2810ICT/7810ICT Software Technologies, Trimester 3, 2019

Workshop 2 - Python Revision 2

When	Day 2
Goal	In this workshop you will continue your Python revision by solving problems with data structures and objects.
Marks	6
Due	Solutions to the marked exercises are due by the end of your day 2 workshop. You can complete them beforehand if you wish, but you will need to reset your solution and demonstrate your ability to write the code in your workshop class.

1. Preparation

Before your lab class:

- Read all of this document.
- Review the lecture notes for days 1 and 2.
- If you have not already, ensure that you have completed the Griffith Sciences Laboratory Induction test on Learning@Griffith, within the last year. Print or save the certificate and bring it to your lab class.

2. Workshop activities

2.1 Task 1

Set up your Python programming environment:

- Make sure you can access the Python problem set
- Please see the Python problem set section of the L@G page for instructions on how to do this.

2.2 Task 2

Complete the following problems as part of your Python revision. Your tutors will have some suggestions if you get stuck. The marks for some exercises are indicated.

- **W1C2 (lists - easy)**
- **W1G1 (lists - hard) (UG, Ma: 2 mark)**
- **W2B2 (files)**
- **Dictionaries/sets/tuples (UG: 2 marks, Ma: 1 mark)**
  - Download mbox-short.txt from the problem set page on Learning@Griffith.
  - Read file "mbox-short.txt" and record the domain name where the message was sent from instead of who the mail came from (i.e., the whole email address). At the end of the program, print out the values of your dictionary sorted by domain name.
- **Classes (UG, Ma: 2 marks)**
  - Define a bank account class called BankAccount which takes name of customer and balance as input for the constructor.
  - Write a method called deposit that adds an amount to the account's balance.
  - Write a method called withdraw that subtracts an amount from the account's balance and print the remaining balance. If the initial balance is lower than the requested amount, output an error message.
- Classes
  - Define a MyMath class and implement addition, subtraction, multiplication, and equal functions in __add__(self, other), __sub__(self, other), __mul__(self, other), and __eq__(self, other). See https://docs.python.org/3.3/reference/datamodel.html#emulating-numeric-types.

Masters students and undergraduates seeking more practice should also attempt:

- **W1M1**
- **W1E6 (list/maths) (Ma: 1 mark)**
- **W1E4**
- **W1F6**