Algorithms - a simple introduction in Python: Part Ten

In almost every computation a great variety of arrangements for the succession of the processes is possible, and various considerations must influence the selections amongst them for the purposes of a calculating engine. One essential object is to choose that arrangement which shall tend to reduce to a minimum the time necessary for completing the calculation.

Ada Lovelace

τ or π?

You may have heard that some mathematicians want to get rid of pi! The proponents of "tau" (which equals 2π) say that this is the more sensible value to use. Well, whether you want to calculate pi or tau, I am going to present one method for doing so.

How do we calculate pi? Imagine a circle with radius 1. The circumference will be 2π. The arctan function gives us the angle when we pass it the slope (or gradient) of a line. A 45° angle has a slope of 1 (for every unit we move right, we move up one unit). 45° is one eighth of a circle.

Now, for this equation, we need to measure the angle in radians instead of degrees. The complete circle (360°) corresponds to 2π radians[1]. So our quarter-circle, with its angle equal to arctan of 1, in radians will give us π/4.

The Scottish mathematician, James Gregory, discovered this method for working out arctan:

$$arctan(t) = t - \frac{t^3}{3} + \frac{t^5}{5} - \frac{t^7}{7} + \frac{t^9}{9} - ...$$

This series converges towards π/4 (or τ/2) for values of t of 1 or less. But it converges fairly slowly when t = 1. Euler came up with another equation that is helpful here:

$$\frac{\pi}{4} = \arctan\left(\frac{1}{2}\right) + \arctan\left(\frac{1}{3}\right)$$

It turns out that we can expand this sequence, so that we are dealing with even smaller values for t:

$$\frac{\pi}{4} = \arctan\left(\frac{1}{2}\right) + \arctan\left(\frac{1}{5}\right) + \arctan\left(\frac{1}{8}\right)$$

Now, you might recognise the denominators of those first two arctan expressions as the odd-numbered terms in the Fibonacci series: (0, **1**,) 1, **2**, 3, **5**, 8, 13 ... We can continue expanding like this to get an infinite series.

$$\frac{\pi}{4} = \arctan\left(\frac{1}{F_3}\right) + \arctan\left(\frac{1}{F_5}\right) + \arctan\left(\frac{1}{F_6}\right)$$

Each term even-numbered member of the Fibonacci series (eg F(6) = 8) can be expanded into the next pair so:

$$\arctan(1/F_6) \rightarrow \arctan(1/F_7) + \arctan(1/F_8) ...$$

This is the method we are going to use.

---

[1] Or just tau radians, if you prefer!

Now, if we just use the standard floating-point numbers in Python, our result will not be accurate once we get past about 16 decimal places. If we want to calculate tau or pi to more decimal places, we need to use arbitrary precision floating-point numbers. In this program, we are using the “Decimal” data-type. With this, we can set the precision to whatever value we like. Of course, the more accurate the value of tau or pi we want, the longer it is going to take the program to complete. We only need to use Decimal once, when we first call the euler() function, and Python will automatically calculate the results of mathematical operations on this value as “Decimal” objects.

In this version, the user has the ability to decide on how many decimal places are required. Here’s the code. You’ll notice I add two to the number the user gives. This is so that the precision value allows for the initial “3“ and gives a result with 11 places. When we print the slice of the result [:-1], we are omitting the final place, as this will most-likely not be accurate.

```
# tau.py
# arctan(1)*8 = tau = 2pi
from decimal import *

def main():
        print("*** Pi and Tau ***")
        print("Type the number of decimal places of Tau you require.")
        ans = input("For Pi, type a comma and the number of decimal places.\n? ")
        try:
                prec = int(ans)
                c = "Tau"
                k = 8
        except ValueError:
                try:
                        c, prec = ans.split(',')
                        c = "Pi"
                        k = 4
                        prec = int(prec)
                except ValueError:
                        print("Oops.")
                        return 1
        getcontext().prec = prec + 2
        print("Calculating", c, ". . .")
        print(str(euler(1, Decimal(2)) * k)[:-1])

def arctan(n):
    """Uses Gregory's formula for calculating atan."""
    copy_of_n = n
    atan = None
    i = 3
    while atan != n:
            atan = n
            n = n - copy_of_n ** i / i + copy_of_n ** (i + 2) / (i + 2)
            i += 4
    return n

def euler(a, b):
    """Uses Euler's formula and Fibonacci numbers."""
    euler = 0
    check = None
    while check != euler:
            check = euler
            euler += arctan(1/b)
            a = b + a
            b = b + a
    return euler

# main
if __name__ == "__main__":
    main()
```

Sample Output:

```
*** Pi and Tau ***
Type the number of decimal places of Tau you require.
For Pi, type a comma and the number of decimal places.
? 20
Calculating Tau ...
6.28318530717958647693
>>>
```