COM4013 – Introduction to Software Development
Week 14 – Lists, Stacks, and Stack Class Python

*Always select "**Enable Editing**" if prompted by Microsoft Word*

Lab Exercises

Always refer to the lecture notes for examples and guidance.
Also look at your previous work, many exercises are similar.

There are two "stages" marked in the lab sheet. Stage 1 is the *absolute minimum* point you should reach. Hopefully, you can reach it in the lab session. Reaching stage 2 suggests you are on target for a top-class mark for the module. Set your goals sensibly - do not just aim for the minimum or you may struggle to pass the module.

I use the term integer, string and float, and Boolean to infer what the input/output looks like. Unlike many other languages Python does not expect a data type before the variable names.

So if I ask that you declare an integer num1 to 1, then do as follows:

```
num1 = 1
```

For declaring a string and assigning the value hello we can do this

```
greeting = "hello"
```

For floats we can declare it like this

```
Money = 2.50
```

For Booleans (bool) we can declare it like this

```
isHeavy = True
```

- Create a Jupyter Notebook project called **Lab 14 Exercises**. Refer to Week 1's lecture/worksheet/video if you have forgotten how to create a new project/file/program or use the shell code.

Nested 'for' Loops (From semester 1 – Week 5)

- Try to do these on your own first...
- Write a 'for' loop that displays 10 X's in a row.

  XXXXXXXXXX

♦ Add a newline print (in the same cell), ***nest*** this 'for' loop in another 'for' loop to display a 10x20 box of X's. **Again, this is shown in the lecture notes in Python**.

Note: You must use nested for loops in all these exercises. ***Do not try to solve these problems with many repeated print() statements!***

XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX

Create a List

- Copy the following code into your cell.

```
def main():
    list = [ 2, 5, 8, 3 ];

if __name__ == "__main__": main()
```

- Output the first element of the List list. The List index begins at 0.
- Output the last element of the List. Remember that the last index of the last element of the List will be *1 less* than the size of the List
- Use a *for* loop to output the contents of the List.

- Declare a List, of type integer and initialise the List in the declaration to the values:

- Print the number of elements in the List:
- Print the List contents in **reverse** using a **for in range loop**.
- Print the contents of the list using the **for (each element) iterator loop**.

Rainfall

- Declare a List of type *int* and call the List rainfall.
- When you declare the List, initialise the elements of the List to the values:

4, 2, 0, 5, 1, 3

- Use a *for* loop to go through the List.
- Calculate the lowest (min) rainfall, the highest (max) rainfall and the total (sum) amount of rainfall.
- Print to the console.

Nested For Loop List

- Write a program that creates and the following 3 by 3 matrix List.

```
1 2 3
4 5 6
7 8 9
```

- Copy this code into your program, to begin with: it's a 3*3 matrix.

```
list = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
```

- Use a nested for loop or another method to assign the values above to the list
- Use a nested loop to display all of the elements of the List. (See the lecture slide to help you with this).
- Write another loop to find the sum of all of the elements in the List.

- Extra task: Use a for each element iterator loop to view the matrix
    - Note the square brackets and commas

Empty List Fill

- Write a program that fills an empty four-element integer List:

```
list = [0, 0, 0, 0]
```

…with multiples of two and then displays the contents of each List element.

- The output should look like:

```
2 4 6 8
```

- **Advance task:** in Python there's a way to assign these values to a List in one line, and there is also a one-line way to print a Lists content… Try to pull it off.

Summing Two Lists

- Write a program to create **three** Lists.
- Initialise the first List to:

```
1 3 5
```

- Initialise the second List to:

```
2 4 3
```

- Initialise the third list with 3 0's.
- Add the first and second List together and place the result in the third List. You need to add the Lists together element by element. The values held within the third List at the end should be:

```
3 7 8
```

List Search

- Create an integer List of size 6 and initialise it to the values: 4 6 8 5 2 3.
- Write a loop to go through each of the elements of the List.
- Test each of the elements of the List to see if it is equal to a value of 5.
- If the List element is equal to a value of 5 then output "found" and the *index* of the element.
- Test your code. You should find an element with a value of 5. The index of the variable is 3.

Print List

- Write a function called PrintList. The function accepts a single dimensional integer List.
- The function should output all of the elements of any List which is sent to it.
- Initialise a List of size 10 to the values: 8 6 5 4 3 1 2 5 7 8.
- Pass the List to the function PrintList.
- Test the function in your main function.

Set List Element Values to 0

- Write a new function called SetToZero. The function has a single parameter: an integer List.
- Add an integer type hint to your function – for readability.
- The function will set each element to of the List passed to a value of 0, i.e. assign a value of 0 to each element of the List.
- Test your function using SetToZero and PrintList in main.

Print List

- Complete the stack class – look at the lecture slides to find the last few methods names, and try to implement them.
- If you find that you don't know what those are meant to do – then ask…
- Test it in main