Fundamentals of Python Programming

Getting Started with Python Programming

- Early 1990s: Guido van Rossum
  - invented the Python programming language
- **Python** is a high-level, general-purpose programming language for solving problems on modern computer systems
- Useful resources at **www.python.org**

Running Code in the Interactive Shell (1 of 2)

- Python is an **interpreted** language
- Simple Python expressions and statements can be run in the **shell**
  - Easiest way to open a Python shell is to launch the IDLE
  - To quit, select the window's close box or press Control+D
  - Shell is useful for:
    - Experimenting with short expressions or statements
    - Consulting the documentation

Running Code in the Interactive Shell (2 of 2)

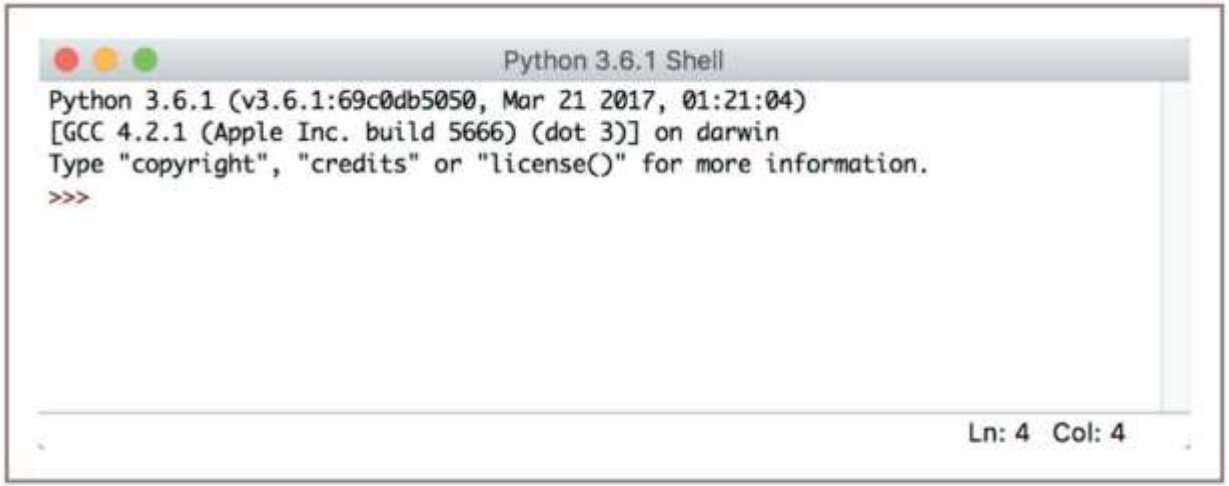

Figure 1-6 Python shell window

Input, Processing, and Output (1 of 5)

- Programs usually accept inputs from a source, process them, and output results to a destination
  - In terminal-based interactive programs, these are the keyboard and terminal display
- In Python, inputs are Python expressions or statements
  - Outputs are the results displayed in the shell
- Programmers can also force output of a value by using the print function
  - **print (<expression>)**
- Example:

```
>>>print ("Hi there")
Hi there
```

Input, Processing, and Output (2 of 5)

- The following example receives an input string from the user and saves it for further processing:

```
>>> name = input("Enter your name:")
Enter your name: Ken Lambert
>>> name
'Ken Lambert'
>>> print(name)
Ken Lambert
>>>
```

Input, Processing, and Output (3 of 5)

- The **input** function always builds a string from the user's keystrokes and returns it to the program
- Strings that represent numbers must be converted from strings to appropriate number types
  - Two type conversion functions: **int** (for integers) and **float** (for floating-point numbers)

Input, Processing, and Output (4 of 5)

- The next session inputs two integers and displays their sum:

```
>>> first = int(input("Enter the first number: "))
Enter the first number: 23
>>> second = int(input("Enter the second number:"))
Enter the second number: 44
>>> print("The sum is", first + second)
The sum is 67
```

Input, Processing, and Output (5 of 5)

Function	**What It Does**
float(<a *string of digits*>)	Converts a string of digits to a floating-point value.
int(<a string of digits>)	Converts a string of digits to an integer value.
input(<a *string prompt*>)	Displays the string prompt and waits for keyboard input. Returns the string of characters entered by the user.
print(<expression>, ...,<expression>)	Evaluates the expressions and displays them, separated by one space, in the console window.
<string 1> + <string 2>	Glues the two strings together and returns the result.

