1.1 - Week 1 - Applied - Theory

Objectives of this Applied Session

- To get you to know your tutor and the other students (even if it is online).
- To discuss any doubts you have regarding the organisation of this unit.
- To revise some of the material you covered in FIT1045 or equivalent.
- To get an idea of your skill level for understanding and implementing basic programming tasks.

Important

Some quick info about Applied Session classes:

- The purpose of these classes is to give you a chance to attempt to problem solve with the content in the unit, as well as generate discussion about challenging and interesting concepts.
- Solutions are released weekly.
- It is recommended that you prepare in some manner beforehand for the class. At least be familiar with the lecture content the applied class is on.
- While we encourage all students to attempt all questions in the applied theory and practical classes, it is worth noting that we try to leave some particularly difficult extension tasks near the end of the class. Don't feel bad asking for help, from your TAs or your peers!

FIT1008 - Assessments - Group Work and Contract

As you may already know, the assessments in FIT1008 are all either individual or group based (1st solo, 2nd and 3rd group), so you will be selecting a group that you will work with for the final two assignments.

To ensure a fair share and high standards of group work, we are providing a template for a group contract that you **must** fill in, sign, and submit to Moodle before your group can be fully finalised. The deadline for selecting your group (of 4) and submitting the completed group contract is **End of Week 3, Sunday the 14th of August**.

Failure to submit this document by then will result in you being allocated to a randomised group.

This document should also be submitted as a part of your assignment submission for A2 and A3. Please note that some of the fields can be carried over to each assessment, however you may need to edit the others.

Template: A link to the template can be found here. Please note that you will need to be signed in with your Monash account to get access to this link. You will need to download this template, edit it on your computer and then upload it as a part of your submission for the group assignment.

If you have any questions with regards to filling in the document, please ask your TA in either of the Applied sessions.

Algorithms and Scalability

Consider the following algorithm:

```
define function mystery(my_list)
    count = 0
    add = 0
    i = 0
    while i < length(my_list) {
        print "Outside"
        count = count+1
        add = add + my_list[i]
        j = 0
        while j < length(my_list) {
            print "     Inside"
            count = count+1
            add = add + my_list[j]
            j = j + 1
        }
        i = i + 1
    }
    print "Addition: " + add
    return count
```

Lets assume that for the first 4 questions, we pass a list `[1,2,3,4,5,6,7]` of length `7`

and for the remaining questions, we pass a list `[1,2,3,4.....n-2,n-1,n]` of length `n`

Question 1 *Submitted Jul 25th 2022 at 7:23:09 pm*

If `my_list` has a length of 7, how many times does the algorithm print "*Outside*" and *"Inside"*?

- 0, 0
- **7, 49**
- 14, 98
- 49, 7

Question 2 *Submitted Jul 25th 2022 at 7:23:33 pm*

What is the value of `count` returned for a list of length 7?

- 0
- 7
- 49
- 56

Question 3 *Submitted Jul 25th 2022 at 7:23:45 pm*

What value is printed for `add` when `my_list` contains numbers 1,2,3,4,5,6 and 7?

- 0
- 28
- 224
- 784

Question 4 *Submitted Jul 25th 2022 at 7:23:54 pm*

For a list of length `n`, how many times is `Outside` printed?

- 1
- n
- n^2
- $2n$

Question 5 *Submitted Jul 25th 2022 at 7:24:00 pm*

For a list of length `n`, how many times is `Inside` printed?

- 1
- n
- n^2
- $2n$

Question 6 *Submitted Jul 25th 2022 at 7:24:07 pm*

What is the value of `count` returned for a list of length `n`?

- 1
- n
- $2n^2 + n$
- $n^2 + n$

Question 7 *Submitted Jul 25th 2022 at 7:24:31 pm*

For a list `[1,2,3,4.....,n-2,n-1,n]` that contains `n` elements, what is the value of `add` ?

