Binäre Zahlen

dezbin.py

Üblicherweise schreiben wir Zahlen im Zehner- oder Dezimalsystem mit der Basis 10. Das bedeutet, dass alle Zahlen mit Ziffern aus {0, 1, 2, …, 9} geschrieben werden und je nach ihrer Position in der Zahl das Vielfache einer Potenz von 10 ausdrücken:

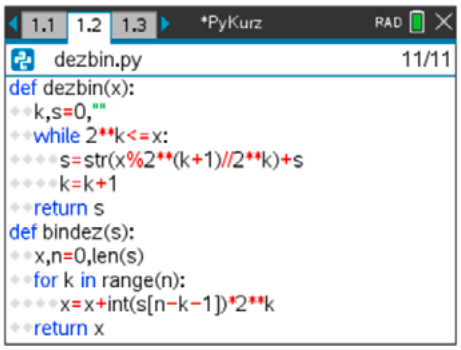
$$1273 = 1 \cdot 10^3 + 2 \cdot 10^2 + 7 \cdot 10^1 + 3 \cdot 10^0$$

Computer verwenden intern ein anderes Zahlensystem, nämlich binäre Zahlen oder auch Dualzahlen genannt. Das ist ein Zahlensystem mit der Basis 2, wobei jede Zahl nur aus den Ziffern aus {0, 1} besteht, die wiederum je nach ihrer Position eine Potenz von 2 darstellen:

$$101011 = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$$

Das ist die Zahl 43 dargestellt als binäre Zahl.

Das folgende Programm erzeugt zwei Funktionen, die Zahlen von einem System ins andere übertragen.



Beachte, dass die Dualzahlen als Zeichenketten (strings) eingegeben werden müssen – und auch als solche ausgegeben werden.

Bemerkung: In der 4. Programmzeile treten zwei Operationszeichen auf, die einer Erklärung bedürfen:

- Das %-Zeichen entspricht der mod-Operation und ergibt den Rest der Division der links stehenden Zahl durch die rechts stehende: `25%7` = 4.
- Das doppelte Divisionszeichen // ergibt den ganzen Teil der Division: `25//7` = 3.