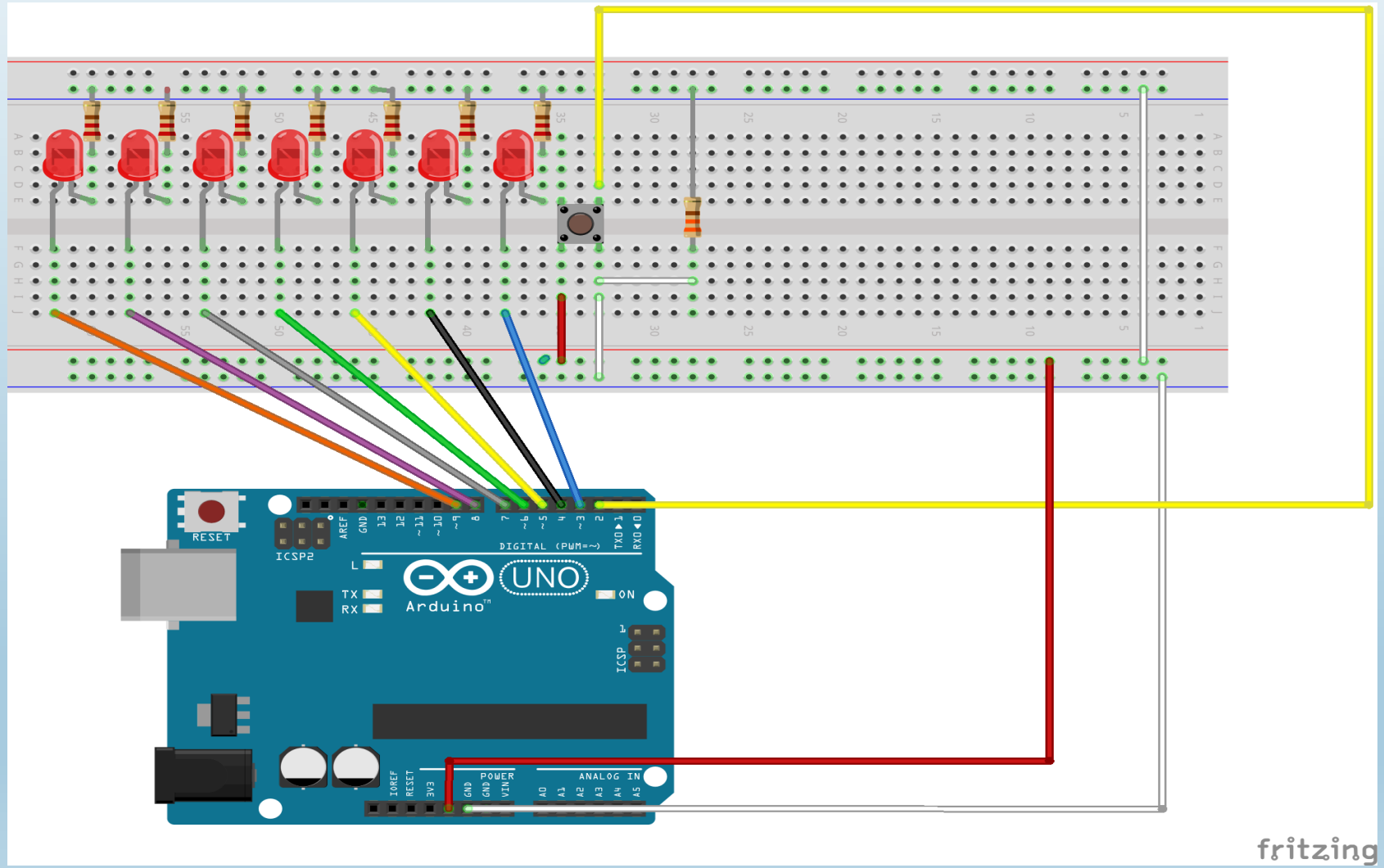
Reaktometer



Gliederung

- Struktogramm
- Hauptprogramm

  Reaktometer.ino
- .H File

  Reaktometer.h
- .CPP File

  Konstruktor

  Funktionen wait_until_release() und reset()

