Info zwischendurch:

Deklaration
Mit der Deklaration benennen wir eine Variable und machen diese dem Compiler bekannt. D.h. der Compiler weiß nun, ob wir uns im Laufe unseres Programms beim Eintippen des Variablennames vertippt haben.

Definition
Durch die Definition wird einer Variable ein Speicherbereich zugeteilt. Der Speicherbereich hat eine eindeutige Adresse und dient dazu, Werte abspeichern zu können. Mit dem Variablennamen sprechen wir im Prinzip nur einen Speicherbereich an. Unsere bisherige Art und Weise Variablen zu "erstellen", ist eine Deklaration **und** Definition:
int zahl; Durch „int" wird ein Speicherbereich (hier 2 Byte) zugeordnet und „zahl" deklariert den Namen der Variable.

Initialisierung
Haben wir eine Variable deklariert und definiert, so hat sie einen beliebigen Wert, je nach dem was gerade im zugewiesenen Speicherbereich steht. Da wir mit solch einem Zufallswert nicht arbeiten wollen, können wir mittels Initialisierung die Variable auf einen Anfangswert setzen. Variablen sollten möglichst immer initialisiert werden, um zu vermeiden, dass mit einem Zufallswert gearbeitet wird.
Beispiel: int a=5, b=-2, c=a+b;

1. Der serielle Monitor (SM)

Öffne den originalen Sketch „Blink" (Beispiele…) und initialisiere im Setup den seriellen Monitor mit **Serial.begin(9600);** Notiere jeweils hinter die digitalWrite-Anweisungen **Serial.print(„LOW-");** bzw. **Serial.print(„High-");** Ändere danach den Sketch mit **Serial.println** so ab, dass der Inhalt nicht neben-einander, sondern untereinander geschrieben wird.

2. Overflow sichtbar machen

Den Overflow aus Aufgabe 4II), 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. Eventuelle Erweiterung: Führe ein ähnliches Experiment mit einer byte-Variablen durch.

3. 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.

4. 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 aufeinander-folgenden Tönen erfolgt und die LED nach einer Zeit-verzögerung von ca. 4 Sekunden ausgeht.

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

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

5. 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.