Editing, Saving, and Running a Script (1 of 3)

- We can then run Python program files or **scripts** within IDLE or from the O S's command prompt
  - Run within IDLE using menu option, F5 (Windows), or Control+F5 (Mac or Linux)
- Python program files use .**py** extension
- Running a script from IDLE allows you to construct some complex programs, test them, and save them in **program libraries** to reuse or share with others

Editing, Saving, and Running a Script (2 of 3)

Figure 1-7 Python script in an IDLE window

Editing, Saving, and Running a Script (3 of 3)

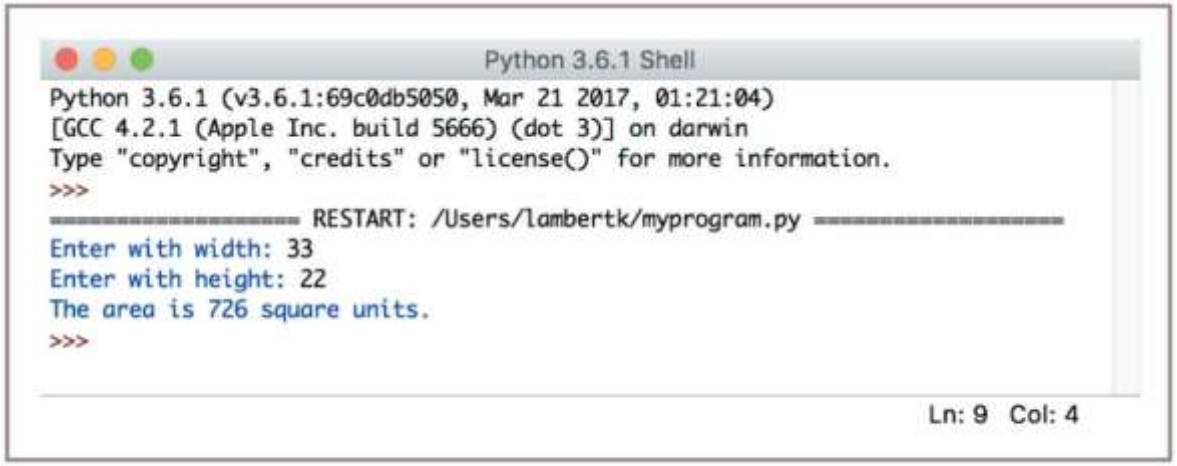

Figure 1-8 Interaction with a script in a shell window

Behind the Scenes: How Python Works

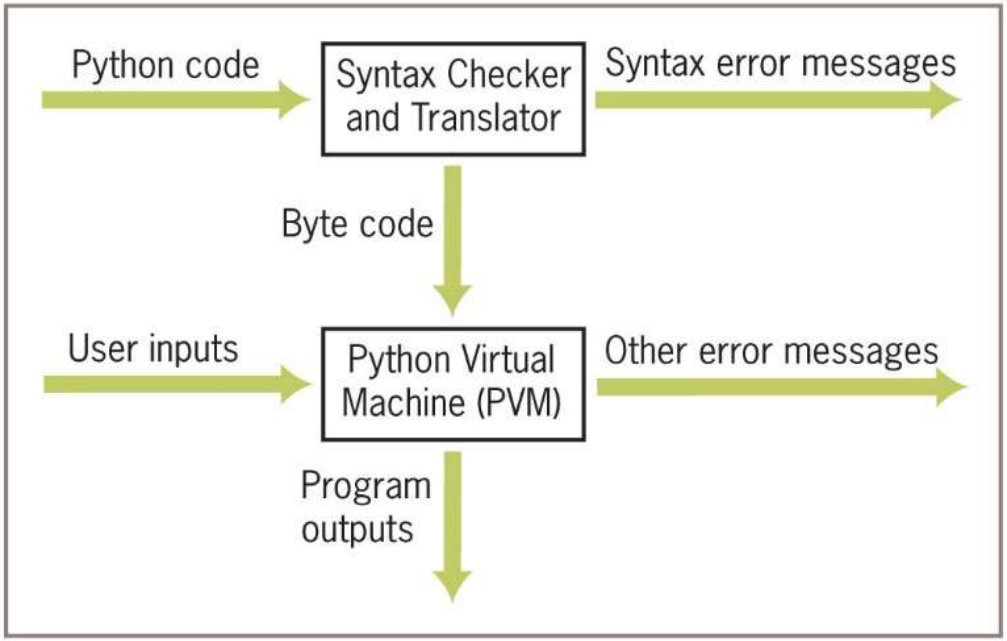

Figure 1-9 Steps in interpreting a Python program

Detecting and Correcting Syntax Errors (1 of 2)

- Programmers inevitably make typographical errors when editing programs, called **syntax errors**
  - The Python interpreter will usually detect these
- **Syntax:** rules for forming sentences in a language
- When Python encounters a syntax error in a program, it halts execution with an error message
- Example:

```
>>> length = int(input("Enter the length: "))
Enter the length: 44
>>> print(lenth)
Traceback (most recent call last):
File "<pyshell#l>", line 1, in <module>
NameError: name 'lenth' is not defined
```

Detecting and Correcting Syntax Errors (2 of 2)

- The next statement attempts to print the value of the correctly spelled variable:

```
>>> print(length)
SyntaxError: unexpected indent
```

- Final example, programmer attempts to add two numbers, but forgets to include the second one:

```
>>> 3 +
SyntaxError: invalid syntax
```