FIT1045 Algorithmic Problem Solving – Workshop 3.

Objectives

The **objectives of this workshop** are:

- To get familiar with Python and debugging code.
- To implement and manipulate the list and table data structures in Python.
- To implement iteration and selection in Python.

Useful Material

Iteration and Selection: Documentation on control statements is available at `https://docs.python.org/3/tutorial/controlflow.html`. A chapter on flow control is available at `https://automatetheboringstuff.com/chapter2/` (A link to this book is available at `https://wiki.python.org/moin/BeginnersGuide/NonProgrammers`).

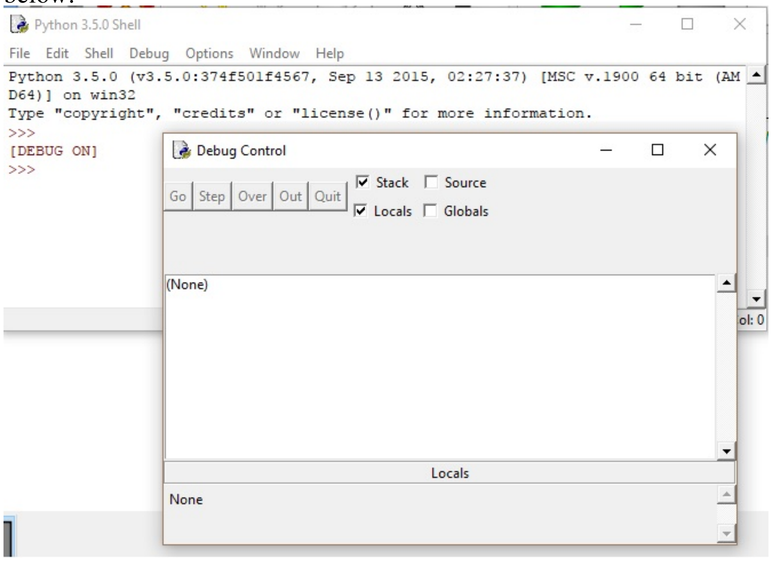
Lists and Tuples: Documentation on lists is available at `https://docs.python.org/3/tutorial/datastructures.html`.

Task 0: Prelab

Use the Python interactive shell to experiment with the creation and manipulation of lists. A good introduction would be to complete the exercises at `http://www.diveintopython.net/native_data_types/lists.html`.

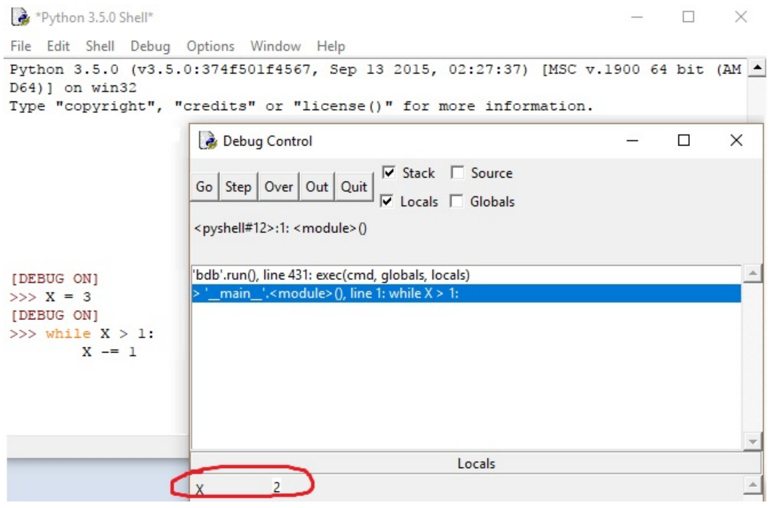
Task 1: Using IDLE's debugger

IDLE provides a simple debugger that you can use to step through your code *(without PythonTutor's restricted number of steps)*.
You will be using this debugger to step through a simple loop
In order to to this you will click on the **debug -> debugger** option from IDLE's main menu and a window will open as below:



```
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:27:37) [MSC v.1900 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
[DEBUG ON]
>>>
```

This has a button called step which allows you to move through the code one line at a time and values appear down the bottom
Note: this is much easier to do when your code is saved as a file otherwise you'll be clicking the step button a lot



```
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:27:37) [MSC v.1900 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.

[DEBUG ON]
>>> X = 3
[DEBUG ON]
>>> while X > 1:
        X -= 1
```

[✓ Stack] [ ] Source
[✓ Locals] [ ] Globals

```
<pyshell#12>:1: <module>()

'bdb'.run(), line 431: exec(cmd, globals, locals)
> '__main__'.<module>(), line 1: while X > 1:
```

Locals

```
X          2
```

Write up a python program that implements the following algorithm.

```
Input: Two numbers $X$ and $Y$ representing numbers to multiply together
Output: An integer holding the result of $X * Y$
1 $Sum \leftarrow 0$
2 while $Y > 0$ do
3     $Sum \leftarrow Sum + X$
4     $Y \leftarrow Y - 1$
5 return $Sum$
```

Step through this code with IDLE's debugger and notice how the values change as you use the step command.

Note: there are many great IDEs (integrated development environments) which include debuggers that allow you to stop at a chosen line. We don't cover these in this unit, but if you want to invest a small amount of time into learning one we suggest pycharm (`https://www.jetbrains.com/pycharm/`)
Your demonstrator will be able to help you with using pycharm if you need

Task 2: Lists

Discuss as a group methods to iterate over a list.
Write a Python program that uses a while loop to ask the user to input 5 numbers. The program should then enter each of these numbers into a list. Then, once all numbers are entered, prints the list.