  Struktogramm Funktion menuauswahl ()

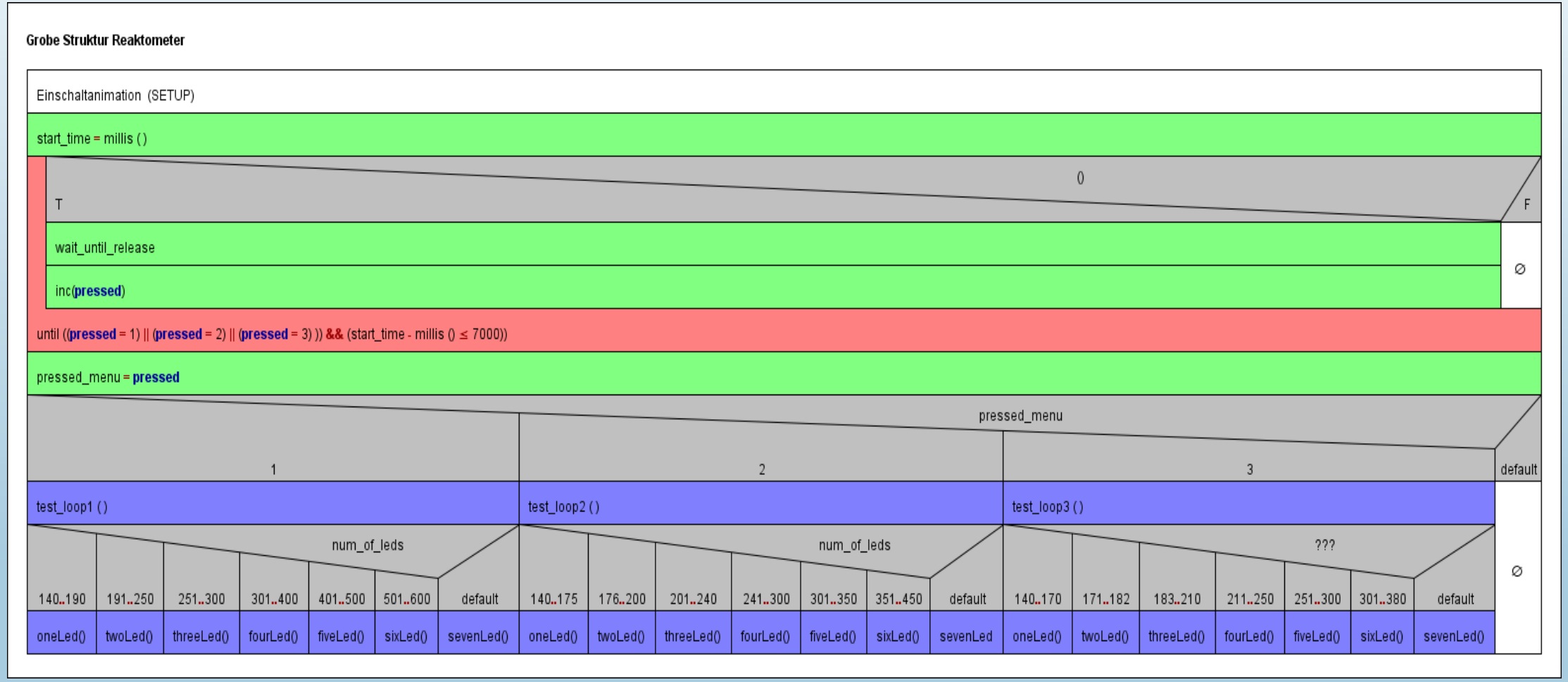
  Funktion menuauswahl ()

  Struktogramm Funktion test_loop ()

  Funktion test_loop ()

  Fortsetzung test_loop ()

Struktogramm



Hauptprogramm

Reaktometer.ino

- Pins, LED‘s, Button sowie die „Pressed-Zähler“ Variablen werden bei Erzeugung des Objekts definiert und initialisiert für Hauptprogramm

