Laboratory Manual
for
CE/CZ1003
Introduction to Computational Thinking

Practical Exercise #3:
Basic Python Programming

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

Ex. #3 – Basic Python Programming

Learning Objectives

The manual provide information and practical exercises to let you learn how to write simple Python programs through hands-on coding exercises on the Raspberry Pi. It also shows how variables and data types can be useful when coding program.

Intended Learning Outcomes

At the end of this exercise, you should

- know the basic structure of a Python program.

- be able to make use of variables and different data types in the program.

Equipment and accessories required

i) Raspberry Pi 3 Model B (RPi3) board with Sense HAT add-on display module/board.
ii) A USB power source to power the RPi3 board (E.g. Power Bank, Adaptor or USB port of a desktop computer).
iii) 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, 'MS Remote Desktop Connection' and WinSCP

_____

1. Basic Python Program

A program is basically a sequence of instructions that get executed in the order that it is coded. Writing (or coding) a program is to create the sequence of instructions according to your design and comply with the rules of the chosen programming language, which is Python in this course. A good programming style is to code the program such that it easy to be read and understood by other people.

The following is a simple program that should be intuitive obvious on what it is doing.

```
Line
 1    import math

 2    radius_str = input("Enter the radius of your circle: ")
 3    radius_int = int(radius_str)

 4    circumference = 2 * math.pi * radius_int
 5    area = math.pi * (radius_int ** 2)

 6    print ("The cirumference is:",circumference,  \
 7           ", and the area is:",area)
```

Ways to edit and run the exercises: (Choose either option 1 or Option 2)

1. Use **cloud9 IDE**
   Connect RPi to PC, run "**Putty**" and start the Cloud9 IDE (refer to **Practical Exercise manual #**2), and use PC web browser to access cloud9 IDLE. Check that cloud9 IDE is configured for Python 3 – See Lab #2 section 4.4 for the procedure.

   Or

2. Use "**Remote Desktop Connection**"
   Connect RPi to PC, run "Remote Desktop Connection" on PC desktop to connect to Rpi, and run the Python 3 IDLE within Rpi directly. *Refer to Appendix* for steps.

Ex. #3 – Basic Python Programming

Coding Exercise 3a

- Key in the above program code. Remember to save your program file in the folder of your group
- Run the program and observe its output (which is shown in ….)

'Analysis' of the program code

Line 1 in the code is to import a mathematic module, **math** that contains some commonly used code (in the form of functions) and constants for solving mathematical problems.
- A module is a file that contains program code written by someone else which you can 're-use' by importing it into your program using the keyword **import**.
- This is an example of the Abstraction concept used in the Computational thinking, which is to hide the underlying detail of the code from the user.

(Side note: It turns out that this **math** module is written in C programming language)

Line 2 is a python program assignment statement that involve a *variable*, a *function* and the assignment *symbol*.
- On the right side of the assignment symbol '=', **input()** is a built-in small Python program known as a function.
- It prints the characters indicated in the parentheses and then waits for you to input character(s) through the keyboard.
- Once you type the sequence of character(s) through the keyboard and press the Enter key, the function stores the sequence of characters somewhere in the memory, and assign this sequence of character to the **variable radius_str**, which is on the left side of the '=' symbol.
- The sequence of characters is known as a *string* in computing. That is the data type of this variable is string (**str** n short).
- Hence the variable **radius** is appended with **_str** to make it explicitly clear to the reader that this variable is a string.

Line 3 is another python program assignment.
- **int** is a built-in function that reads the value of variable **radius_str** and returns the integer value corresponding to the variable (What happen if you enter something that cannot be convert to integer? Try this by running your program,)
- The statement then assigns the integer value to the variable **radius_int.**
- Again it is a good practice to make it clear to the reader that this variable is of data type integer by appending it with **_int**.

Line 4 is a statement to calculate the circumference of a circle using the formula as shown, and assign it to the variable **circumference**
- The value of the pi is available through the math module as **math.pi**.

