COM4013 – Introduction to Software Development
Week 3 – Elif, Operators, and While Loop

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

Lab Exercises

The exercises in this lab follow the topics in the lecture notes. Refer to the lecture notes for examples and guidance.

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 to infer what the input/output looks like. Python does not expect a data type before the variableName like many other languages.

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

```
greeting = "hello"
```

For floats we can declare it like this

```
Money = 2.50
```

For Booleans (bool) we can declare it like this

```
isHeavy = True
```

Expressions / Operator Precedence

- For each of the following expressions calculate the two possible answers, then say which one Python will use. See the lecture notes for details on this exercise. Check your answers with the tutor.

  ```
  5 + 3 * 6
  12 / 3 + 1
  4 * 8 / 2
  ```

Increment, Decrement and Assignment Operators

♦ Create a new Jupyter Notebook project called **Lab 3 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.

♦ Copy the code below into the correct place in the shell code (*Use copy and paste, don't type it in!*). Remember to paste into the correct location in the shell code:

```
userNumber = int(input("Enter a number, any number: "))
startNumber = userNumber

print("Now add 1...")
userNumber = userNumber + 1
print(f"You now have { userNumber }... ")

print("Double it...")
userNumber = userNumber * 2
print(f"You now have { userNumber }... ")

print("Now add 4...")
userNumber = userNumber + 4
print(f"You now have { userNumber }... ")

print("Now divide by 2...")
userNumber = userNumber / 2
print(f"You now have { userNumber }... ")

print("Now take away the number you first thought of...")
userNumber = userNumber - startNumber
print(f"You now have three : { userNumber }")
```

♦ Run the code, it's a simple maths trick - the result is always three.

♦ Comment this code, to a level that makes sense

♦ Update the code to use the operators introduced in this week's lecture such as:

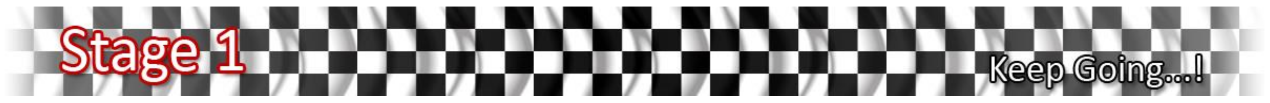
```
+=   -=   *=   /=
```

Make sure that the trick still works after your changes

– Notice how this program uses just one line to read an integer:

```
userNumber = int(input("Enter a number, any number"))
```

In previous exercises, this may have been done in two lines and needed an extra string variable for the input message. You might want to use this shorter version in your programs (I leave it to you to decide).



And, Or in 'if' Statements

- Create a new cell in Jupyter Notebook (use the shell code).

- Copy this code into the shell code. It's the example from the lecture to display if the user is in their twenties.

```
int age = int(input( "Enter your age: " )

if (age >= 20 and age <= 29):
    print("You are in your twenties")
```

- Add an 'else' statement to display a message when the user is **not** in their twenties

- Add more 'if-else' statements to display if the user is in their thirties, forties, or if they are a teenager (13-19). Refer to the lecture notes, the slide titled, *"Chaining 'if' and 'else' Statements"* to see the style of code required.

- Leave the else statement as is

A Simple 'while' Loop

- Create a new cell in Jupyter Notebook (use the shell code).

- Copy this code into the shell code. It's the counting example from the lecture.

```
counter = 1
while (counter <= 10):
    print( counter )
    counter+=1
```

- Run the code and see that it works.

- Add the following code `, end=', '` to the print statement after counter.
    - What does it do?

- Add the following code `print("\n")` after the while loop
    - This is the newline Unicode character that adds a line to our code.
    - Note that this can be used in conjunction with any string based aspects of our code such as within `input( )` to create more neatness in our code.

A more complex 'while' Loop

- Now add a completely new while loop afterwards that displays **the even numbers only** from 0 to 100. This will be a very similar piece of code.
  - Be careful about declaring the same variable twice.
  - Assign a new value to the `counter` variable
- Display it on a single line, separated by commas like this…

```
0, 2, 4, 6, 8, 10, 12, 14...
```

- Change your loop so the numbers appear in reverse order, 100, 98, 96...
- Comment out the `print("\n")` before the second while loop to see what happens.

Stage 2 ~Boss Battle~

Adventure Game (The answer to these will not be released: Ask for advice as you go along)

Now we will use the techniques we have covered to create a text-based adventure game.

- Create a new cell in Jupyter Notebook (use the shell code).

Starting the Game

- First present an opening message welcoming the user to the adventure. It's a fantasy scenario with warriors and dragons so give it a suitable name.
- Look at the code snippet below. Implement a start screen task where you ask the user to press any button to start the game.

Code Snippets

# Code that stop the code from executing until a key is pressed

```
import keyboard  # Just put this with your other libraries

# Some code you want to run - ask user to press any button to begin

keyboard.read_event(suppress=True)

# Code that will run once the key is clicked
```

- You might think the screen is a little dirty looking, and with too much text. So, lets clean it up by clearing the console.

```
from IPython.display import clear_output

clear_output()  # Use this function wherever we want to clear output
```

- Next declare an integer variable called `characterHealth`. Initialise it to 100.
- Next the user must choose to be a warrior or a scout. A warrior is stronger, but a scout is faster. Declare a variable called `characterType`. Ask the user whether they want to be warrior or scout, and based on their answer assign "Warrior" or "Scout" to your `characterType` variable.
    - In this exercise the exact text of the user messages and the form of the user input is up to you. You may hit problems if you get incorrect user input, e.g. if you are expecting the user to type "Warrior" with a capital 'W', but they type it with a lower case 'w'.

The First Trial - A Riddle

At the start of the adventure, the character first reaches a huge stone door, which won't open. A huge voice booms out:

- Answer this riddle to pass safely: What can you catch but cannot throw?
- Write some text introducing this situation to the user. You might want to format your output nicely: use multiple WriteLine statements to stop the words from wrapping. Also, you can write blank lines like this:

    `Print( )` or by using `\n` newline Unicode in the input itself

Ask the user for the answer to the riddle storing the answer into a string variable

- Then use an 'if-else' statement to test if their answer was correct. The answer is "a cold", but write your 'if' statement to accept either "a cold" or just "cold". Use the Python or operator - read the lecture notes if you have forgotten how to write it.
- In the first block of the 'if-else' statement, when the user guesses correctly, the booming voice should say something like "Well done, you may pass..." and the door opens. Simply display text to describe this.

- However, in the 'else' block, when the user guesses incorrectly, the voice says nothing, the door opens and fire jets out reducing the character's health. How much damage depends on the character type. Add another 'if-else' statement inside this block (this is a nested 'if' statement as discussed in the lecture). If the character is a warrior their health is reduced by 20, but (else) a scout will only have their health reduced by 10 as they dodge away from the fire quickly. Use appropriate messages to describe these events.

- Optional: Actually, other reasonable answers to the riddle are "a bus", "a train" etc. Extend your 'if-else' statement to include `elif` statements (in a similar way to the Decades exercise above) so there are three alternatives: the "correct" answer (a cold), these alternative answers (a bus, a train), and the wrong answer. When the user types an alternative answer, the booming voice should say something sulky ("There are no such machines in this world...") but let the character pass unharmed anyway.

THAT IS ALL FOR NOW

Submission

1. Please refrain from submitting your work to the Teams area. Last week marked the final instance in which I will request you to submit your practical session assignments. If you have any doubts about your work, you can continue submitting, but kindly seek feedback during the session. Given that some of you possess advanced programming skills, it might be more beneficial to review my solutions and compare them to your own understanding.