- $n^2 + n$
- $\frac{n(n^2+n)}{4}$
- $\frac{n(n+1)^2}{2}$
- I have no idea, this is too much math

String Manipulation

A string can be seen as a list of characters, and we could then use the notation `w[k-1]` to denote the k^{th} character in the string **w**.

A palindrome is a string that is spelt the same ways forwards as backwards. For example, string **ab1ba** is a palindrome, while strings **a3bbbaba** and **321236** are not.

Write a Python function `is_palindrome()` in `Exercise 2.py` that takes as a parameter a string and returns **True** if the string is a palindrome, and **False** otherwise. Assume all characters are lowercase alphanumerics and, thus, you do not need to deal with issues brought by matching uppercases to lowercases.

> Try to do it without using any of the magic python functionality. Try to do it intuitively and implement a solution that will be useful in all programming languages

- What is the Big O complexity of your algorithm? Write your answer and explanation in `explanation.txt`.

Brief Conceptual Explanations

In this unit, we are going to be studying various topics that will build on your pre-existing knowledge of algorithms and data structures.
Try to answer the questions in the next quiz from the knowledge that you have. Your demonstrators will then go through the concepts **briefly** with you to prepare you for the semester ahead!

Here we go!

Preemptive quiz

Question 1 *Submitted Jul 26th 2022 at 9:12:22 am*

What is the difference between an upper bound and lower bound for the runtime of an algorithm?

Lower bound for the runtime of an algorithm is the most efficient way an algorithm can run.

Upper bound for the runtime of an algorithm is the least efficient way an algorithm can run.

Question 2 *Submitted Jul 26th 2022 at 9:12:27 am*

How does a stack differ from a queue?

Stack operates by last-in-first-out where all addition and removal of data is done at the same side.

Queue operates by first-in-first-out where addition and removal of data is done at separated side.

Stack is usually faster than queue as addition and removal of data is done at the same side.

Question 3 *Submitted Jul 26th 2022 at 9:14:12 am*

How does recursion differ from iteration?

Recursion is usually a whole function where there is a process applied to data.

- Base
- Recursion

Iteration is usually applied to a function where the aim is allow the said function to be repeatedly executed.

- For loop
- While loop

Recursion is slower than iteration due to it being stored in stack.

Scalability Pt 2

Alright! We are going to flirt a little more with complexity! Although you may not know a lot about this topic, you will be taught about complexity later in the unit. Until then, try and answer these questions to the best of your knowledge and intuition

Consider the following function:

```
def mystery(n: int) -> int:
    total = 0
    for i in range(1, n+1):
        for j in range(1, i+1):
            total += i
    return total
```

And now, try to answer the following questions below:

Question 1 *Submitted Jul 26th 2022 at 9:22:39 am*

What does this mystery function do?

Return the sum of square value of n

Question 2 *Submitted Jul 26th 2022 at 9:20:57 am*

What is the best and worst case time complexity of this function?

- Best Case $O(1)$, Worst Case $O(n)$ where n is the input integer
- Best Case $O(n)$, Worst Case $O(n)$ where n is the input integer
- **Best Case $O(n^2)$, Worst Case $O(n^2)$ where n is the input integer**
- Best Case $O(n^2)$, Worst Case $O(2^n)$ where n is the input integer

