# All the king's wine

## General information

### Convert numbers to another base

We usually work with the decimale representation of numbers. Decimal representation means that the numbers are represented as a sequence of the digits 0 to 9, whose base-10 value depends on their position. However, we can also represent numbers using other bases. Probably the best known is base 2 (binary), where numbers are represented as a sequence of the digits 0 and 1. In computer science, other representations that are often used are base 8 (octal) and base 16 (hexadecimal).

If you have the string representation of a number in a certain base, you can easily convert it into its decimal value by using the built-in function `int`. The function `int` has a second optional parameter that indicates the base of the representation that is passed as the first argument. The default value of the second parameter is 10.

```
>>> representation = '101'
>>> int(representation)
101
>>> int(representation, 10)
101
>>> int(representation, 2) # base 2: binary
5

>>> representation = '45'
>>> int(representation, 2) # base 2: binary
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 2: '45'
>>> int(representation, 8)
37
>>> int(representation, 16)
69
>>> int('2A', 16)
42
```

# Number walks

## General information

### Remarks

### Trigonometric functions from the `math` module

The `math` module from the Python Standard Library defines a couple of **trigonometric functions** such as the sine function (`sin`), the cosine function (`cos`) and the tangent function (`tan`). It's important to pay attention to the fact that these functions expect an angle expressed in radians, and not in degreees. Luckily enough, the `math` module also defines functions to convert an angle expressed in degrees into radians (`radians`) and *vice versa* (`degrees`).

```
>>> import math
>>> angle = 90
>>> radians = math.radians(angle)
>>> radians
1.5707963267948966
>>> radians == math.pi / 2
True
>>> math.cos(radians) # must evaluate to 0, but note the rounding error
6.123233995736766e-17
>>> math.sin(radians)
1.0
```

## Specific information

If you start at a given point in the Euclidean plane and take a unit step in a direction described by the (clockwise) angle with the positive Y-acis, you can use the sine and cosine of that angle to determine your new position after having taken the step. After all, from our goniometrics classes we recall that the displacement in the X-direction is given by the sine of the angle and the displacement in the Y-direction by the cosine of the angle. This is illustrated in the figure below, that shows the unit circle.

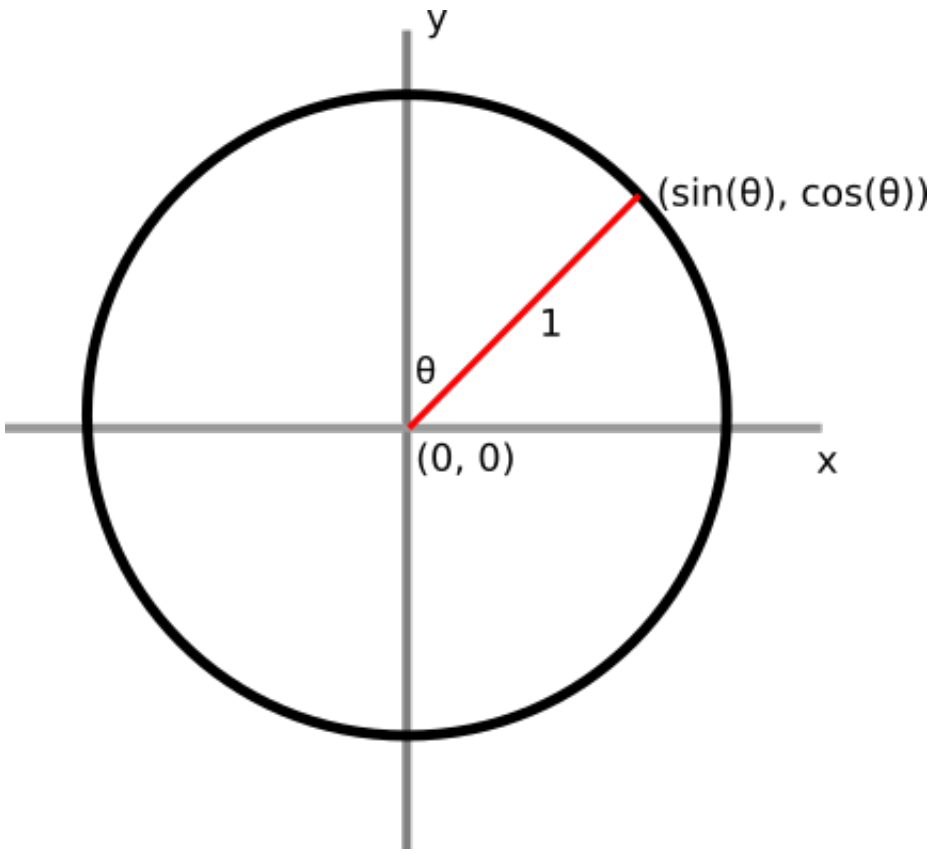


Figure 1: Goneometrische cirkel

## Wow! signal

### Specific information

A possible strategy to tackle this assignment, is to postpone the processing of a group of successive alphanumerical characters (letters or digits) until you observe a non-alphanumerical character while traversing to the characters on a line. After all, any non-alphanumerical character ends the sequence of alphanumerical characters in front of it. This is illustrated in the following example.

