1. In this exercise we want to develop a *new* standard for representing Swiss postcodes in binary format. Swiss postcodes have four digits and ranges from 1000-9999.

(a) (2 points) How many bits in total are needed if we want to represent the Swiss postcodes using **two's complement** binary format? Encode the postcode of Wilchingen, SH (8217)[1] using this format.

Solution: 15 bits are needed: there are 9'000 different possible postcodes (more or less), and you need at least 14 bits to represent them (2^{14}==16'536). You need another bit for the two's complement.
$(8217)_{10} = 8192 + 16 + 8 + 1 = (010\,0000\,0001\,1001)_2$

(b) (2 points) How many bits in total are needed if we represent each decimal digit as a separate unsigned binary number? Encode the postcode of Wilchingen, SH (8217) using this format.

Solution: 16 bits are needed: For each decimal digit 4 bits will be needed. There are 4 digits, that makes 16 in total
$(8, 2, 1, 7)_{10} = (1000, 0010, 0001, 0111)_2$

(c) (2 points) How many bits in total are needed if we represent two decimal digits at a time as a separate unsigned binary number? Encode the postcode of Wilchingen, SH (8217) using this format.

Solution: 14 bits are needed: For two decimal digits (00-99), 7 bits will be needed (2^7==128). There are two such numbers digits, that makes 14 in total
$(82, 17)_{10} = (101\,0010, 001\,0001)_2$

[1] We needed a *not so difficult* postcode for this exercise, otherwise there is no special reason for Wilchingen.