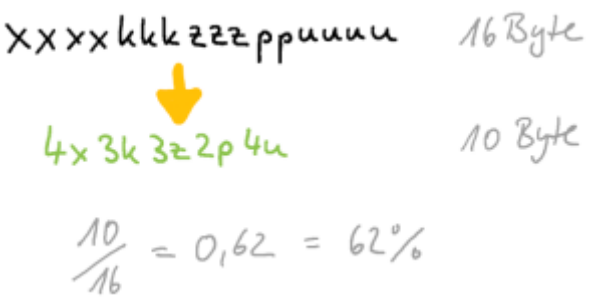
Lauflängencodierung

Eine einfache Methode der verlustfreien Komprimierung von Nutzdaten ist die **Lauflängencodierung** (RLE[1)]):

- Voraussetzung: in der zu komprimierenden Zeichenfolge gibt es Zeichen, die sich wiederholen.
- Idee: man gibt die Anzahl der Wiederholungen an und dann das Zeichen, das sich wiederholt.

Ein einfaches Beispiel:



Ein besseres Verfahren

Das kann man jetzt natürlich durchaus etwas ausgefeilter umsetzen. Wir betrachten dazu ein S/W PBM Bild.

```
P1
20 11

00000000000000000000
00000001110000000000
00000011100000000000
00000011100001111111
11111111100011111111
11111110011111111111
11100000000000000111
00000000000000011100
00000000000011100000
00000000001110000000
00000000000000000011
```

Nun kann man natürlich (wie oben) schreiben: 26w3s15w3s15w3s4w14s3w15s2w14s13w3s13w3s13w3s22w2s, das sind 51 ASCII-Zeichen, also 51 Byte. Die Bildinformationen umfassen 220Bit - kein wirklicher Gewinn.

Nun vereinbaren wir folgenden Code:

- Wir bilden Code-Worte aus 4 Bit
- Das erste Bit legt die Farbe fest (0 für weiß, 1 für schwarz)

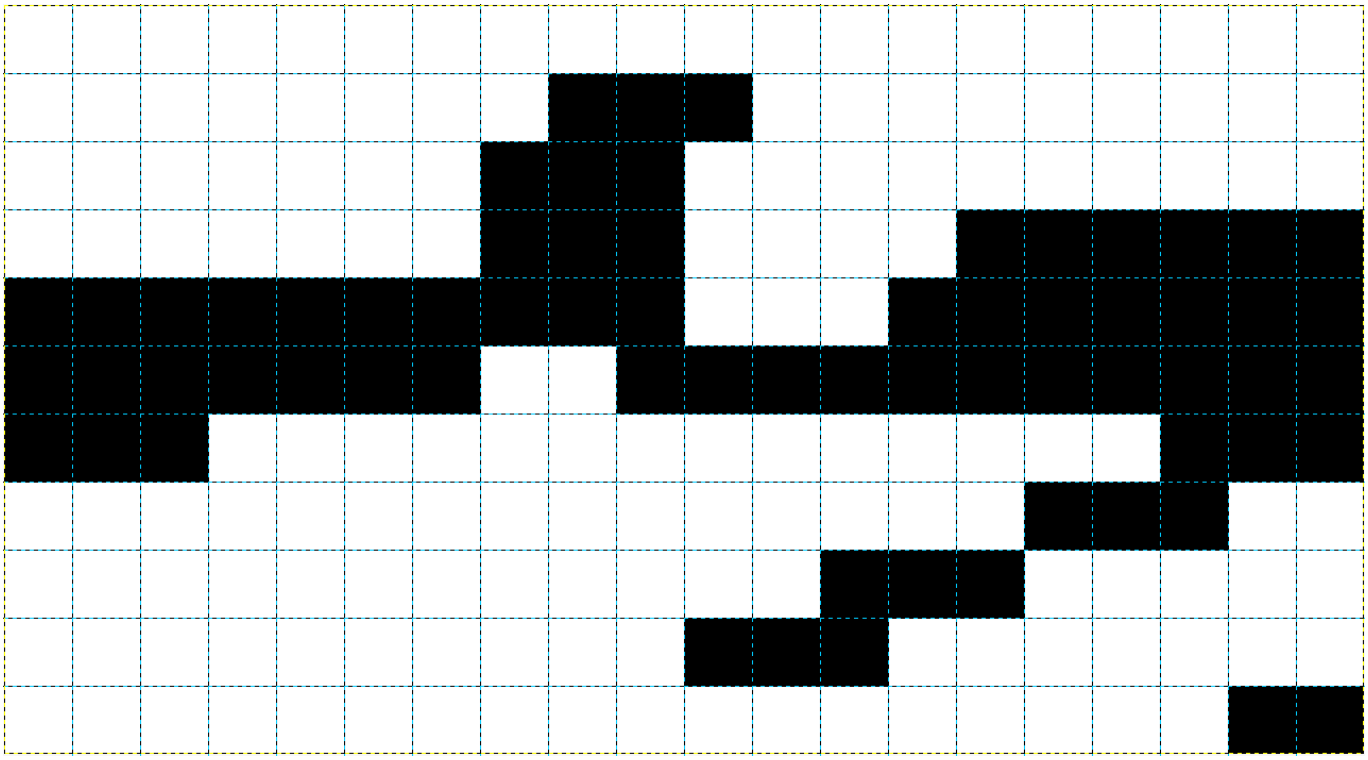
- Die folgenden 3 Bit geben an, wie oft diese Farbe wiederholt wird (000 für 1 Mal, 001 für 2 Mal, 010 für 3 Mal usw.)

