COSC1076 | Semester 1 2021 | Advanced Programming Techniques

# Tutorial/Lab | Week 03

## Overview

The week 03 tutorial/lab is for you to practice working with:

- Typedef
- Array representation in C/C++
- Pointers
- References
- Allocating and De-allocating memory

The code snippets can be accessed from the COSC1076 Git Repository.

## Lab Questions

*It is a good idea to attempt the lab questions before coming to class. The lab might also be longer than you can complete in 2 hours. It is a good to finish the lab at home.*

1. Examine the following code, contained in various files and answer the questions:
   (a) Complete the `#include` statements.
   (b) What is an alternative way for writing line 11 in `main.cpp`?
   (c) What important method is this `Example` class missing? Explain why this is a serious issue.
   (d) The `Example` class contains a siginificant couple of errors. Find and fix the errors.

```cpp
class Example {
public:
   Example(double value);

   void setValue(double value);
   double getValue();

private:
   double* ptrValue;
};
```

```cpp
#include "???"

Example::Example(double value) {
   this->ptrValue = new double(value);
}

void Example::setValue(double value) {
   this->ptrValue = value;
}

double Example::getValue() {
   return this->ptrValue;
}
```

```cpp
#include <iostream>
#include "???"

#define EXIT_SUCCESS    0
#define LENGTH          10

int main (void) {
   Example* example = new Example(7.5);

   double dbl = 10;
   example->setValue(dbl);
   std::cout << example->getValue() << std::endl;

   return EXIT_SUCCESS;
}
```

2. Write a C++ program that, for each of the following types:
   - `int`
   - `char`
   - `std::string`
   - Array of `double`'s
   - Array of *pointers* to `float`'s

   Does the following:
   (a) Creates a `typedef` for the type - your choice of name
   (b) Allocates memory for a variable of that type on the *Program Call Stack*, using the typedef
   (c) Initialises the variable's value, to sensible a value of your choice
   (d) Prints out the contents of the variable

3. Using the same types listed in Question 2 above, write a C++ program that:
   (a) Creates memory on the *Heap* for each variable. (Remeber to use pointers as necessary).
   (b) Initialises the values of the variables
   (c) Prints out the contents of each variable
   (d) Ensures all memory is correctly de-allocated (deleted/freed) at the end of the program.

4. Write a function:

```cpp
void doubleArray(int values[], int length);
```

   That is given an array of integers, and multiplies every value by 2.
   The write a simple C++ program (with a main function) that creates an array of 10 integers, on the *Program Call Stack*, calls this function, and then prints out the values of the array.
   *(Hint: Work from previous labs should help you with this.).*

5. Change the function prototype in Question 5 above to be:

```cpp
void doubleArray(int* values, int length);
```

   What other changes did you have to make to your program so that it compiles and runs? ***Explain*** your answer to your tutor.

6. We are going to start combining some of the code you have written for the Red7 over the past few weeks. Complete the following activities:
   (a) Move the *declaration* of the `Card` class from Week 3 to a header file `Card.h`
   (b) Move the *implementation* of the `Card` class from Week 3 to a header file `Card.cpp`
   (c) *Modify* your code for Week 2 question 5 that reads in an integer and *instead* create a `Card` object (using the class) on the *Program Heap*. Then it should use the *methods* of the `Card` class to print out information about the card, such as the colour, number and rule.
   (d) *Extend* your reading program to create and array of 5 Card objects on the *Heap*.
   (e) Read in 5 card objects, storing them in the array, and print out the information.

7. *(This question will help you with Assignment 1!).* The following simple program creates a 2D array of characters, with 4 rows and 5 columns.

```cpp
#include <iostream>

#define ROWS   4
#define COLS   5
```

```cpp

int main (void) {
   char maze[ROWS][COLS] = {};

   std::cout << maze[0][0] << std::endl;

   return EXIT_SUCCESS;
}
```

We are going to use this 2D array to store a *maze*, such as the one below. Each character is one 'location' in the maze.

```
=====
=..<=
=G=.=
=====
```

We will use `(x,y)` co-ordinates to refer to each location in the maze. The top-left corner is `(0,0)`, and the bottom-right is `(5,4)`.

(a) Using the `(x,y)` co-ordinate system, what is the co-ordinate of the `<` and `G` symbols?

(b) Recall that a 2D array is specified by *rows* first then by columns. This means to "lookup" a location we first look-up the `y` co-ordinate in the 2D array, then the `x` co-ordinate. Thus, write the line of code look-up where the location of the `S` and `E` would be in the 2D array.

(c) Implement the function:

```cpp
void readMaze(char maze[ROWS][COLS]);
```

This function reads from standard input a 4x5 grid of characters and stores them in the 2D array.

8. *(Challenge)* Take your code from Week 2, question 4 that reads in an array of characters from the user and prints out the result. Do the following:
   - Modify the type of the array to:

```cpp
char* characters;
```

   - Allocate memory for this array on the *Program Heap*
   - Read in character from the user until EOF (or you run out of space in the array)