Introduction to Programming

Python Lab 2:
Variables

PythonLab2 lecture slides.ppt

Ping Brennan (p.brennan@bbk.ac.uk)

8 October 2019 or
24 January 2020

Getting Started

- Create a new folder in your disk space with the name **PythonLab2**

- 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** 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 **PythonLab2** and double click on it.
  - In the box **File name** at the bottom of the window

    1. Type **Variables.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 Variables.py.

Question 2. Program to compute volumes

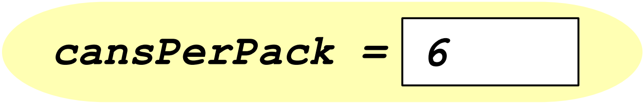
- Copy the following code, taken from Python for Everyone (PFE) Section 2.1.5, into the *Editor*.

```
##
# This program computes the volume (in litres) of a six-pack of soda
# cans and the total volume of a six-pack and a two-litre bottle.
#

# Litres in a 12-ounce can and a two-litre bottle.
CAN_VOLUME = 0.355
BOTTLE_VOLUME = 2.0

# Number of cans per pack.
cansPerPack = 6

# Calculate total volume in the cans.
totalVolume = cansPerPack * CAN_VOLUME
print("A six-pack of 12-ounce cans contains", totalVolume, "litres.")

# Calculate total volume in the cans and a 2-litre bottle.
totalVolume = totalVolume + BOTTLE_VOLUME
print("A six-pack and a two-litre bottle contain", totalVolume, "litres.")
```

Question 2. Program to compute volumes (2)

- Declaring variables in the Python program:
  - Statements such as

    `cansPerPack = 6`

    cansPerPack = 6

    assign values to variables.

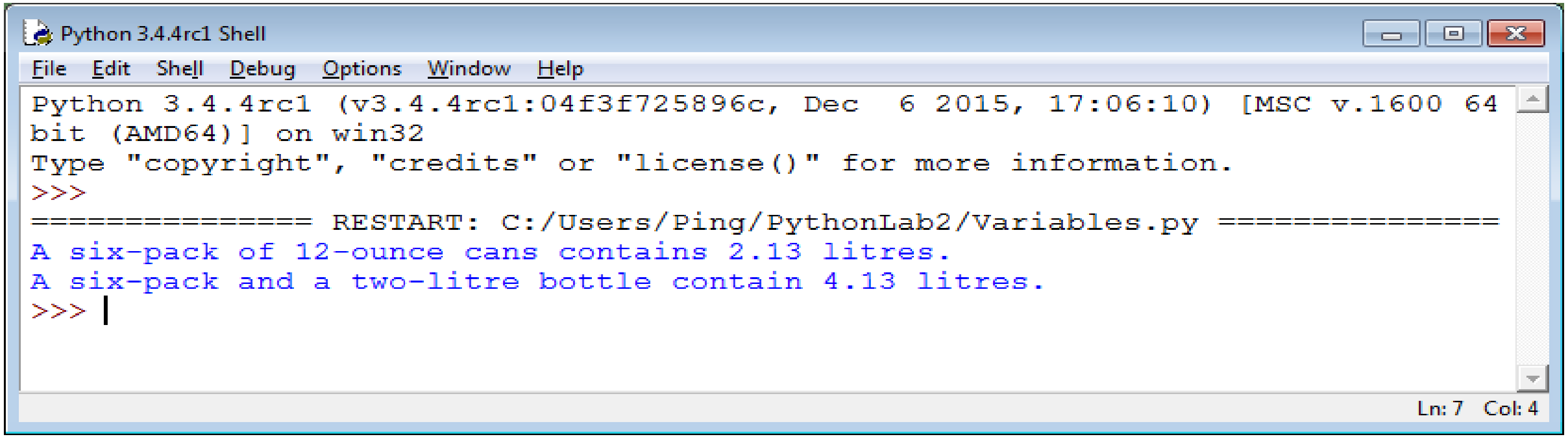
  - This is the first occurrence of the variable name `cansPerPack`.

    Thus the variable is created and assigned a value, in this case `6`. The type of the value is **int**.