Das ergibt die folgende Code Tabelle:

Code	Bedeutung
0000	1 weißes Pixel
0001	2 weißePixel
0010	3 weiße Pixel
0011	4 weiße Pixel
0100	5 weiße Pixel
0101	6 weiße Pixel
0110	7 weiße Pixel
0111	8 weiße Pixel
1000	1 schwarzes Pixel
1001	2 schwarze Pixel
1010	3 schwarze Pixel
1011	4 schwarze Pixel
1100	5 schwarze Pixel
1101	6 schwarze Pixel
1110	7 schwarze Pixel
1111	8 schwarze Pixel

(A1)

Codiere das Testbild mit dieser Code-Tabelle. Wie viele Bit benötigst Du dafür?



Lösung

```
0111 0111 0111 0011     // 28 we
1010                    // 3  sw
0111 0111               // 16 we
1010                    // 3  sw
0111 0111 0000          // 17 we
1010                    // 3  sw
0011                    // 4  we
1111 1111               // 16 sw
0010                    // 3  we
1111 1101               // 14 sw
0001                    // 2we
1111 1101               // 14 sw
0111 0101               // 14 we
1010                    // 3 sw
0111 0100               // 15 we
1010                    // 3 sw
0111 0101               // 14 we
1010                    // 3 sw
0111 0110               // 15 we
1010                    // 3 sw
0111 0111 0111 0000     // 25 we
1001                    // 2 sw
```

- Komprimiert: 38 * 4 Bit = 152 Bit
- Original: 20*11 = 220Bit
- Verhältnis: 152/220 = 0,69 = 69%

(A2)

Dekodiere folgende Daten für ein Bild mit 8 Pixel Breite und 11 Pixel Höhe.

```
0111-0111-0001-1011-0110-1000-0110-1000-0101-1001-0100-1000-0101-1000-0110-1011-0111-0111-0001
```

Erstelle eine bmp-Datei. Ermittle die Kompressionsrate in Prozent.

(A3)

Beschreibe ein Schwarz-Weiß-Bild, das gut mit Lauflängencodierung komprimiert werden kann und eines, das schlecht mit Lauflängencodierung komprimiert werden kann.

(A4)

Ein anderes System für eine Lauflängencodierung könnte folgendermaßen funktionieren.

- Die Code-Blöcke bestehen aus 3 Bit
- Der erste Code-Block einer Datei bezeichnet stets weiße Zeichen.
- Es kommt die folgende Codetabelle zur Anwendung:

Code	Bedeutung
000	Farbwechsel
001	1 Zeichen der aktuellen Farbe
010	2 Zeichen der aktuellen Farbe
011	3 Zeichen der aktuellen Farbe
100	4 Zeichen der aktuellen Farbe
101	5 Zeichen der aktuellen Farbe
110	6 Zeichen der aktuellen Farbe
111	7 Zeichen der aktuellen Farbe

Codiere das Testbild mit diesem Verfahren. Beurteile das Verfahren.

Material

01_kompression_rle.odp	273.8 KiB	30.11.2023 07:09
01_kompression_rle.pdf	317.7 KiB	30.11.2023 07:09
rle.png	61.4 KiB	03.10.2022 17:28
testbild.png	7.7 KiB	03.10.2022 18:08

1)

Run Length Encoding