Algorithms and Data Structures: Week 2

Question 1

Consider the following code.

```
Input: integer $L$, integer $H$
Output: ?
  integer $x = L$
  integer $p = 1$
  while $x \leq H$ do
    if $x$ is odd then
      $p = p \times x$
    end if
    $x = x + 1$
  end while
  print "result is ", $p$
```

(a) What does this algorithm do, that is, what is being computed in the variable p?

(b) Rewrite this code to use a **for** loop instead of a **while** loop.

Question 2

Design an algorithm in pseudocode that, when given some number n and numbers $a_0, a_2, \ldots, a_{n-1}$ as input, computes and outputs the second smallest of those numbers. In other words, the input/output specification looks something like

Input: integer n, integers $a_0, a_1, \ldots, a_{n-1}$
Output: second smallest of $a_0, a_1, \ldots a_{n-1}$

Finding the smallest is a problem given as an exercise for you to do during the lectures. So look back at (or now construct) your solution and think about how to modify it?

(Note you should **not** include a line in your solution that says something like

sort $a_0, a_1, \ldots, a_{n-1}$ so they are in order from smallest to largest

This would not be efficient, as sorting the integers is a more difficult problem than the one you are asked to solve.)

Question 3

Recall that a natural number k is called prime if it can be divided without remainder only by one and itself. Write a simple algorithm in pseudocode that tests whether a given number, i.e., a number that is given as input, is prime. You may use a condition such as "a divides b" and you can also use a command **exit** that will cause the algorithm to terminate (i.e. once you have found that k is not a prime you can stop).

Question 4

Write a simple algorithm in pseudocode that takes as input the date (given as three integers for year, month and day) and a further integer k. The output should be the date k days after the input. You can, if you wish, make the simplying assumption that there are no leap years and, moreover, that every month has the same number of days (so a year contains 360 days divided into 12 30-day months, say). Also assume that for two integers a and b, a/b gives an integer and $a \,\%\, b$ gives the remainder. (For example $800/360 = 2$ and $800 \,\%\, 360 = 80$ so 800 days from now is 2 years and 80 days away).

If you finish early, it is good practice to try to implement some of your solutions. Take Question 4 for example, the ideas needed are essentially just simple mathematics, but possibly your first attempt at writing an algorithm in pseudocode has some intricacy (and everyone's attempt will be slightly different) so how can we check it? The easiest way is to convert the pseudocode to code and test it. So if you can, have a go.

Of course, I know that you might not have studied programming before, and learning to program is not the focus of this module. But testing your ideas with snippets of code is a very useful supplementary activity and as you progress in the Computational Thinking module you will be able to apply the skills you gain in this module. I will often provide examples using python. Also when I write pseudocode, I will often use the conventions of python, easing the process of converting the pseudocode to code. There is a document on duo that explains in detail how to get started and uses the problem of finding the smallest number in a set as an example. Have a look at this and then see if you can write the code for question 2 or 4.

If you have any time left, have another look at Project Euler – Problem 19, for example.