



**Laboratory Manual  
for  
CE/CZ1003  
Introduction to Computational Thinking**

**Practical Exercise #5:  
Data Abstraction  
(Data Structure)**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING  
NANYANG TECHNOLOGICAL UNIVERSITY**

## Ex. #5 – Data Abstraction

### Learning Objectives

The manual provides information and practical exercises to let you learn how data abstraction can be used in program to support and simplify the implementation of an algorithm.

### Intended Learning Outcomes

At the end of this exercise, you should be able to

- make use of appropriate data structure to improve the structure of a program.
- apply various concepts learnt in earlier topics (conditional selection, looping) and various functions to develop more complex programs.

### Equipment and accessories required

- Raspberry Pi 3 Model B (RPi3) board with Sense HAT add-on display module/board.
- A USB power source to power the RPi3 board (E.g. Power Bank, Adaptor or USB port of a desktop computer).
- A computer (desktop PC or notebook) with Ethernet port and cable for remote access of RPi3. Software (open source) to be installed on the computer – PuTTY, VNC Viewer and WinSCP

## 1. Data Structure

A data structure is an organization of data in such a way to allow more efficient manipulation of the data by a program. For instance, you have been using integer and string in your programming exercises, which are basic (known as primitive in Python) data structure that contain single value of a data, consisting of a sequence of numbers and characters respectively. With data arranged in certain structure, you can then apply various operations to these data structure; such as arithmetic operations for integers and string manipulation (e.g. `isdigit()` function).

In this exercise, you will learn how to use two data structures that store a collection of values (instead of single value) to simplify your program organization. They are **Tuple** and **List** (which are known as non-primitive data structure in Python).

For example, you can use (and had been using) Tuple data structure to specify the R-G-B element of various colours in terms of numbers when displaying message on the Sense Hat module, such as the following:

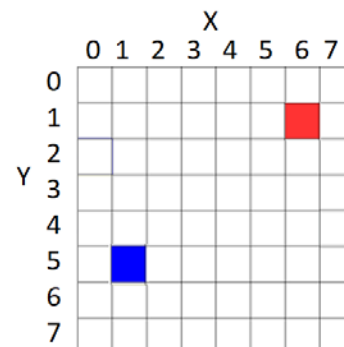
```
red = (255,0,0)
yellow = (255,255,0)
```

You can further combine it with the List data structure to display image on the Sense Hat display, which you will learn in this exercise.

## 2. Pixel display on Sense Hat board

The Sense Hat board's display is organized using an 8x8 LED matrix, where each LED can be used as one pixel of an image and addressed by using the x-y coordinate system. The numbering of both axes begins at 0 in the top left-hand corner as shown here. The red pixel is hence at coordinate (6, 1), while the blue pixel is at coordinate (?, ?). You can then set the colour of the individual pixel by using the `set_pixel()` function as follows:

```
set_pixel(6, 1, red)
```



## Ex. #5 – Data Abstraction

### Coding Exercise 5a

- Code a program that makes use of the appropriate Tuple data structure to light up the LED on the four corners of the Sense Hat display with four different colours.
- Change your program such that it displays the 4 colours at random positions on the Sense Hat display.

The following functions will be useful to implement these features:

- a) The `sleep()` function, which can be imported through the `time` module. For example `sleep(1)` will provide a 1 second delay in your program.
- b) The `randint()` function, which can be imported through the `random` module. For example `randint(1,10)` will produce a random integer with value between 1 and 10 (inclusive).

### 3. Image display on Sense Hat board

To display an image, you can set multiple pixels individually using multiple (64) `set_pixel()` function calls, but this will obviously be very tedious to code, and prone to error. Instead, you can use the **List** data structure to construct an image, and then display the whole image using one function call. A List data structure consists of multiple elements, and is defined by using the square brackets `[e1, e2,...]`. (Compare this with Tuple which uses parentheses, i.e. the round bracket `(e1, e2, ..)` ).

The following code snippet shows how you can use the List data structure, whose elements in turn consists of Tuples, to define an image and has it displayed on the Sense Hat board by calling the `sense.set_pixels()` function.

```
y = (255, 255, 0)
b = (0, 0, 0)
image_pixels = [b, b, b, b, b, b, b, b,
                 b, b, b, y, b, b, b, b,
                 b, b, y, y, y, b, b, b,
                 b, y, b, y, b, y, b, b,
                 b, b, b, y, b, b, b, b,
                 b, b, b, y, b, b, b, b,
                 b, b, b, y, b, b, b, b,
                 b, b, b, b, b, b, b, g]

set_pixels(image_pixels)
```

### Coding Exercise 5b

- Code a program that includes the above code, run it and observe the image that is coded in the List based data structure.
- Change the image to something of your fancy.
- Modify your program such that the image will automatically switch between two colours (e.g. yellow and green colour) at 1 second interval.
- Modify your program such that the image will also rotate in 4 random orientations with the image displayed in alternate colours.

### Coding Exercise 5c(Optional)

- Modify your program such that the image can switch between random colours.