Line 5 is a statement to calculate the area of a circle as shown and assign it to the variable **area.**

Line 6 is to use the built-in function **print** to display the quoted string, follows by the value of the variable **circumference** onto the monitor.
- The backslash character '\' indicates that the statement is to continue onto the next line, for situation when the statement is too long to fix on one line.

Line 7 is the continuation of the print statement.

Ex. #3 – Basic Python Programming

2 Displaying messages on Sense HAT Add-on Board

Now that you have learnt the basic structure of a program, you will apply the knowledge to code a program to display and manipulate the text shown on the Sense HAT (SH) add-on board on the RPi using functions provided by the Sense Hat module (also refer to as software library).

2.1 The first thing to do when coding the program for the SH board is to import the module that contains the various functions required to operate the SH board. This is done by using the following two statements.

```
from sense_hat import SenseHat
sense = SenseHat()
```
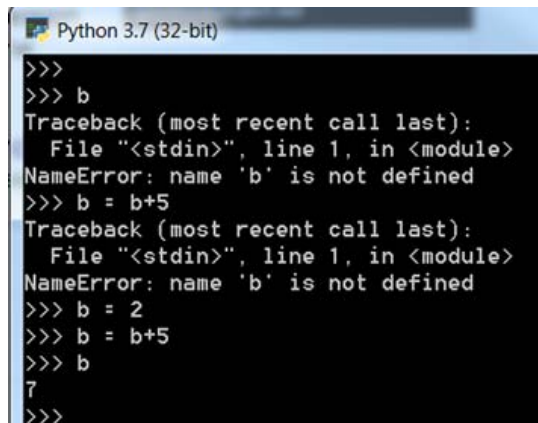
This is similar (but not the same) to create a variable and assign it with certain initial value - but the **sense** here is known as an object in Python. (And of course you can use other name of your choice.)

Side Note (for those who are curious - the following provide a bit more detail about what the two lines of code are about):
Slightly different from the math module you saw earlier, here you first import a 'class' definition **SenseHat** from the **sense_hat** module. A 'class' contains code that describe the functions (more correctly, it is known as methods in Python) that can be used by our program.

The way to use the class definition is you first create an object and initialize it to have the characteristics of the class. In this case, we create an object with the name **sense** and assign it to have the **SenseHat** class. This is described as to instantiate **sense** to be a **SenseHat** object.

This is somewhat similar (but not exactly the same) to when you create a variable b , you can't use it before you assign a value to it. (see below example that was entered in IDEL3 to illustrate the point)

2.2 With the object **sense** just created, you can now use it with the various commands (i.e. functions, or methods) provided by the **sense_hat** module. The following are some of these commands that you can use to manipulate the display on the SH board.

- To display a message on the SH board, you can use the command **show_message()** as follows (which you had done several time in earlier exercises):
```
sense.show_message("Hello")
```

Ex. #3 – Basic Python Programming

- You can use the command **set_rotation()** to change the orientation of the display shown on the SH board using one of the parameters (0, 90, 180, 270).

      sense.set_rotation(180)

- You can change how the message is displayed by adding extra *parameters* to the **show_message** command.
    - **text_colour** to change the colour of the message, such as
      sense.show_message("Hello", text_colour = (255,255,0))

    - **back_colour** to change the background colour of the message
      sense.show_message("Hello", back_colour = (255,0,0))

    - **scroll_speed** to change the speed of scrolling the message (default value is 0.1)
      sense.show_message("Hello", scroll_speed = 0.5)

- You can use the command **sense.clear()** to clear the display on the SH board, or set it to a specific colour.

      sense.clear() or sense.clear(100,100,100)

Coding Exercises 3b:
a) Basic Program Structure
- Write a program to display the message "This is fun!" on the SH module.
- Change the orientation of the message by 180 degree.
- Add the **text_colour** parameter to you message as shown in example above.
- Change the 3 numbers used in the **text_colour** to various values and observe its effect.
- Things to ponder (and ask your friend sitting beside you):
    - What do the 3 numbers mean?
    - What is the maximum value you can use for each of the number, and what does it imply?
- Change the background of the message such that the message is in yellow colour with the back ground in blue colour.
- Change the speed of the display using the **scroll_speed** parameter.

b) Use of variables
Now create three variables, **color_msg, color_bg, speed** in your program such that you can display messages using the statement as follows.

