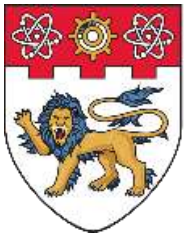

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Laboratory Manual
for
CE/CZ1103
Introduction to Computational Thinking and Programming

Practical Exercise #3:
Basic Python Programming -
Conditional Executions and Iterations

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

Ex. #3 – Basic Python Programming

Learning Objectives

The manual provides information and practical exercises to let you learn how to apply the relational operations, selections and repetition to implement robust programs through hands-on coding exercises on the Raspberry Pi.

Intended Learning Outcomes

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

- apply the concept of relational operations with selection to form conditional expressions.
- make use of iteration in your program to perform repetitive tasks.

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 and WinSCP

---

1. Conditional Execution

For a program to be useful and practical, it is essential that the program has the ability to check conditions and alter its behaviour accordingly. The simplest way to perform this conditional execution is to make use of the **`if`** statement, which has the general form of

```
if <boolean expression>:
    statement
^^^^   :
    statement
```

Note that the body of the compound statement that get executed when the condition (i.e. the boolean expression) is true must be indented by the same amount, and it is recommended to use 4 spaces for indenting in Python.

In practice, it is more common to use the **`if-else`** and **`if-elif-else`** statements. (Refers to LAMS module 3.2 for detail and examples on how they can be used.)

2. Iteration

Repeated execution of a set of statements is known as iteration. Iteration is used extensively in practical programs to repeat similar tasks, either for a specified number of times or until a condition is met.

There are two methods to perform iterations in Python: using the **`while`** statement and using the **`for`** statement. (Refers to LAMS module 3.3 for detail and examples on how they can be used.)

Coding Exercise 3a

Turn on your RPi board. Either using remote desktop connection to connect to Rpi or start the Cloud9 IDE. Make sure that the Cloud9 IDE is configured for Python 3 (See Lab #2 Appendix A for detail) if you are using Cloud9.

- Code a Python 3 program to display a message (e.g. "This is fun!") on the Sense Hat board with the following specifications (**similar to the one you developed in Exercise 3b earlier**).
  - The program will prompt the user to choose the colour of the message, and the scroll speed of the message.

Coding Exercise 3b: Error checking (runtime error)

Ask your team member sitting beside/nearby you to run your program (on your RPi) to test the robustness of your program, while you also test your team member's program on his/her RPi (e.g. swap your seat).

- The sample test cases are as follows:

  1) Enter "fast" while asking float speed value
  2) Enter the following value while getting colour values:

     Enter the value of the red color for message: 355
     Enter the value of the green color for message: 567
     Enter the value of the blue color for message: 789
     Enter the value of the red color for background: 345
     Enter the value of the green color for background: 564
     Enter the value of the blue color for background: 678

- If your program crashed during execution, ask your team member what he/she has done to cause it to malfunction. In computing, this is known as having "**bug(s**)" in your program.

- Study your code carefully, and think about how you will resolve the problem, i.e. how to debug your program.

- Sketch/Draw a flowchart of your refined design

- Implement the new design in your program.

- Once completed, ask your team member to test you program again (and vice versa) until you are satisfied that your program can perform properly under all conditions.

- Compare each other approach used to solve the problem, and discuss which one you think is the 'better' solution.

Challenge - Coding Exercise 3c Further Error checking (logic error)

Add the following feature to your program

- The program will repeat (i.e. ask for different colour) until the user issues a command to quit.

- Add the function to also allow the user to choose the colour of the background.
  - What will happen if the user choose the same colour for both the message and the background. How would you resolve this in your design to let the user try again and again until the user enter correct input?
  - How can you control the maximum trial for the same error to be 3 times? That is, after the user choose the same colour for both the message and the background for three times, display the message that the user has exceeded the number of trials, then end the program.