)0);

}

}

void sendTrigger()

{

digitalWrite(2, HIGH);

delayMicroseconds(10);

digitalWrite(2, LOW);

}

long fetchPulse()

{

uint32\_t zeit = micros();

while (!digitalRead(3))

{

if (micros() - zeit > 23324) //8m (4 hin 4 zurück) /343 m/s = 0.0233236s also ~23324 us

{

return -1;

}

}

zeit = micros();

while (digitalRead(3))

{

if (micros() - zeit > 23324)

{

return -1;

}

}

return micros() - zeit;

}

double getDist()

{

sendTrigger();

double abstand = (fetchPulse()\*343.0)/20000.0;

return abstand;

}