      **sense.show_message("I got it!", text_colour = color_msg, \
                              back_colour = color_bg, \
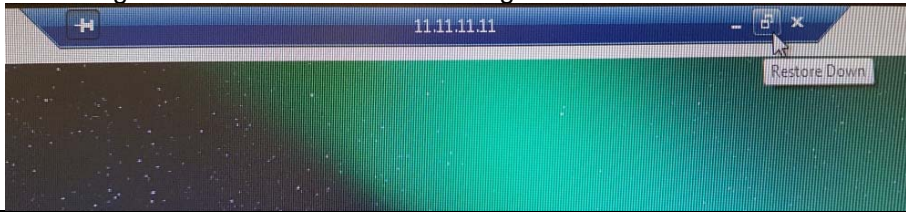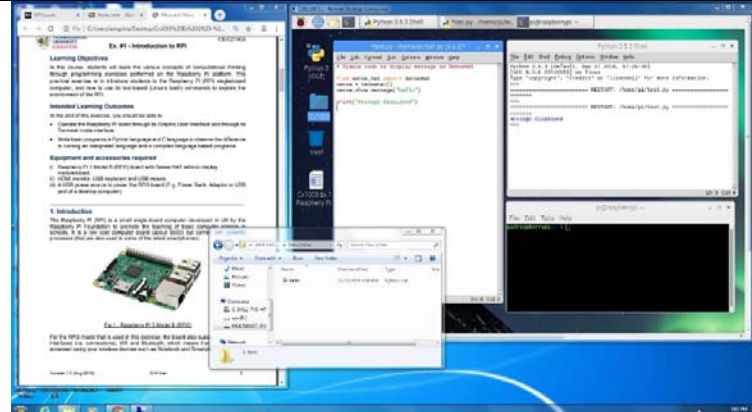   scroll_speed = speed)**

c) Challenge - Use of data types
Add the **input()** function to your program to allow the user to choose different colours and speed for your message display.

Tip: You may want to use the built-in **print()** function to print messages in the console when debugging your code.
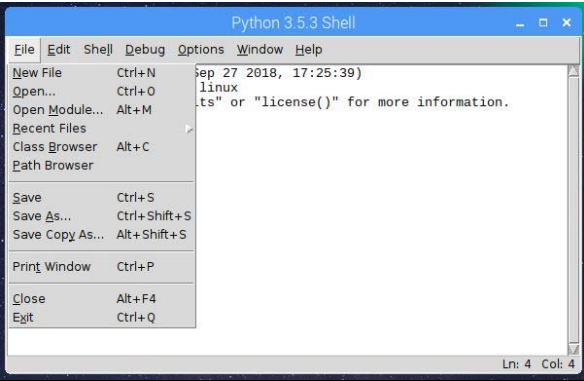
Ex. #3 – Basic Python Programming

Appendix – Using "Remote Desktop Connection" to connect to Rpi

1	Launch "Remote Desktop Connection" from PC desktop	
2	Use IP address **11.11.11.11** Click "Yes" on the Identity pop up window	
3	Use the following to login to Pi : Session : Xorg Username : pi Password : raspberry	
4	Click on the Resize Button on the Top blue bar to resize the remote desktop for ease of accessing the PC and remote session together	
5	You could use "right click" copy and paste between PC and the raspberry remote desktop session for easy file transfer and even copy/paste codes Remarks : right click copy/paste will not work on folder	

Ex. #3 – Basic Python Programming

To use Rpi Python IDLE within Remote Desktop session

1	Launch "Python3 (IDLE)" from Raspberry Pi desktop Alternatively you could launch from menu bar, Programming, Python 3(IDLE)	
2	Click "File" "New File" to open the Python IDLE editor 	
3	To execute the code, select "Run" "Run Module F5" from the **editor menu bar**, this will invoke a new python shell window where you can interact with the game. (Remarks : you can use the <F5> function key to execute the code as well) Keep both the editor and python shell side by side for ease of use 	