# *FIT1008 – Intro to Computer Science*
# *Tutorial 6*

Semester 1, 2018

*Objectives of this tutorial*

- To understand Classes and Objects.
- To understand namespaces and scoping.
- To understand memory diagrams.

## *Exercise 1*

This exercise and the next one are about scoping and namespaces. Examine these nested functions:

```
def silly():
    x = 'ping'
    def g():
        print(x)
    x = 'pong'
    def f(x):
        print(x)
    g()
    f(x)

silly()
```

We expected the `silly()` function call to print "ping" and then "pong". What will it print? Why?

## *Exercise 2*

More scoping and namespaces. Now, with objects! A student named Juan is asked to make a Member class for a community pool. The only thing we care about is our pool's name, and our members' name, age and gender. So Juan writes:

```
class PoolMember:
    poolname = "FIT1008␣students␣community␣pool"
    name = ""
    age = 0
    gender = None

    def __init__(self, name, age, gender):
        PoolMember.name = name
        PoolMember.age = age
```

```
        PoolMember.gender = gender
```

Juan makes himself the first member of the pool, and also its admin. It all seems to go so well!

```
>>> admin = PoolMember('Juan', 18, 'male')
>>> admin.name
'Juan'
>>> admin.age
18
>>> admin.gender
'male'
>>> admin.poolname
'FIT1008 students community pool'
>>>
```

But an order comes from above. The pool has to be open to any Monash students and staff, not just FIT1008 students. And we need to change the name before other students come! So we tell Juan to change the pool name before we make the new membership card for a well-respected professor, Emilia, who will be our supervisor:

```
>>> admin.poolname = "MONASH all-inclusive community pool"
>>> admin.poolname
'MONASH all-inclusive community pool'                # it looks allright
>>> supervisor = PoolMember(Emilia, 26, 'female')    # let's make Emilia's user
```

Phew! Disaster averted! All seems alright again, until Emilia wants to check that the pool's name was really changed

```
>>> supervisor.poolname
'FIT1008 community pool'          # what, this can't be!
>>> PoolMember.poolname           # let's check the global name
'FIT1008 community pool'          # it's the same!
```

It's still the old name! Emilia wants to find who is responsible for the mistake, so she looks at the details for the admin:

```
>>> admin.name
'Emilia'
>>> admin.age
26
>> admin.gender
'female'
```

- What? Is Juan framing Emilia for his mistakes? Or is it just an unfortunate series of bugs?
- What did Juan do wrong? Can you write the correct class and the correct way to change the pool's name?

## *Exercise 3*

- Draw a memory diagram representing the state of memory after the following instructions have been executed.
- What is printed in the end?
-

```
numbers = [1, 2, 3 ]
other = numbers
numbers.append(4)
x = numbers[0]
x += 1
print(other)
print(numbers)
```

-

```
a_matrix = 3*[3*[None]]
a_matrix[1][2] = 3
print(a_matrix)
```

## *Exercise 4*

Given below is a snippet of a problem description:

*A coffee store in a popular shopping centre is planning to introduce a customer loyalty system. This program is designed to offer rewards for their frequent customers, as well as make their ordering process easier. Each customer has an ID (assigned automatically by the system), name, phone number, and a count of loyalty points. The system will record points earned, accumulated by each customer when they order their coffee. These points can then be redeemed (used) for free coffee on subsequent visits. The system also manages up to 5 different coffees that the store makes. Each coffee has a name and price (in whole dollars). Each coffee is also half price on one day of the week. This day is different for each coffee, and is checked when an order is placed.*

1. Describe the classes (including attributes) you would require for a program to implement this customer loyalty system.