For example:

```
Enter a Number: 1
Enter a Number: 3
Enter a Number: 6
Enter a Number: 2
Enter a number: 3
[1, 3, 6, 2, 3]
```

NOTE: The output ['1', '9', '5', '10', '11'] is a list of strings and is not the correct answer.

Modify your program such that it outputs the list containing the square of each entered number.

For example:

```
Enter a Number: 1
Enter a Number: 3
Enter a Number: 6
Enter a Number: 2
Enter a number: 3
[1, 9, 36, 4, 9]
```

Task 3: Assignment and Indexing

Now let's consider how to assign elements into a list

1. Discuss with the person sitting next to how you can assign values to a specific index of an list
2. Run the code given in 'list1.py' (available on Moodle)
3. Discuss the result with the person sitting next to you
4. Determine and correct the problem together **without changing lines 5 or 6**

Once you are comfortable with the notion of simple list assignment, modify the program you wrote in task 2 to use list assignment.

NOTE: Your demonstrators are there to help, if you cannot find a solution try discussing the problem with them, they might have some good ideas.

Task 4: Lists of Lists, Tables and References

Discuss how lists containing lists can be used to represent tables in Python.
Write a program in Python that asks user to input 5 lines of numbers. Your program should represent these lines as a table.

NOTE: Think carefully about which lists will have a fixed size and which lists will have a variable size.

For example:

```
Enter some numbers: 1 2 3 4 5
Enter some numbers: 5 6 7
Enter some numbers: 1 1 1 1
Enter some numbers: 1.5 6 7.5 2
Enter some numbers: 0
[[1, 2, 3, 4, 5], [5, 6, 7], [1, 1, 1, 1], [1.5, 6, 7.5, 2], [0]]
```

NOTE: As the input function returns a string, you will be required to split the string. The documentation for string splitting can be found here `https://docs.python.org/3/library/stdtypes.html#str.split`

Once you have some experience with tables as lists of lists, consider the following line of code:

```
new_table = [[1, 2, 3]]*5
```

Using the techniques you have learnt so far, work out what this code does. As a pair discuss what you think will happen when you change the first element of the first list to 0.

```
new_table[0][0] = 0
```

What happened? Did the table behave as expected? If not, what possible explanation could there be for the behaviour of the table? Discuss your findings with your demonstrator.
If you haven't already, you might light to step through the code with PythonTutor and IDLE's default debugger.

Task 5: Using Lists

Write a program in Python which simulates the rolling of 1000 dice, by generating a number from 1 to 6, and stores these numbers in a list. Calculate the mean average (a.k.a. arithmetic mean) of this list but adding each element together and dividing by the number of elements.

NOTE: The arithmetic mean is more precisely given by,

$$\sum_{i=0}^{n-1} \frac{x_i}{n}$$

where x_i is the i^{th} element of a list of size n.

Task 6: Tables and Matrices

This task relates to some common matrix operations. In this question, we will use the term matrix to refer to a list of length m, where each element is a list of length n. This represents a matrix with m rows and n columns. A shorthand for such a matrix is an $m \times n$ matrix.

1. Write a function which takes an input an $m \times n$ matrix and returns a $n \times m$ matrix which is the transpose of the original. Transposing a table means flipping the table around its diagonal such that the i-th column of the original table becomes the i-th row of the transposed table and vice versa (for a more detailed definition you can check "transpose" on Wikipedia).

2. Write a function which takes as input a vector and a matrix, and multiplies them. This means that it constructs a new vector, where the element at position i is the dot product of the input vector with column i from the matrix.

3. Consider the following table:

```
cols = ['energy', 'water', 'protein', 'carbs', 'sugars', 'fat', 'fiber']
rows = ['apple', 'orange', 'broccoli', 'beef', 'lamb', 'bread']

nutr_vals = [[229, 84.3, 0.4, 12.0, 11.8, 0.0, 2.3],
             [186, 84.3, 1, 9.5, 8.3, 0.2, 2.1],
             [124, 89.6, 3.2, 2.0, 2.0, 0.1, 4.1],
             [613, 70, 22.8, 0.2, 0.0, 6.0, 0.0],
             [1057, 60.2, 18.6, 0.0, 0.0, 20.2, 0.0],
             [1446, 37.6, 8.4, 43.5, 1.5, 2.6, 6.9]]
```

   It details dietary information about several different foods. For example, the first row contains the energy, water, protein, carbohydrate, sugars, far and fiber content (in g) of an apple (per 100g).
   You are given a list of length 6, where each index corresponds to one of the 6 foods in "rows". Each element in this list is a number, indicating how much of that food a person eats. For example, the list `[1,0,0,3,2,1]` indicates they eat 100g of apple, no orange or brocolli, 300g beef, 200g lamb and 100g bread. Your task is to output a list containing 7 elements, one for each nutrient type, containing the total amount of that nutrient that would be consumed by eating the given amounts of each food.

Extension Question

Imagine you have a deck of cards, placed one by one in a row face down. You have the ability to flip a single card. However this triggers the card immediately to the left to flip also. If there is no card to the left then you ignore this rule. Your goal is to write a program that simulates this situation and finishes when all cards appear face up.

NOTE: Think carefully about how you should represent the cards.

ADVANCED NOTE: This problem has some interesting properties and as such has some interesting questions. Try changing the starting state of the game, mixing cards that are face up with face down. Is there any configuration of the starting state that makes the problem solvable/unsolvable?