- Ausführen des Hauptprogramms

```
#include "Reaktometer.h"

// Objekt erzeugen
RT reaktometer;

void setup() {
  reaktometer.animation();
}

void loop() {
  reaktometer.menuauswahl();
}
```

.H File

Reaktometer.h

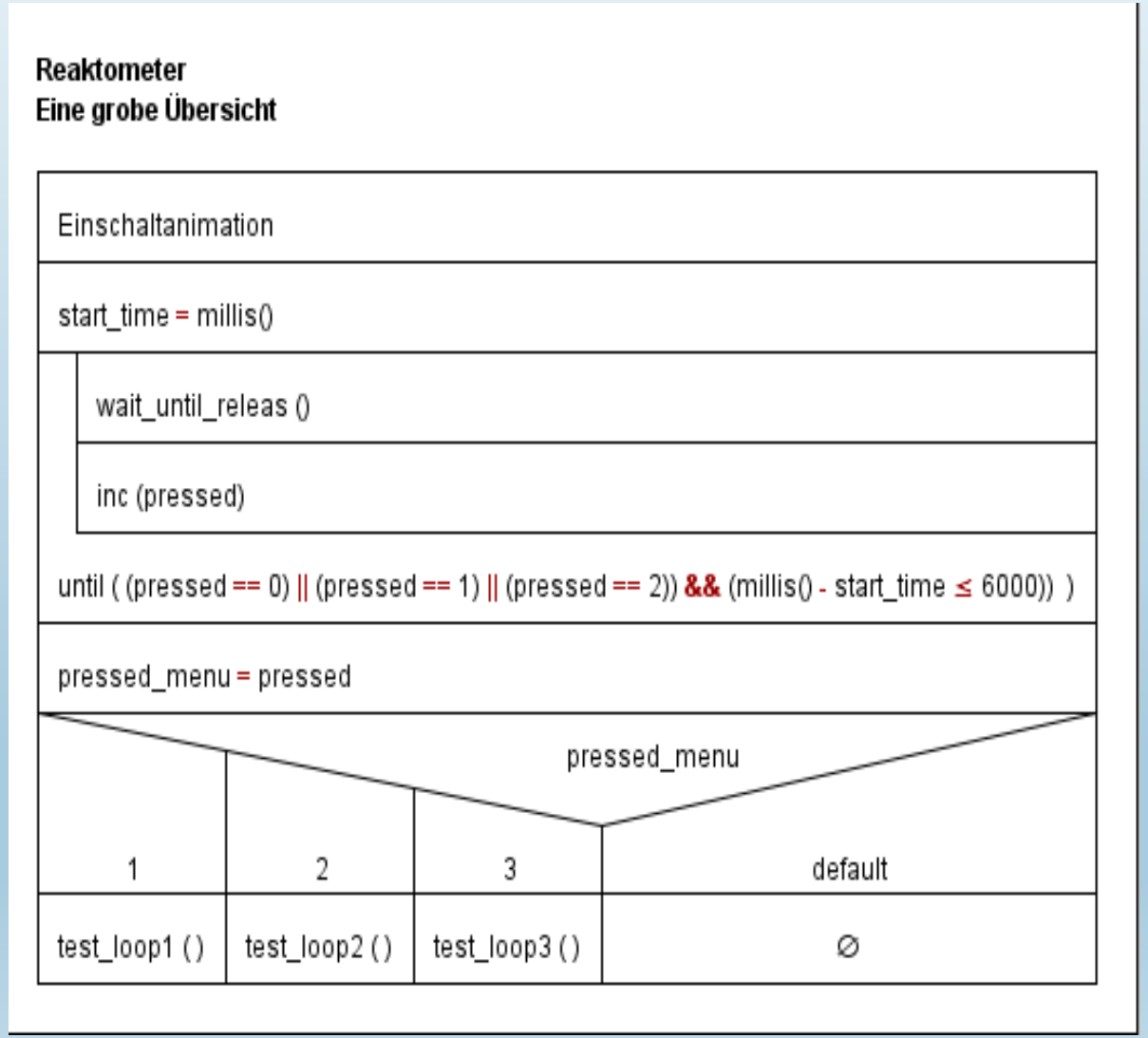
```
#ifndef Reaktometer_h
#define Reaktometer_h
#include "Arduino.h"
class RT {
  public:
    RT();
    void animation();
    void menuauswahl();
  private:
    void testAnimation();
    void wait_until_release();
    void test_loop1();
    void test_loop2();
    void test_loop3();
    void reset();
    void oneLed();
    void twoLed();
    void threeLed();
    void fourLed();
    void fiveLed();
    void sixLed();
    void sevenLed();
    int led1;
    int led2;
    int led3;
    int led4;
    int led5;
    int led6;
    int led7;
    int buttonPin;
    long start_time;
};
#endif
```