  - The statement `CAN_VOLUME  = 0.355` creates the variable `CAN_VOLUME` and assigns it a floating-point value of `0.355`. The type of the value is **float** for any floating-point number that contains a fractional part.

Question 2. Program to compute volumes (3)

- In the *Editor* click on **Run**, then on the drop down menu click on **Run Module**. A window will appear with the message **Source Must Be Saved**. Click on **OK**. The program in the *Editor* is saved to the file Variables.py and then run to produce the text from the two print statements in the *Shell* window.



```
Python 3.4.4rc1 (v3.4.4rc1:04f3f725896c, Dec  6 2015, 17:06:10) [MSC v.1600 64
bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
=============== RESTART: C:/Users/Ping/PythonLab2/Variables.py ===============
A six-pack of 12-ounce cans contains 2.13 litres.
A six-pack and a two-litre bottle contain 4.13 litres.
>>> 
```

- Move the statement

```
cansPerPack = 6
```

to the end of the program and check the compiler error message. Then move the statement back to the correct place.

Extend the program in Question 2 to calculate purchase price

- By using **two new variables**, extend the program in Question 2 (page 5) to calculate the total purchase price for the number of bottles that are purchased.

- Problem solving

  1. Declare and initialize two new variables, `unitPrice` and `quantity` :

     `unitPrice` - contains the price of a single 2-litre bottle in dollars

     `quantity` - contains the number of bottles that are purchased

     Use reasonable initial values for the two variables, for example,

     ```
     quantity = 2
     ```

  2. Calculate and print the total purchase price for the bottles. Use the formula below as the argument for the function `print()` :

     ```
     unitPrice * quantity
     ```

- Convert the above solution into Python statements.

- Add comments in your code and update the comment at the top of the program. Then run your extended program.

Question 4. Program to print the value of `mystery`

- Review question R2.1 in PFE which contains the following statements:

```
mystery = 1                  # line 1
mystery = 1 - 2 * mystery    # line 2
mystery = mystery + 1        # line 3
```

- Create a new *Editor* for a new file called Mystery.py
- Copy the above code into the *Editor*.
- Problem solving
  1. Print the initial value of the variable `mystery` after line 1 in the above code.
  2. Print the final value of the variable `mystery` after line 3 in the above code.
- Convert the above solution into two Python statements that call the function `print()`.
- Add a comment at the top of the source file to explain the origin and purpose of the code. Then run your program.

Question 5. Program to calculate square root of 2

- Create a new *Editor* for a new file called Sqrt2.py . Type the following code into the *Editor*.

```
# Initialize x
x=1.0
print("x*x:", x*x)

# First iteration
delta = (1/x)-x/2
x = x+delta
print("x*x:" , x*x)


# Second iteration
delta = (1/x)-x/2
x = x+delta
print("x*x:", x*x)
```

- Add a comment at the top of the source file to describe the origin and purpose of the program.

Question 5. Program to calculate square root of 2 (2)

- Problem solving (extend the program on page 10)
  1. Add a third iteration to determine the new `delta` and `x` values, and then print the square of `x` (i.e. `x*x`)
  2. Add a fourth iteration to determine the new `delta` and `x` values, and then print the square of `x` (i.e. `x*x`)
- Convert the above solution into Python statements and add these to the end of the code on page 10.
- Run your program.
- Examine the numbers that are printed out.

Supplementary Questions for Private Study

- The laboratory worksheet contains supplementary questions 6.1, 6.2 and 6.3 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 5.

Appendix A Variable Names
(Python For Everyone, Section 2.1.3)

When you define a variable, you need to give it a name that describes its purpose. You must follow a few simple rules when naming something in Python:

- Names must start with a letter or the underscore (_) character, and the remaining characters must be letters, numbers, or underscores.
- Other symbols such as ? or % cannot be used, and spaces are not permitted inside names.
- Names are case sensitive in Python.
- Python reserved words cannot be used as names.

Recommended practice:

- Use descriptive names for variables.
- Begin names of variables with a lowercase letter.
- By convention, name of a variable that is all uppercase indicates a constant, that is the value of the variable does not change.