Practical 4. Variables, Booleans and Loops

Rob Young, Caitlin McNiff, Zhengyang Yuan

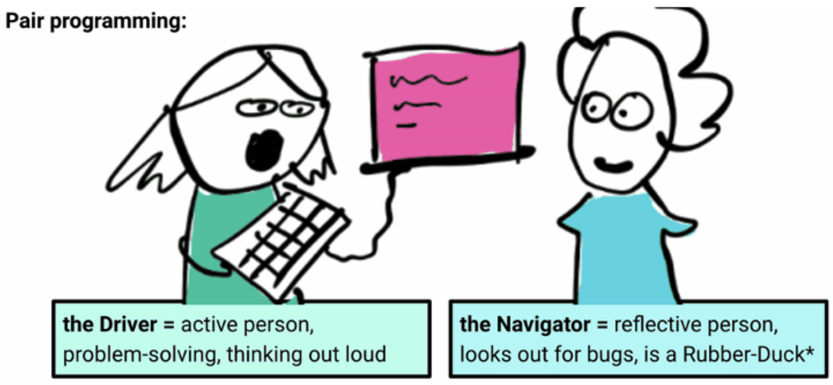
IBI1, 2024/25

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 Paired programming

In this week's practical, we are going to work in pairs. When you arrive in the class you will be paired (randomly) with another member of the class. One of you will type at the keyboard and act as the 'Driver' while the other will ask questions and help out with bugs, they are the 'Navigator'. After 15 mins or so, we will swap partners so that you both get a chance to be the driver and the navigator.


Pair programming:
the Driver = active person, problem-solving, thinking out loud
the Navigator = reflective person, looks out for bugs, is a Rubber-Duck*

If you have completed the work for this week's portfolio, please still come to the practical session. You can explain your work and receive instant feedback from your partner as well as the instructors.

You will be paired with a different partner for each paired programming session this semester.

3 Working with Python

- You can start Python in interactive mode by typing ‘python’ in a Terminal window 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: 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.

4 Working with variables

- Create a new ‘Practical4’ directory within your GitHub repository folder next to ‘Practical3’. Store your scripts from this week’s practical within that directory.
- Start a new file called variables.py

4.1 Some simple math

- An office work is trying to decide how to commute to their office. They can either walk to the bus stop and take the bus directly to their office, or drive to a nearby car park and then walk the final stage. Which is quicker?
- The walk to the bus stop is 15 mins. Store this time in a variable called a.
- The bus journey takes 1 hr and 15 mins. Store this time in a variable called b.
- Store the total length of time for this commute in a third variable called c.
- The drive takes 1 hr and 30 mins while the walk from the car park takes 5 mins. Store these two times in variables d and e, respectively, and store the total length of time for the car-based commute in a variable called f.
- Compare c to e. Which of them is longer? Which method of transport is quicker? Record your answer to this question as a comment in your script.

4.2 Booleans

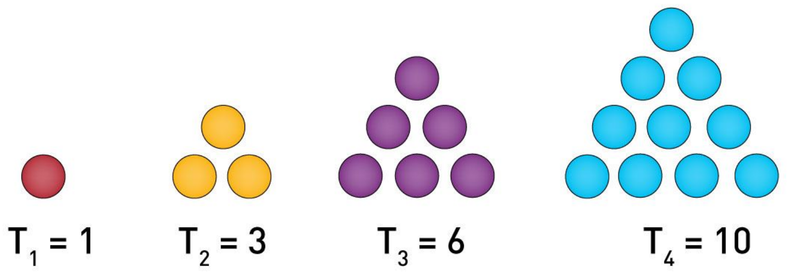
- Create two variables X and Y which can be used to store Boolean variables. To begin with, variable X should be True and variable Y should be False.

- Now create a new variable called W which encodes the Boolean variable 'both X and Y'.

- What does the truth table for W look like? Store your answer to this question as a comment in your script.

5 Triangle sequence

- Start a new file called triangle.py
- In the Triangle sequence, each number records the number of dots required to form an equilateral triangle. The nth triangle number is the number of dots or balls in a triangle with n dots on each side. It can be calculated as the sum of the n numbers from 1 to n. For example, the third number is 1 + 2 + 3 = 6. Pictorially, the first four triangular numbers can be represented as below:



- Write a script that calculates the triangular sequence and displays the first ten values (Python may decide to display some figure(s) after the comma, e.g. "10.0" instead of "10". 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.

6 Mystery Code

- Download the file mystery_code.py from this week's Learn page. Transfer it into your own directory for this week's practical 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.

7 BMI Calculator

- BMI stands for Body Mass Index and is a simple measure used to help clinicians decide whether a person has a healthy body weight, given their height.
- It can be calculated using the following equation: $BMI = \frac{Weight\ (kg)}{Height\ (m)^2}$. A BMI greater than 30 indicates that a person is obese, while underweight individuals have a measure less than 18.5. People in the intermediate range are considered to have a normal weight.
- Start a new file called bmi_calculator.py
- Write some code that stores the value of a person's weight (in kg), and height (in m).
- Calculate their BMI and decide on which of the above three categories they can be placed into.
- The output should include a sentence stating the person's BMI and as whether they should be considered underweight, normal weight, or obese.
- 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"

8 For your portfolio

The markers will look for and assess the following:

File variables.py

- The marker will look at your variables c and f, confirm that they were created in the way that was specified in the instructions, and compare c to f.
- The marker will verify that you have included a comment describing which method of commuting is shortest.
- The marker will test the values of W for different combinations of X and Y and verify that your truth table matches these values.

File triangle.py

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

File mystery_code.py

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

File bmi_calculator.py

- The marker will verify that you have used pseudocode to plan and comment your project
- The marker will test that the output BMI measure and weight category matches the values for the input variables supplied.

- 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!