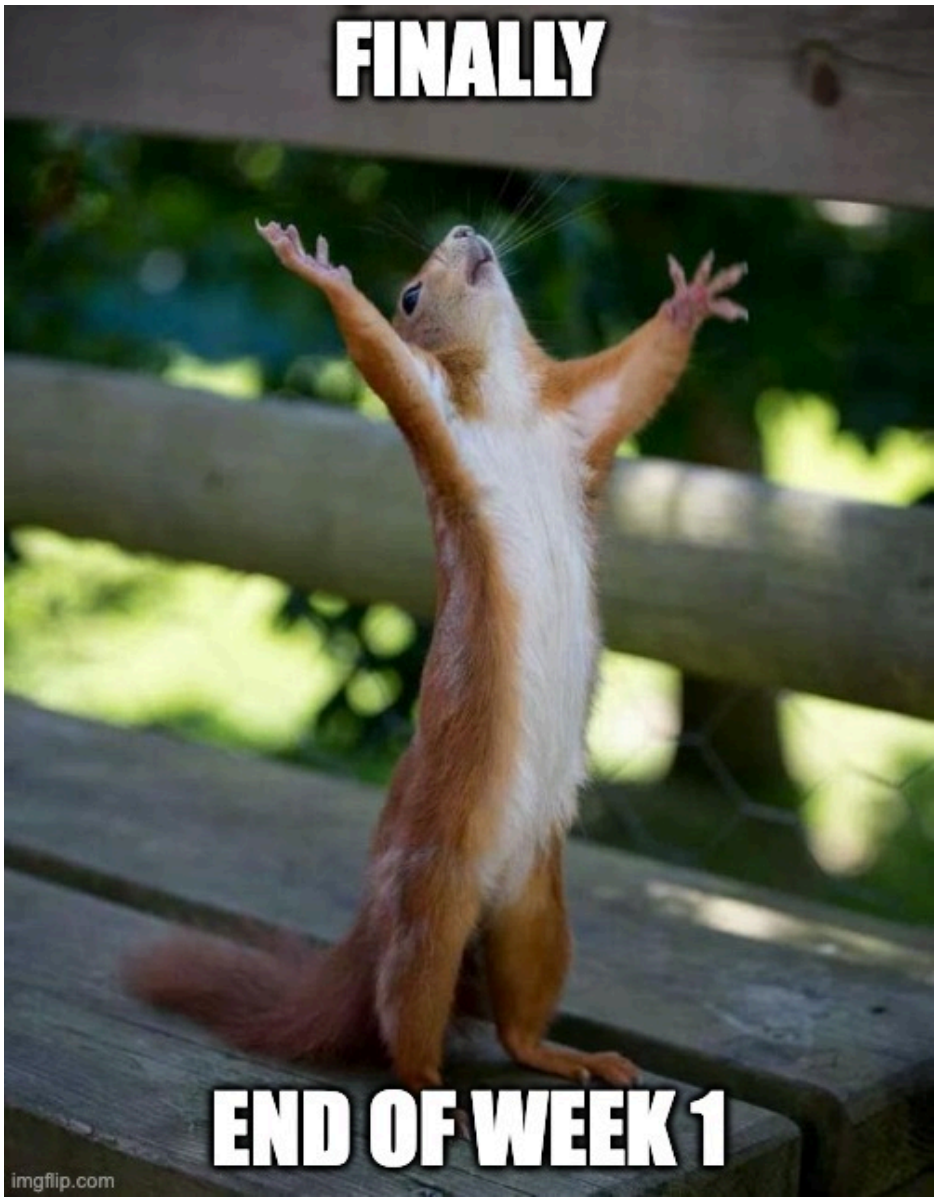
Question 3 *Submitted Jul 26th 2022 at 9:22:23 am*

Can you find a function that provides the same output as this mystery function, but has a better runtime?

```
def prob_better(n):
    return ((n * (n + 1) * (2*n + 1)) / 6)
```

Runtime of O(n)

END OF LESSON (Or is it?)



We have reached the end of the compulsory exercises.
Do the rest of the exercises if time persists in the class or as post class activities

Difficulty Level: Legend

In `list_intersection.py` write a Python function `print_intersection()` that takes as parameters two lists, **list1** and **list2**, and prints out all the items that belong to both lists.

For example, for lists [2, 4, 8] and [9, 3, 2], it should print 2.

Assume no element appears twice in the list.

Please note: Chad programmers don't use the `set` function. Try doing this question without using it.

Difficulty: Nightmare

Reversing a string sounds easy, but what about reversing an integer? In `reverse_integer.py` Define a Python function `reverse_integer()` that takes an integer **n** as a parameter and returns integer **m**, which value is the reverse form of integer **n**.

For example, with an input integer 2941, it should return 1492.

You are not allowed to use a string to reverse the given integer.