Additional Python challenges

Introduction

If you are already quite experienced in programming, then here are some more mathematically problems to be solved using Python. These problems are **completely optional** and won't be assessed – they are intended as additional examples to practise the algorithmic approach outlined in the programming lectures.

For each problem, I will provide my attempt at a solution that uses (as much as possible) only those aspects of Python already covered in the lab sheets. I encourage you to try writing your own program before looking at my attempt.

Once you have solved a problem, think about how you might **refactor** your program to make it run faster! Each problem is designed to be solvable in less than a minute of CPU time on a standard computer.

These problems are taken from Project Euler (`https://projecteuler.net`), a website containing a list of mathematical problems intended to be solved with computer programs. The problem number refers to the Project Euler list; so for example, Problem 5 can be found at `https://projecteuler.net/problem=5`.

It is easy to find programs that solve these problems online, but that is not the point – I hope that you will try the problems yourself! If you have any questions or would like to show me your attempt at a problem, please use the online discussion board.

Problem 1: Multiples of 3 and 5

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23. What is the sum of all the multiples of 3 or 5 below 1000?

Here is my first attempt at a program to solve this problem:

```
total = 0
i = 0
while i < 1000:
    if i % 3 == 0 or i % 5 == 0:
        total = total + i
    i = i + 1
print("The sum of all multiples of 3 or 5 below 1000 is", total)
```

This attempt runs fairly quickly and gives the correct answer, which is 233168. However, my attempt runs very slowly if we want to calculate the sum of all multiples of 3 or 5 below a much larger number, say 10^9. How long does your program take to run? Try refactoring your program to speed it up – it is possible to solve this larger problem in less than a second of CPU time!

Problem 5: Smallest multiple

2520 is the smallest number that can be divided by each of the numbers from 1 to 10 without any remainder. What is the smallest positive number that is divisible by all of the numbers from 1 to 20?

Here is my first attempt at a program to solve this problem:

```
n = 1
searching = True
while searching:
```

```
    i = 1
    while i <= 20:
        if n % i == 0:
            i = i + 1
        else:
            break
    if i == 21:
        searching = False
    else:
        n = n + 1
print("The smallest positive number divisible by 1 to 20 is", n)
```

This attempt gives the correct answer, which is 232,792,560. However, it takes an extremely long time to run! How long does your program take to run? If it is taking longer than a few minutes, then try replacing the upper limit 20 with a smaller number, say 12. Try refactoring your program to speed it up – it is possible to solve this problem in less than a second of CPU time!

Timing a program

There are several ways that you can time how long a program takes to run. One way is to import and use the `time` module, as shown below.

```
import time
start = time.time()

# Insert your program here

end = time.time()
print("This program took", end - start, "seconds to run")
```

If you don't know what Python modules are, then it may better to revisit these problems after the Week 8 Python lab.