.CPP File

Konstruktor

```
// Konstruktor
RT::RT() {
  //Pins definieren
  led1 = 3;
  led2 = 4;
  led3 = 5;
  led4 = 6;
  led5 = 7;
  led6 = 8;
  led7 = 9;
  buttonPin = 2;
  //LEDs und Button initialisieren
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led4, OUTPUT);
  pinMode(led5, OUTPUT);
  pinMode(led6, OUTPUT);
  pinMode(led7, OUTPUT);
  pinMode(buttonPin, INPUT);
}
```

Struktogramm Funktion menuauswahl ()



Funktion menuauswahl ()

Fälle

Taster wird nicht gedrückt

- pressed = 0 und wenn Zeitdiff. Größer 6 Sekunden -> kein Fall tritt ein –> Default -> neuer Durchgang in do While Schleife

Taster wird gedrückt

- Pressed = 1 entweder wartet Benutzer bis Zeitdiff. < 6 Sekunden um ist, und wählt somit case 1 aus oder er drückt ein weiteres Mal und landet in case 2 nachdem der Counter abgelaufen ist.

Taster wurde bereits zwei Mal gedrückt

- Benutzer drückt ein drittes Mal pressed =3 -> False vs True -> False geht aus Schleife raus und in case 3
- Schleife kann schon vor 6 Sek verlassen werden

```
void RT::menuauswahl() {
  Serial.begin(9600);
  randomSeed(analogRead(0));
  start_time = millis();
  do {
    if (digitalRead(buttonPin) == HIGH) {
      wait_until_release();
      pressed = pressed + 1;
    }
  }
  while (((pressed == 0) || (pressed == 1) || (pressed == 2)) && (millis() - start_time <= 6000));
  pressed_menu = pressed;

  switch (pressed_menu) {
    case 1 :
      oneLed();
      delay(3000);
      reset();
      delay(500);
      test_loop1();
      delay(4000);
      reset();
      break;
    case 2 :
      twoLed();
      delay(3000);
      reset();
      delay(500);
      test_loop2();
      delay(4000);
      reset();
      break;
```

Funktionen wait_until_release() und reset()

