Practical Sheet 3

More Advanced Python

1 Multiple Dimensional Arrays

Exercise 1.1: Sum Columns of Table

Write a function to sum the columns of a 2 dimensional array (i.e. a list of lists). The function should work for any size table (providing it is regular in shape). *Note: this exercise assumes you are familiar with functions. We revise functions in the next session.*

For example, with:

```
table1 = [         \
    [1, 2, 3, 4], \
    [4, 5, 6, 7], \
    [8, 9,10,11]  ]
```

we get:

```
>>> sumCol(table1,0)
13
>>> sumCol(table1,1)
16
```

Exercise 1.2: Swapping Rows and Columns

Write a function to swap row and columns, for any size or shape of table. This function is usually called 'transpose'. For example, with table:

```
table2 = [ \
    [1,2,3,4], \
    [5,6,7,8]  ]
```

we get:

```
>>> transpose(table2)
[[1, 5], [2, 6], [3, 7], [4, 8]]
```

Note: this exercise is quite hard.

2 Python's Built-in Types

2.1 Ranges

Exercise 2.1: Trying Ranges

The list() function creates a list from a range. Use this to experiment with ranges, checking that you understand how they work. For example:

```
range(10)
range(10, 20)
range(10, 40, 2)
```

Here is an example:

```
>>> r = range(10)
>>> list(r)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Exercise 2.2: Using For Loops

Practice using for loops. For example, rewrite the following while loop using a for loop and a range:

```
x=0
name="william"
while x < len(name):
    print(x,":",name[x])
    x = x + 1
```

Create and solve some similar problems.

Exercise 2.3: Teaching For Loops

Discuss with another course member when / whether to teach for loops (and ranges) in Python.

2.2 Tuples

Exercise 2.4: Trying Tuple

Enter a tuple such as: `t = (3,1,5,8)` Then try each of the follow expressions to see if a) it is legal b) what the result is:

```
t[1]

t[1:2]

t[1:3]

len(t)

t[2]=7
```

Exercise 2.6: Two Lists to a List of Pairs

Write a function that takes two lists and returns a list of pairs. This function is often called 'zip'. For example:

```
>>> zip([1,2,3],["a","b","c"])
[(1, 'a'), (2, 'b'), (3, 'c')]
```

2.3 Sets

Exercise 2.7: Multiples of 7 and 5

Suppose we want to get the multiples of some number N, below a limit L. We can express this as a range: range(N, L, N). For example:

Use this idea and the set intersection operation to write a function to give multiples of two numbers below some limit. Here are some examples:

```
>>> multiplesNandM(5, 7, 100)
[35, 70]
>>> multiplesNandM(13, 17, 1000)
[442, 884, 221, 663]
```

2.4 Dictionaries

Exercise 2.8: Shopping List Representation

A shopping list looks like:

Apples	12
Biscuits	1
Crisp packets	3

The following representations are suggested:

- A pair of lists: `(['Apples', 'Biscuits', 'Crisp packets'], [12,1,3])`
- A list of pairs: `[('Apples', 12), ...]`
- A dictionary

Try out two or more of these. Write the following functions:

1. A function to add a new item to the list.
2. A function to look up the number of items required (or report that it is not on the list).

Exercise 2.9: Representation Challenge

Suggest **two** representations each for minesweeper and / or hangman:

- Write Python to create examples
- Write Python to update the state: e.g. a new location in the minefield tested or a new letter guessed in hangman.

Discuss the pros and cons of the representations. Which will be easier to use?

3 Exceptions

Exercise 3.1: Number in Range

Write a robust function to ask the user for a number in a specific range. The user is repeatedly prompted until a number in the correct range is entered. Print a suitable error is a) the entered text is not a valid number or b) the number is out of range.

Exercise 3.2: Opening Files

Write a function that prompts for the name of a file and the prints the number of lines in the file, before closing it. However, the function should not fail if the given file does not exist: instead print an error message.