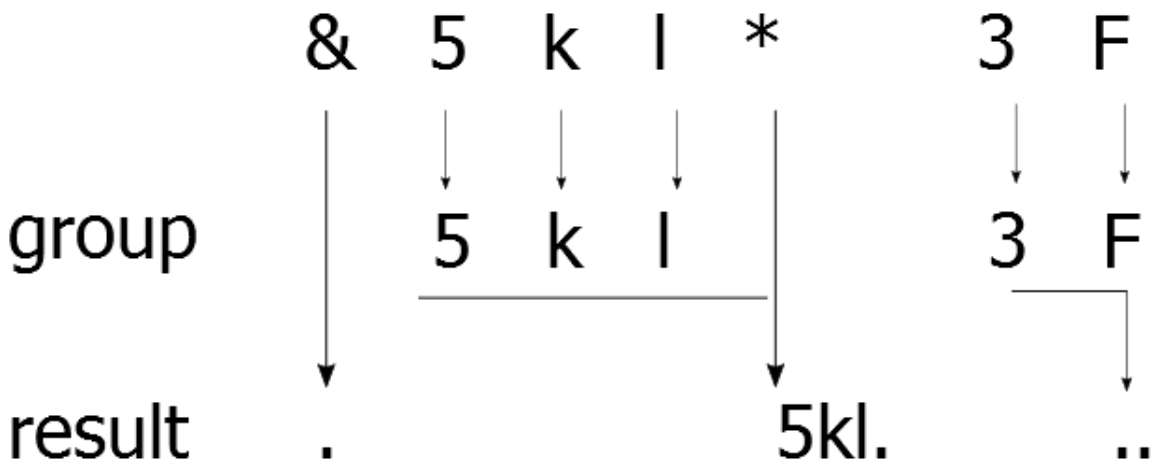


Figure 2: Conversion of Wow! signal

The strategy exists of traversing the characters on a line one by one. You can only decide whether or not you have found a Wow! signal if the group of successive alphanumerical characters has been read completely. This can be done by appending any letters or digits that you encounter to the end of a string variable. The group of characters in this string variable is not processed until the group is complete: this is the case if a non-alphanumerical character is encountered or if the end of the line is reached.

## The missing number

### Specific information

For each possible number of digits $d$ in the first number of the sequence of integers we apply the following procedure. We take the number $n$ that was last observed in the sequence. At first, the number $n$ is equal to the number formed by the first $c$ digits in the sequence of digits. We define *restseq* as the remaining part of the sequence of digits that still needs to be checked.

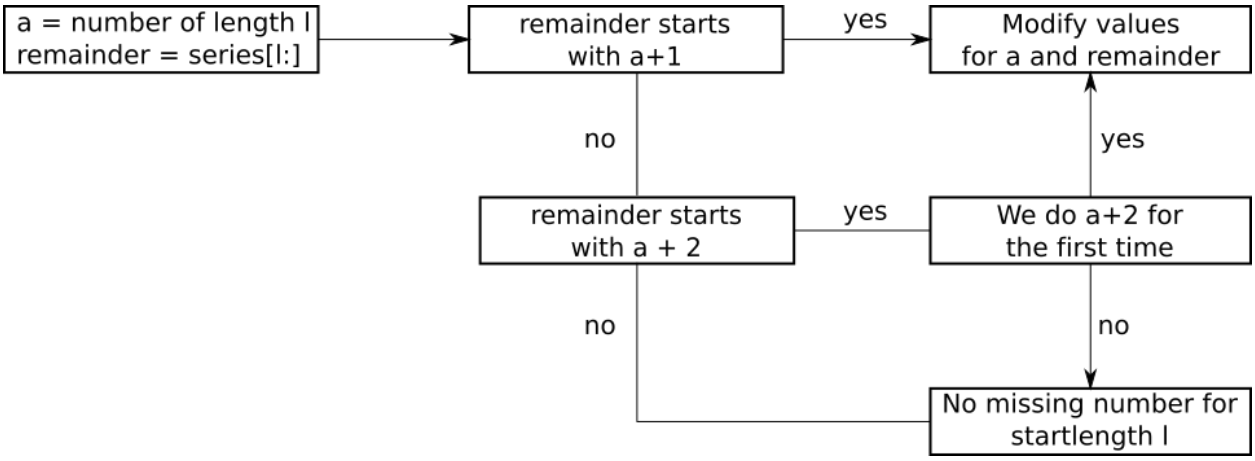


Figure 3: Schemr