

Introduction to Programming

Python Lab 8: Loops

Getting Started

- Create a new folder in your disk space with the name **PythonLab8**
- Launch the Python Integrated Development Environment (IDLE) - begin with the **Start** icon in the lower left corner of the screen.
- If you are in a DCSIS laboratory, search using the keyword **Python** and click on **IDLE (Python 3.6 64-bit)** .

A window with the title **Python 3.6.2 Shell** should appear. This window is the **Shell**.

Getting Started (2)

- If you are in the ITS laboratory MAL 109, then right mouse click on the **Start** icon in the lower left corner of the screen.

A list of menu options should appear and click on ***Search***. Type ***Python*** in the search text box at the bottom of the pop-up window. A list of Apps should appear and select

Python 3.4 IDLE(PythonGUI)

A window with the title **Python 3.4.3 Shell** should appear. This window is the ***Shell***.

- In the ***Shell*** click on **File**. A drop down menu will appear. Click on **New File**. A window with the `title` **Untitled** should appear. This window is the ***Editor***.

Getting Started (3)

- In the Editor, click on **File**, and then in the drop down menu click on **Save As...** .

A window showing a list of folders should appear.

- To search any folder on the list, double click on the folder.
- Find the folder **PythonLab8** and double click on it.
- In the box **File name** at the bottom of the window
 1. Type `Vowels.py`
 2. Then click on the button **Save** in the lower right corner of the window.

The title of the Editor should change to show the location of the file `Vowels.py`.

Objectives of the exercises set

- **Objectives**

- Use `for` statements to implement count-controlled loops that iterate over a range of integer values or the contents of any container.

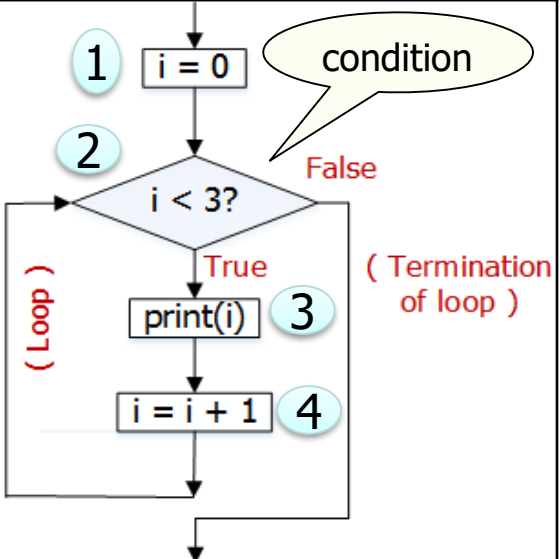
Syntax	Example	Explanation															
<p>for variable in container: statements #loop body</p> <p>↑ <i>All the statements in the block (loop body) have the same level of indentation.</i></p> <p><i>The header ends in a colon —</i></p>	<pre>stateName = "Ohio" for letter in stateName: print(letter) # loop body</pre> <p><i>The variable letter takes each of the values 'o', 'h', 'i', 'o' in turn for each iteration.</i></p> <table border="1" data-bbox="788 982 1362 1359"> <thead> <tr> <th>iteration</th> <th>letter</th> <th>Output</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>O</td> <td>O</td> </tr> <tr> <td>2</td> <td>h</td> <td>h</td> </tr> <tr> <td>3</td> <td>i</td> <td>i</td> </tr> <tr> <td>4</td> <td>o</td> <td>o</td> </tr> </tbody> </table>	iteration	letter	Output	1	O	O	2	h	h	3	i	i	4	o	o	<p>The string "Ohio" is stored in the variable <code>stateName</code>.</p> <p>The loop body is executed for each successive character of the string in <code>stateName</code>, starting with the first character.</p>
iteration	letter	Output															
1	O	O															
2	h	h															
3	i	i															
4	o	o															

Objectives of the exercises set (2)

- **Objectives**

- Use `while` statements to implement event-controlled loops.

A `while` loop executes instructions repeatedly while a condition is true.

Syntax	Example	Flow chart
<pre>while condition : statements</pre> <p>↑ All the statements in the block (loop body) have the same indentation.</p> <p>↑ The header ends in a colon.</p>	<pre>i = 0 1 while i < 3 : 2 print(i) 3 i = i + 1 4</pre> <p>Output 0 1 2</p>	



In the example, the variable `i` is initialised outside the while loop (statement 1) and updated in the loop body (statement 4).

Objectives of the exercises set (3)

- The table below shows the working of the `while` loop example shown on page 6.

i	i < 3 ?	Output using print(i)	i = i + 1
0	True	0	1
1	True	1	2
2	True	2	3
3	False – end of the <code>while</code> loop		

Program Vowels.py: Vowels

- **Question 2: Problem statement**

Consider the following `for` loop,

```
stateName = "Ohio"  
for letter in stateName :  
    print(letter)
```

The variable `letter` takes each of the values "O", "h", "i", "o" in turn.

