Practical 5. Variables, Booleans, and loops

Rob Young

IBI1, 2019/20

1 Learning objectives

- Explain how variables work within a piece of code
- Explain the concept of loops
- Understand and use Booleans
- Plan a project using pseudocode
- Create and use variables in Python
- Create and use loops and Booleans in Python

2 Working with Python

- Python is installed directly on the ZJE server. Log into this as in previous weeks.
- You can start Python in interactive mode by typing 'python' and the command prompt should change to >>>.
- See how it works by typing the following commands:

```
a = "hello "
b = "world!"
print(a+b)
```

- What does the output look like?
- You can leave the interactive session by typing exit() at the command prompt.
- A python script is a collection of commands saved to a file. You can write several lines of code in one file called a script. Before you run the script for the first type, you will have to save it to a .py file. You can then run this script containing all the command by typing 'python name_of_script.py" (make sure to exit interactive mode before running this). If there is a result, it will be shown in the console.
- Try it out: Use nano or your favourite text editor to type the commands above into a file called helloworld.py and then run it. What does the output look like?
- Python files are not that different from text files. In particular, they can be put under version control using git and GitHub. For more complex projects in particular, make sure you commit changes on a regular basis, so you can go back to earlier versions if you run into problems.

3 Working with variables

• Create a new 'Practical5' directory within your GitHub repository folder next to 'Practical4'. Store your scripts from this week's practical within that directory.

- Start a new file called variables.py

3.1 Some simple math

- Store your birthday as a six-digit number in a variable called a. For example, Rob's birthday is 19 July 1984 so his number would be 190784. Store Rob's birthday in a variable called b.
- Now store today's date as another six digit number into a variable called c.
- Calculate the absolute difference between your birthday value and today's data. Store this in a variable called d.
- Calculate the absolute difference between your birthday value and Rob's value. Stored this in a variable called e.
- Compare d to e. Which of them is greater?

3.2 Booleans

- You learned in lecture that "either X or Y" is the same as "(X and not Y) or (Y and not X)". Make Boolean variables X and Y. Make a variable Z that encodes "(X and not Y) or (Y and not X)" and verify that it true if either X or Y (but not both) are true.

- Make a variable W that encodes Zhiwen's more elegant solution ("X !=Y") and verify that W and Z are always the same, no matter the values of X and Y.

4 Fibonacci sequence

- Start a new file called fibonacci.py
- In the Fibonacci sequence, each number is the sum of the previous two numbers. The first two places in the sequence are 1, 1. Fibonacci calculated this sequence up to its 13 place. Can you work out what the 13^{th} value of this Fibonacci sequence is?
- Write a script that starts with two positive integers 1 and 1 and computes and displays the first 13 values of the Fibonacci sequence (Python may decide to display some figure(s) after the comma, e.g. "25.0" instead of "25". This is nothing to worry about at this stage.) Before you write the actual code, plan your project using pseudocode and comments (lines starting with #). When you write the actual code, leave the pseudocode comments in.

5 Mystery Code

- Download the file mystery_code.py from this week's Learn page. Transfer it into your own directory so that you can push it to your GitHub repo.
- Look at file mystery_code.py
- What does it do? Run it a few times and formulate a hypothesis.
- Check whether your hypothesis is correct by going through the code line by line. Sometimes, it's helpful to do this using pen and paper, but it is up to you. If you find out what a particular line does, use comments (lines starting with #) to note your thoughts. We have done this at the beginning of the document, e.g. in lines 4-6 to explain line 7.

- Once you have confirmed what the code does, write a one-sentence description next to # Answer: on line 2.

6 R rate

- Start a new file called r_rate.py
- The rate of reproduction of a virus, e.g. COVID-19, can be quantified using the 'r' number which is the average number of individuals infected by each individual with the virus. An r rate of 1.2 means that each infected individual will on average infect 1.2 individuals.
- Write some code that starts with the number of IBI1 students (n = 84) and calculates the number of infected individuals after 5 rounds of infection for a given r rate. Please define the r rate as a separate variable so that we can check your code works.
- The output should be a sentence describing the r rate and the total number of individuals infected after 5 generations. Before you write the actual code, plan your project using pseudocode and comments (lines starting with #). When you write the actual code, leave the pseudocode comments in.
- You may find the command str() helpful: It converts a number into a string. For instance, str(5) will give "5"

7 For your portfolio

The markers will look for and assess the following:

File variables.py

- The marker will look at your variables d and e, confirm that e has been created in the way that was specified in the instructions, and compare d to e.
- The marker will test all possible values of *X* and *Y* and verify that *W* and *Z* behave as they should.

File collatz.py

- The marker will check that the code runs, terminates, and that it does indeed display the correct 13 values for the Fibonacci sequence.
- The marker will verify that you have used pseudocode to plan and comment your project.

File mysterycode.py

The marker will look at the answer and check whether it is correct.

File r_rate.py

- The marker will verify that you have used pseudocode to plan and comment your project
- The marker will test your script using the number r = 1.1
- Partial grades will be given if the project is incomplete.

You can add or edit things after the Practical session. We do not look at the commit date, we just want it all to be there!