Lösung der Aufgabe 6e) aus Blatt 4:
Hier ging es um den Sketch eines Lauflichts, erzeugt durch ein geeignet angesteuertes Shift-Register **ohne** die Anweisungen **bitSet** und **shiftOut**.
Das Licht einer LED soll in der 8-Bit-Anzeige hin- und herlaufen. Das Verständnis ist im Zusammenhang mit dem Extrablatt 74HC595_8 Bit Schieberegister mit Latch zu sehen.

Sketchname: **Ping-Pong-LED-1**

```
int shiftPin = 8;
int storePin = 9;
int dataPin = 10;
int counter = 0;

int laufLED = 0;
int zaehlRichtung = 0;
int muster[8] ={0,0,0,0,0,0,0,0} ;

void setup() {
  pinMode(storePin, OUTPUT);
  pinMode(shiftPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
}

void loop()
{
  delay(200);
  muster[laufLED] = 1;
  digitalWrite(storePin, LOW);

  for (int i = 0; i < 8; i++)
  {
    digitalWrite(shiftPin, LOW);
    digitalWrite(dataPin, muster[i]);
    digitalWrite(shiftPin, HIGH); }

  digitalWrite(storePin, HIGH);
  muster[laufLED] = 0;
  if (zaehlRichtung == 0) {
    laufLED++;

    if (laufLED == 7) {
      zaehlRichtung = 1;
    }
  }
  else {
    laufLED--;
    if (laufLED == 0) {
      zaehlRichtung = 0;

    }
  }
}
```

Grundsätzlich gilt: Wenn man nach jedem Schiebetakt ins Latch überträgt, bleibt die Schieberichtung dennoch immer gleich. Damit kann die Reflexion des Bit-Lichtes nie gelingen.
Dominik hat eine Lösung gefunden, die hier noch etwas überarbeitet wurde um die Lesbarkeit zu verbessern.

Die wesentliche Idee ist, immer nur komplette Bitmuster anzuzeigen, die als Byte erzeugt werden müssen. Dieses Byte muss nach **jedem** Transfer ins Register und der anschließenden Anzeige verändert werden.
laufLED: Ist der **Index** *) des Arrays, welcher hier den Wert auf „1" setzt, wenn die entsprechende LED einschaltet werden soll. Für {00000**1**00} leuchtet z.B. die 3. LED.
Counter: Wir benötigen einen Zähler (0 bis 7) um die Bits als Byte ins Register zu schieben.
zaehlRichtung: Wie der Name schon sagt, bestimmt der Wert dieser Variablen die Richtung, in die das Bit läuft. Geprüft wird auf „0" (zaehlRichtung==0). Ist der Wert „0", so läuft das Licht von links nach rechts, ist er ungleich „0" (z.B. 1), dann läuft das Licht von rechts nach links.
muster[8]: Ist das Array, dessen Werte (ein Byte) ins Register geschoben werden.
Mit **muster[laufLED]=1** setzen wir das erste Element (ganz rechts) auf „1", da zunächst noch **laufLED=0** gilt.
Nun wird das Byte 00000001 ins Register geschoben und angezeigt. Danach wird dieses Bit mit **muster[laufLED]=0;** wieder zurück gesetzt.
Da die Zählrichtung zunächst noch 0 ist, wird der Wert des **laufLED** um eins erhöht, so dass das zweite Bit im Muster auf 1 gesetzt wird.
Dann wird das entsprechende Byte durch die **for**-Schleife und **digitalWrite** ins Register geladen und angezeigt.
Wenn das **laufLED** auf 7 steht wird die Zählrichtung umgekehrt.

Die erste **if**-Abfrage trifft nun nicht mehr zu, es gilt die Anweisung in **else**. Das zunächst auf 7 stehende **laufLED** wird nun runter gezählt. Dadurch läuft das leuchtende Bit in die entgegengesetzte Richtung.
Ist **laufLED** bei 0 angekommen, muss die Zählrichtung wieder umgekehrt werden.

*) Erklärung: Der Index eines Arrays gibt an, an welcher Stelle der Wert quasi entnommen werden soll.
Beispiel: muster[]={5,3,8,1,0,2,7,4,}. Ist der Index „3", so gilt muster[3]=2. Der Index beginnt grundsätzlich bei 0, hier ist muster[0]=4.