The output of the code is

```
O  
h  
i  
o
```


Program Vowels.py: Vowels (2)

- **Problem statement** (continued)
 - i. Write a program that requests a string from the keyboard and then prints out the characters in the string in a vertical line, as in the above example. (see page 8)
 - ii. Add to your program code to count the number of lower case vowels in the input string and then print out this number after printing out the vertical line of characters. The vowels are a, e, i, o, u.

Program Vowels.py: Vowels (3)

- **Problem solving** - Convert the following pseudo code into a sequence of Python statements in your program.

Input

1. Read in a string and store it in a variable named `s`.
2. Declare a variable named `n` and assign a value of 0 to it.

Process the input and display the output (steps 2 to 5).

The variable `n` is used to keep a count of the number of **lower case** vowels (a, e, i, o, u) in the input string `s`.

3. Add the following `for` statement to print out the characters in the string `s` in a vertical line, as shown in the output example on page 8.

```
for letter in s :  
    print(letter)           # loop body
```

Indent the print statement in the block (loop body).



Program Vowels.py: Vowels (4)

- **Problem solving** (continued)

4. Inside the `for` loop, add an `if` statement to check if the letter is equal to an "a", "e", "i", "o", **or** "u" after the `print` statement in step 3. Align the `print` and `if` statements at the same level of indentation.

If the combined Boolean expression is true, then add 1 to the current value of `n`. Use the `if` statement below to help you get started.

```
if letter == "a" or letter == "e" or complete the combined Boolean expression by adding test conditions for the remaining vowels (i,o,u) :
```

```
    # True branch - if the combined expression is true
```

```
    Indent & write an assignment statement to add 1 to n and store the result in n. Recall n keeps a count of the number of lower case vowels.
```

5. Outside the `for` loop, write the code to print out the number of lower case vowels. Align the `print` and `for` statements.

- Provide a comment at the beginning of the program to explain the purpose of the program together with your name and the date. Save the program to the file `Vowels.py` and then run it. 11

Note the indentation of the code.

Program NumberProperties.py: NumberProperties

- Create a new Editor for a new file called `NumberProperties.py`

- **Question 3: Problem statement**

Write a program that reads a list of strictly positive integers from the keyboard, one at a time.

Use a `while` loop to read these integers.

The end of the input is indicated by the integer **0**. Note that this 0 is a **sentinel** in that it marks the end of the input but does not belong to the set of integers whose properties are sought (PFE Section 4.3).

The program then prints out the following numbers.

- i. the average value
- ii. the smallest of the values
- iii. the largest of the values
- iv. the inclusive range, that is one more than the difference between the smallest and the largest values.

It can be assumed that the input contains at least one number other than 0. See PFE P4.5.

Program NumberProperties.py: NumberProperties (2)


- **Problem statement** (continued)

Include a comment with each number that is printed out, for example:

The average value is: ...

If you need to check how a `while` loop behaves, then test the following code.

```
nextNumber = 5
while not (nextNumber == 0) :      # test for entry to the loop
    print(nextNumber)              # in the loop do something with the data
    nextNumber = nextNumber-1      # get the next number
```



*All the statements
in the block (loop
body) have the
same indentation.*

Program NumberProperties.py: NumberProperties (3)

Input

A set of positive integer numbers.

For example, the numbers:

2 5 3 9

are typed in at the keyboard, one at a time.

Computations

1. average =
 $(2 + 5 + 3 + 9) / 4$
2. mn = min(2, 5, 3, 9)
3. mx = max(2, 5, 3, 9)
4. range = (mx - mn) + 1

