Rotation sum

The integer $(19)_{10}$ (decimal representation) can be written as $(10011)_2$ in the binary numeral system. This allows us to define the *clockwise rotation* of an integer as the operation that puts the last digit at the front in the binary representation of the integer. If we thus perform a clockwise rotation to the binary number $(10011)_2$, we obtain the binary number $(11001)_2 = (25)_{10}$.

If a leading zero appears in a binary number after it has been rotated clockwise, it may be omitted. This way, a series of integers can be constructed by repeatedly applying clockwise rotations on the previous integer. This *clockwise rotation series* ends if the binary representation of the last integer contains no more zeros (because leading zeros are omitted). The clockwise rotation series of the integer 19 is then generated in the following way

19 10011	$\stackrel{\text{cw}}{\longrightarrow}$	25 11001	$\stackrel{\text{cw}}{\longrightarrow}$	28 11100	$\stackrel{\text{cw}}{\longrightarrow}$	14 $0\!\!\!\!\!-\!\!1110$	$\stackrel{\text{cw}}{\longrightarrow}$	7 $0\!\!\!\!\!-\!\!111$

The sum of the integers in the clockwise rotation series is referred to as the *clockwise rotation sum*. The clockwise rotation sum of the integer 19 therefore is computed as $19 + 25 + 28 + 14 + 7 = 93$.

The reverse operation, which puts the first digit in the binary representation of an integer at the back, is then logically referred to as the *counterclockwise rotation* of the integer. As such, the counterclockwise rotation of $(10011)_2 = (19)_{10}$ is $(0\!\!\!\!\!\!-\!\!0\!\!\!\!\!\!-\!\!111)_2 = (7)_{10}$. Note that in this case two leading zeros are omitted. In analogy to the clockwise rotation operations, the *counterclockwise rotation series* and the *counterclockwise rotation sum* can be defined based on the counterclockwise rotation. The counterclockwise rotation sum of 357 is 789, as can be derived from the corresponding counterclockwise rotation series

357 101100101	$\stackrel{\text{ccw}}{\longrightarrow}$	203 $0\!\!\!\!\!-\!\!11001011$	$\stackrel{\text{ccw}}{\longrightarrow}$	151 10010111	$\stackrel{\text{ccw}}{\longrightarrow}$	47 $0\!\!\!\!\!-\!\!0\!\!\!\!\!-\!\!101111$	$\stackrel{\text{ccw}}{\longrightarrow}$	31 $0\!\!\!\!\!-\!\!11111

Assignment

Write three functions `rotation`, `rotationSeries` and `rotationSum`, that can be used to respectively compute the rotation, the rotation series and the rotation sum for a given integer $n \in \mathbb{N}_0$. Each of these functions has a parameter `number` to which the integer n must be passed. In addition, each of the functions also has a parameter `clockwise` that takes a Boolean value. This Boolean indicates whether clockwise (value `True`, the default value) or counterclockwise (value `False`) rotations must be computed. Try to make optimal reuse of your own source code in implementing these three functions.

Tip: examine how the built-in Python functions `bin` and `int` can be used to convert the decimal representation of an integer into its binary representation, and vice versa.

Example

```
>>> rotation(19)
25
>>> rotation(25, clockwise=True)
28
>>> rotation(25, clockwise=False)
19
>>> rotation(19, clockwise=False)
7

>>> rotationSeries(19)
[19, 25, 28, 14, 7]
>>> rotationSeries(69)
[69, 98, 49, 56, 28, 14, 7]
>>> rotationSeries(205, clockwise=True)
[205, 230, 115, 121, 124, 62, 31]
>>> rotationSeries(357, clockwise=False)
[357, 203, 151, 47, 31]
>>> rotationSeries(54321, clockwise=False)
[54321, 43107, 20679, 8591, 799, 575, 127]

>>> rotationSum(19)
93
>>> rotationSum(69)
321
>>> rotationSum(205, clockwise=True)
888
>>> rotationSum(357, clockwise=False)
789
>>> rotationSum(54321, clockwise=False)
128199
```

Het natuurlijk getal $(19)_{10}$ (decimale voorstelling) kan geschreven worden als $(10011)_2$ in het binair talstelsel. Dit laat ons toe om de *rotatie naar rechts* van een natuurlijk getal te definiëren als de bewerking waarbij het laatste cijfer vooraan geplaatst wordt in de binaire voorstelling van het getal. Als we op die manier het binair getal $(10011)_2$ naar rechts roteren, dan bekomen we het binair getal $(11001)_2 = (25)_{10}$.

