1. Overflow sichtbar machen

Den Overflow aus Aufgabe 5b), Blatt 1, können wir nun sichtbar machen und in Zeitlupe beobachten. Initialisiere den SM und programmiere eine for-Schleife, in welcher eine integer-Variable (i) beginnend von z.B. 32760 hochgezählt wird. Setze danach ein delay() auf 1 Sec. damit der Vorgang langsam abläuft.
Mit **Serial.println(i);** kannst du den Overflow genau beobachten.

2. Die for-Schleife

Die LED soll wieder 5-mal blinken. Diesmal verwenden wir jedoch eine **for-Schleife**. Die Zählvariable „n" könnten wir auch in der for-Schleife definieren. Syntax: **for(**int n=0; n<6; n++) **{**mach irgendwas**}**.
Möchte man den Wert jeweils um 2 erhöhen, müsste man n=n+2 schreiben.

a) setze die Schleife in das Setup
b) in den Loop

Beobachte jeweils die LED und versuche den Unterschied zu erklären. Nun definieren wir n mit int n=0; vor dem Setup im globalen Bereich und schreiben in der for-Schleife anstelle **int n=0;** nur **n;** Teste, was passiert und begründe es.

3. Die while-Schleife

Syntax: **while(expression) { statement(s) }**, Beispiel: int n = 0; while(n < 200) { do something; n++; }
Realisiere den Blink-Sketch (LED soll 5-mal blinken) mit Hilfe einer „While-Schleife". Tipp: Es ist empfehlenswert, den Zähler mit n-- runter zählen zu lassen.
Erweitere den Sketch, so dass nach 5-mal langsam Blinken ein schnelles Dauerblinken entsteht.

4. Schöne Töne: Dreiklang

Mit der Anweisung **tone(9, x, y);** erhältst du eine Tonausgabe, hier an Pin9, Frequenz x, Dauer y. Dazu ist ein Lautsprecher an PIN9 und GND anzuschließen.
a) Erzeuge im Setup einen gut klingenden Dreiklang.
b) Erzeuge im Loop mit Hilfe einer While-Schleife einen Dreiklang, der dreimal ertönt.

5. Mit Taster (Button) schalten.....

a) Lade den Sketch „Button" (Beispiele-02Digital) auf den Arduino. Trenne dann das USB Kabel vom Arduino, installierte die LED an Pin13 und den Button (Taster) an Pin2. Verwende den Shield-Button der LOW wird, wenn man den Taster drückt. Daher muss der INPUT noch mit PULLUP ergänzt werden. Die LED soll bei gedrücktem Button leuchten.
b) Prüfe anschließend, ob folgender Button-mini-Sketch auch funktioniert. Vergleiche mit dem original-Sketch.

Hier wird wieder der Shield-Button verwendet.

```
void setup() {
  pinMode(13, OUTPUT);
  pinMode(2, INPUT_PULLUP);
}
void loop() {
  if (digitalRead(2)==LOW) {
    digitalWrite(13, HIGH);
  } else {
    digitalWrite(13, LOW);
  }
}
```

c) Verwende nun den **Joystick-Taster** und erweitere den Sketch so, dass beim Drücken des Tasters eine sehr kurze akustische Rückmeldung aus drei aufeinanderfolgenden Tönen erfolgt und die LED nach einer Zeitverzögerung von ca. 4 Sekunden ausgeht.

d) Für Profis: Versuche nun den Sketch nochmals zu erweitern, so dass die LED erst nach dreimal Drücken des Tasters für 4 Sekunden leuchtet.

Lade zum Schluss wieder das „BareMinimum" auf den Arduino!

6. Mit „Fade" dimmen, Mischfarben erzeugen

a) Öffne den Sketch „fade", teste ihn und versuche diesen zu verstehen. Verändere die Parameter und vergleiche. Mache dir klar, was **PWM** bedeutet. Schließe dazu parallel zur LED ein Oszilloskop an und beobachte das Display. Das schwarze Kabel muss dabei an GND angeschlossen werden. Eventueller Zusatz: Initialisiere den Seriellen Monitor und lasse den Logik-Wert der LED auf dem Laptop darstellen.

b) Erweitere den **fade**-Sketch für die **3-Farben-LED**.

Dabei sollte jede Farbe extra in der Intensität hoch- und runtergefahren werden. „Bunt" wird das Licht nur, wenn die 3 LEDs nicht synchron laufen. Dies gelingt am besten, wenn **fadeAmount** für alle 3 Farben im Bereich von 3 bis 6 zufällig gewählt werden: **random(x,y);**
x: Niedrigster int-Wert, y: Höchster int-Wert.
Damit erhält man alle denkbaren Mischfarben.