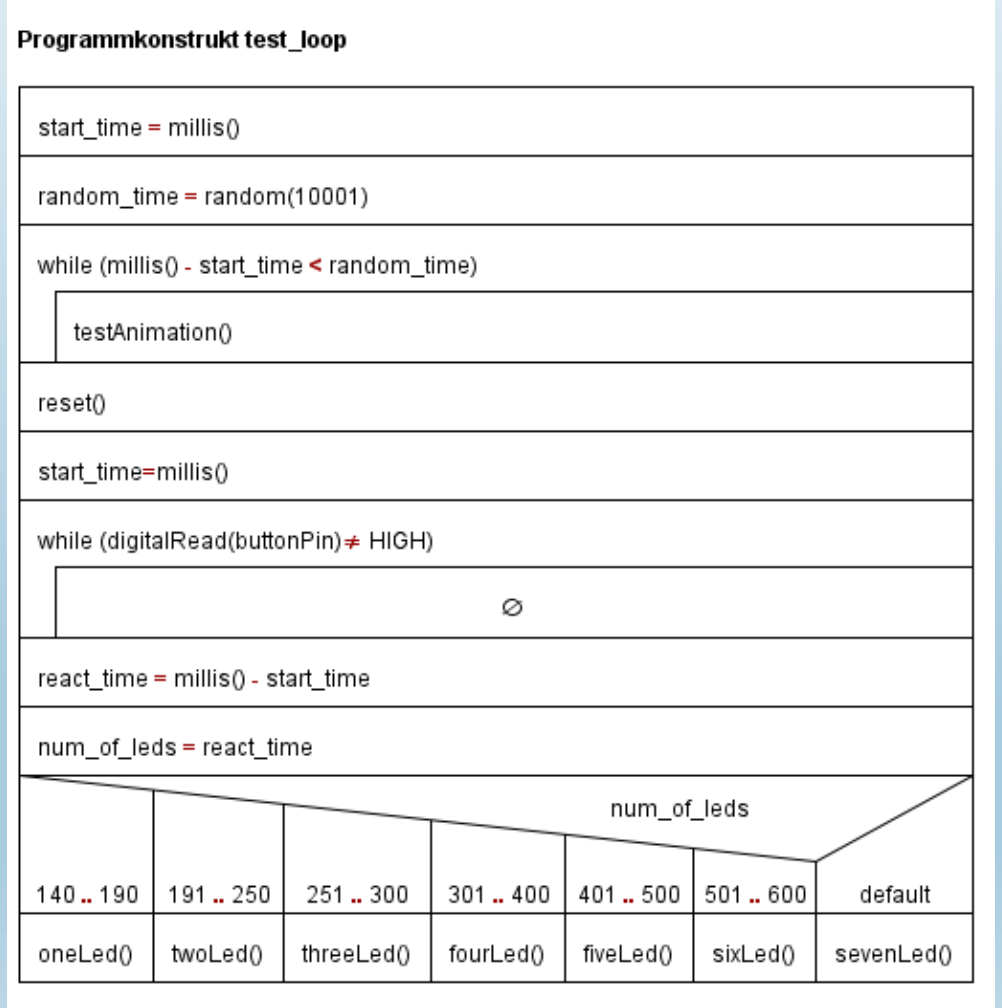
- Funktion, um den Taster beliebig lang zu drücken

```
// Funktion, die langes halten ermöglicht
void RT::wait_until_release() {
  while (digitalRead(buttonPin) != LOW) {
    // Serial.println(digitalRead(buttonPin));
    }
  }
```

- LED‘s zurücksetzen und iteratives Initialisieren der „Pressed-Zähler“ vor jedem Durchlauf

```
void RT::reset () {
  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);
  digitalWrite(led4, LOW);
  digitalWrite(led5, LOW);
  digitalWrite(led6, LOW);
  digitalWrite(led7, LOW);
  pressed = 0;
  pressed_menu = 0;
}
```

Struktogramm Funktion test_loop ()



Funktion test_loop ()

Realisiert Zeitmessung sowie die Ausgabe des Ergebnisses

Anzeigedauer der Testanimation wird über Zufallszahl gesteuert

Nach reset() startet Timer

Werte zuweisen sobald die Schleife verlassen wird

```
void RT::test_loop3() {
  start_time = millis();
  random_time = random(10001);
  Serial.println(random_time);
  while ((millis() - start_time < random_time)) {
    testAnimation();
  }
  reset();
  start_time = millis();
  while (digitalRead(buttonPin) != HIGH) {
  }
  react_time = millis() - start_time;
  Serial.println(react_time);
  num_of_leds = react_time;
```

Fortsetzung test_loop ()

- Case-Anweisung um Reaktionszeit mittels LEDs anzuzeigen
- Wertebereich Case 140 .. 160
- Wertebereich default
- Jedem Fall wird ein Schwierigkeitsgrad zugeordnet
- test_loop1() | test_loop2() | test_loop3()

```
switch (num_of_leds) {
  case 140 ... 160:
    oneLed();
    break;

  case 161 ... 170:
    twoLed();
    break;

  case 171 ... 190:
    threeLed();
    break;

  case 191 ... 250:
    fourLed();
    break;

  case 251 ... 300:
    fiveLed();
    break;

  case 301 ... 380:
    sixLed();
    break;
  default:
    sevenLed();
    break;
```

ENDE