Als na een rotatie naar rechts een nul vooraan een binair getal komt te staan, dan mag die weggelaten worden. Op die manier kan een reeks getallen opgebouwd worden, waarbij het volgende getal steeds bekomen wordt door een rotatie naar rechts uit te voeren. Deze *rechtse rotatiereeks* eindigt als de binaire voorstelling van het laatste getal geen nullen meer bevat (omdat voorloopnullen weggelaten worden). De rechtse rotatiereeks van het getal 19 wordt dan bijvoorbeeld

19 10011	$\stackrel{\text{r}}{\longrightarrow}$	25 11001	$\stackrel{\text{r}}{\longrightarrow}$	28 11100	$\stackrel{\text{r}}{\longrightarrow}$	14 $0\!\!\!\!\!-\!\!1110$	$\stackrel{\text{r}}{\longrightarrow}$	7 $0\!\!\!\!\!-\!\!111$

De som van de getallen in de rechtse rotatiereeks wordt de *rechtse rotatiesom* genoemd. De rechtse rotatiesom van het getal 19 is bijgevolg $19 + 25 + 28 + 14 + 7 = 93$.

De omgekeerde bewerking, waarbij het eerste cijfer in de binaire voorstelling van een natuurlijk getal achteraan geplaatst wordt, wordt dan logischerwijs de *rotatie naar links* van het getal genoemd. De rotatie naar links van $(10011)_2 = (19)_{10}$ is dan $(0\!\!\!\!\!\!-\!\!0\!\!\!\!\!\!-\!\!111)_2 = (7)_{10}$, waarbij in dit voorbeeld twee voorloopnullen weggelaten worden. Analoog als bij de rotatie naar rechts, kunnen op basis van de rotatie naar links ook de *linkse rotatiereeks* en de *linkse rotatiesom* gedefinieerd worden. De linkse rotatiesom van 357 is dan 789, zoals kan afgeleid worden uit de bijhorende linkse rotatiereeks

357		203		151		47		31
101100101	$\stackrel{\text{l}}{\longrightarrow}$	$0\!\!\!\!\!-\!\!11001011$	$\stackrel{\text{l}}{\longrightarrow}$	10010111	$\stackrel{\text{l}}{\longrightarrow}$	$0\!\!\!\!\!-\!\!0\!\!\!\!\!-\!\!101111$	$\stackrel{\text{l}}{\longrightarrow}$	$0\!\!\!\!\!-\!\!11111$

Opgave

Schrijf drie functies `rotatie`, `rotatiereeks` en `rotatiesom`, waarmee respectievelijk de rotatie, de rotatiereeks en de rotatiesom van een gegeven natuurlijk getal $n \in \mathbb{N}_0$ kunnen berekend worden. Elk van deze functies heeft een parameter `getal` waaraan het getal n moet doorgegeven worden. Daarnaast hebben alle functies nog een parameter `rechts` waaraan optioneel een Booleaanse waarde kan doorgegeven worden. Deze geeft aan of rotaties naar rechts (waarde `True`, de standaardwaarde) of naar links (waarde `False`) moeten uitgevoerd worden. Probeer je eigen programmacode zo optimaal mogelijk te hergebruiken bij het implementeren van deze drie functies.

Tip: ga na hoe de ingebouwde Python functies `bin` en `int` kunnen gebruikt worden om de decimale voorstelling van een natuurlijk getal om te zetten naar zijn binaire voorstelling, en omgekeerd.

Voorbeeld

```
>>> rotatie(19)
25
>>> rotatie(25, rechts=True)
28
>>> rotatie(25, rechts=False)
19
>>> rotatie(19, rechts=False)
7

>>> rotatiereeks(19)
[19, 25, 28, 14, 7]
>>> rotatiereeks(69)
[69, 98, 49, 56, 28, 14, 7]
>>> rotatiereeks(205, rechts=True)
[205, 230, 115, 121, 124, 62, 31]
>>> rotatiereeks(357, rechts=False)
[357, 203, 151, 47, 31]
>>> rotatiereeks(54321, rechts=False)
[54321, 43107, 20679, 8591, 799, 575, 127]

>>> rotatiesom(19)
93
>>> rotatiesom(69)
321
>>> rotatiesom(205, rechts=True)
888
>>> rotatiesom(357, rechts=False)
789
>>> rotatiesom(54321, rechts=False)
128199
```