Output

The average value is 4.75

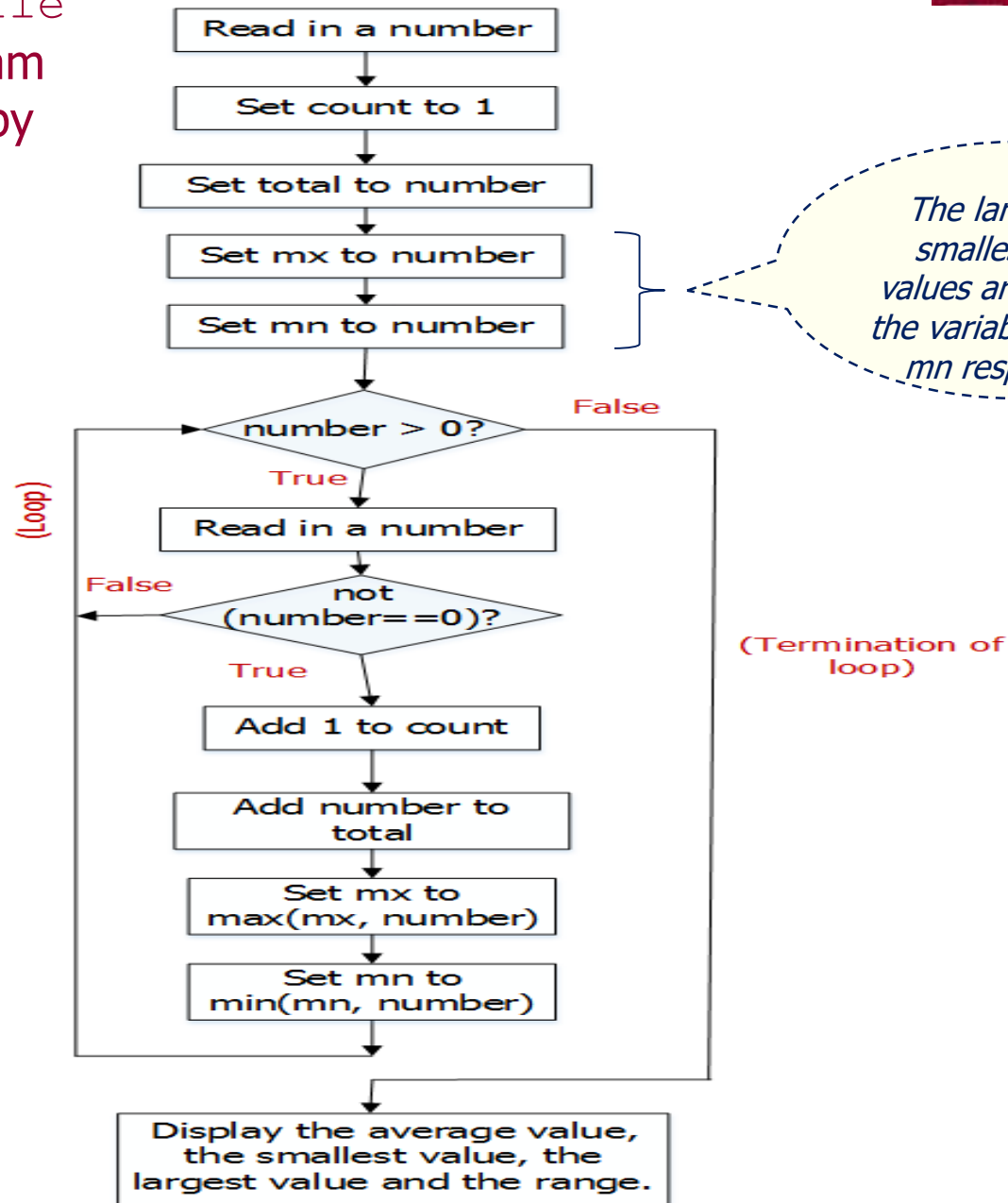
The smallest value is 2

The largest value is 9

The range is 8

Note: The Python built-in function max returns the largest value of the arguments; and the function min returns the smallest value of the arguments.

Flow chart of a while loop for the program NumberProperties.py (4)



The largest and smallest of the values are stored in the variables mx and mn respectively.

Program NumberProperties.py: NumberProperties (5)

- **Problem solving** - Convert the following pseudo code into a sequence of Python statements in your program.

Input

1. Read in an integer and store it in a variable named `number` *

*Process
the input
and
display all
required
results
(steps 2
to 5).*

2. Create a variable named `count` and initialise it with the value of 1. The variable keeps a count of the number of inputs.

3. Create variables, for example, `total` to store the total of the inputs, `mx` to store the largest value and `mn` to store the smallest value.

Set the value of each variable to be the value of the variable `number` initially.

An example is shown below for the variable `total`.

```
total = number
```

* Assume that the first numeric value read in from the keyboard is a positive integer greater than 0.

Program NumberProperties.py: NumberProperties (6)

- **Problem solving** (continued)

4. Write a `while` loop statement to read in more integers as long as the condition, `number > 0`, remains true. Below is an outline of the algorithm needed to solve the problem set inside the loop.

```
while number > 0 :  
    # loop body  
    # Add code below to read in a number  
    number = read in an integer similar to step 1  
    # Add an if statement below to check if the number is not  
    # equal to zero.  
    if (condition tests if the number is not equal to zero) :  
        # True branch of the if statement  
        # i) add 1 to count as shown, indented below.  
        count = count + 1  
        # Write more code inside the True branch to  
        # ii) add number to total and store result in total  
        # iii) Set mx to be the larger of the two values in  
        # the variables mx and number using the max function.  
        # iv) Set mn to be the smaller of the two values in  
        # the variables mn and number using the min function.
```

*Note the
indentation
of the
statements.*

Program NumberProperties.py: NumberProperties (7)

- **Problem solving** (continued)

5. Outside the `while` loop, write additional code to find the average value and the inclusive range. Then write code to print all required results separately together with suitable messages.

For example, the `print` statement below can be used to display the inclusive range.

```
print("The range is ", mx - mn + 1)
```

Align the `while` statement and the additional code for step 5, such as `print` statements.

- Provide a comment at the beginning of the program to explain the purpose of the program together with your name and the date.
- Save your program to the file `NumberProperties.py` and then run it.

Supplementary Questions for Private Study

- The laboratory worksheet contains supplementary questions in section 4 for private study.
- You are encouraged to complete the supplementary questions at home, or in the laboratory if you have time after completing questions 2 to 3.