Example 26. Python Let us return to the task of using Python to express, say, 0.1 in binary. Last time, we copied four lines of code 5 times to produce 5 digits. Instead, to repeat something a certain number of times, we should use a **for loop**. For instance:

```
>>> for i in range(3):
        print('Hello')

  Hello
  Hello
  Hello
```

Homework. Replace `print('Hello')` with `print(i)`.

Important comment. The indentation in the second line serves an important purpose in Python. All lines (after the first) that are indented by the same amount will be repeated. Test this by adding a non-indented line containing `print('Bye!')` at the end.

With this in mind, we can upgrade our previous code as follows:

```
>>> x = 0.1  # or any value < 1
    nr_digits = 5  # we want this many digits of x
    from math import trunc
    for i in range(nr_digits):
        x = 2*x
        digit = trunc(x)
        print(digit)
        x = x-digit

  0
  0
  0
  1
  1
```

Example 27. Python As our next upgrade, let us collect the digits in a list instead of individually printing them to the screen. Here is how we can create a list in Python and add an element to it:

```
>>> L = [1, 2, 3]

>>> L.append(4)

>>> print(L)

  [1, 2, 3, 4]
```

Here is our code adjusted for using a list (and now it is more pleasant to ask for more digits):

```
>>> x = 0.1  # or any value < 1
    nr_digits = 10  # we want this many digits of x
    digits = []  # this list will store the digits of x
    from math import trunc
    for i in range(nr_digits):
        x = 2*x
        digit = trunc(x)
        digits.append(digit)
        x = x-digit
    print(digits)

  [0, 0, 0, 1, 1, 0, 0, 1, 1, 0]
```

Example 28. $\boxed{\text{Python}}$ For our final upgrade, we collect the code into a function that we call `fracpart_digits`. This is crucial for making it possible to use the code on different numbers.

```
>>> def fracpart_digits(x, nr_digits):
        digits = []
        from math import trunc
        for i in range(nr_digits):
            x = 2*x
            digit = trunc(x)
            digits.append(digit)
            x = x-digit
        return digits
```

We are now able to compute the digits of numbers by simply calling our function:

```
>>> fracpart_digits(0.1, 10)

[0, 0, 0, 1, 1, 0, 0, 1, 1, 0]

>>> fracpart_digits(0.2, 10)

[0, 0, 1, 1, 0, 0, 1, 1, 0, 0]

>>> from math import pi

>>> fracpart_digits(pi/4, 10)

[1, 1, 0, 0, 1, 0, 0, 1, 0, 0]
```

Comment. Recall that, if you are not in a Python console, you need to add `print(..)` to see any output.

As an advanced use of lists, here is how we could compute 5 digits of $1/n$ for $n \in \{2, 3, 4, 5\}$:

```
>>> [fracpart_digits(1./n, 5) for n in range(2,6)]

[[1, 0, 0, 0, 0], [0, 1, 0, 1, 0], [0, 1, 0, 0, 0], [0, 0, 1, 1, 0]]
```

Comment. Note how the digits of $1/2 = (0.1)_2$ and $1/4 = (0.01)_2$ are particularly easy to verify.

Errors: absolute and relative

Suppose that the true value is x and that we approximate it with y.

- The **absolute error** is $|y - x|$.
- The **relative error** is $\left|\frac{y - x}{x}\right|$.

For many applications, the relative error is much more important. Note, for instance, that it does not change if we scale both x and y (in other words, it doesn't change if we change units from, say, meters to millimeters). Speaking of units, note that the relative error is dimensionless (it has no units even if x and y do).

Example 29. There are lots of interesting approximations of π. In each of the following cases, determine both the absolute and the relative error.

(a) $\pi \approx \frac{22}{7}$ ($22/7 \approx 3.14286$)

(b) $\pi \approx \sqrt[4]{9^2 + 19^2/22}$ (This approximation is featured in https://xkcd.com/217/.)

Solution.

(a) The absolute error is $\left|\frac{22}{7}-\pi\right| \approx 0.0013 = 1.3 \cdot 10^{-3}$.

The relative error is $\left|\frac{\frac{22}{7}-\pi}{\pi}\right| \approx 0.00040 = 4.0 \cdot 10^{-4}$.

Comment. Sometimes the relative error is quoted as a "percentage error". Here, this is 0.04%.

(b) The absolute error is $\left|\sqrt[4]{9^2+19^2/22}-\pi\right| \approx 1.0 \cdot 10^{-9}$.

The relative error is $\left|\frac{\sqrt[4]{9^2+19^2/22}}{\pi}\right| \approx 3.2 \cdot 10^{-10}$.

Example 30. (homework) π^{10} is rounded to the closest integer. Determine both the absolute and the relative error (to three significant digits).

Solution. $\pi^{10} \approx 93{,}648.0475$

The absolute error is $|93{,}648 - \pi^{10}| \approx 0.0475$.

The relative error is $\left|\frac{93{,}648 - \pi^{10}}{\pi^{10}}\right| \approx 5.07 \cdot 10^{-7}$.

Example 31. Strangely, $e^{\pi} - \pi = 19.999099979\ldots$. Determine both the absolute and the relative error when approximating this number by 20.

https://xkcd.com/217/

Solution. The absolute error is $|20 - (e^{\pi} - \pi)| \approx 9.0 \cdot 10^{-4}$.

The relative error is $\left|\frac{20-(e^{\pi}-\pi)}{e^{\pi}-\pi}\right| \approx 4.5 \cdot 10^{-5}$.

Example 32. One of the most famous/notorious mathematical results is **Fermat's last theorem**. It states that, for $n > 2$, the equation $x^n + y^n = z^n$ has no positive integer solutions!

Pierre de Fermat (1637) claimed in a margin of Diophantus' book *Arithmetica* that he had a proof ("I have discovered a truly marvellous proof of this, which this margin is too narrow to contain.").

It was finally proved by Andrew Wiles in 1995 (using a connection to modular forms and elliptic curves).

This problem is often reported as the one with the largest number of unsuccessful proofs.

On the other hand, in a Simpson's episode, Homer (in 3D!) encounters the formula

$$1782^{12} + 1841^{12} \text{ "="} \ 1922^{12}.$$

If you check this on an old calculator it might confirm the equation. However, the equation is not correct, though it is "nearly": $1782^{12} + 1841^{12} - 1922^{12} \approx -7.002 \cdot 10^{29}$.

Why would that count as "nearly"? Well, the smallest of the three numbers, $1782^{12} \approx 1.025 \cdot 10^{39}$, is bigger by a factor of more than 10^9. So the difference is extremely small in comparison.

More precisely, if $1782^{12} + 1841^{12}$ is the true value, then approximating it with 1922^{12} produces

- an absolute error of $|1782^{12} + 1841^{12} - 1922^{12}| \approx 7.00 \cdot 10^{29}$ (rather large), and
- a relative error of $\left|\frac{1782^{12}+1841^{12}-1922^{12}}{1782^{12}+1841^{12}}\right| \approx 2.76 \cdot 10^{-10}$ (very small).

Comment. We can immediately see that Homer's formula is not quite correct by looking at whether each term is even or odd. Do you see it?

http://www.bbc.com/news/magazine-24724635

Armin Straub
straub@southalabama.edu