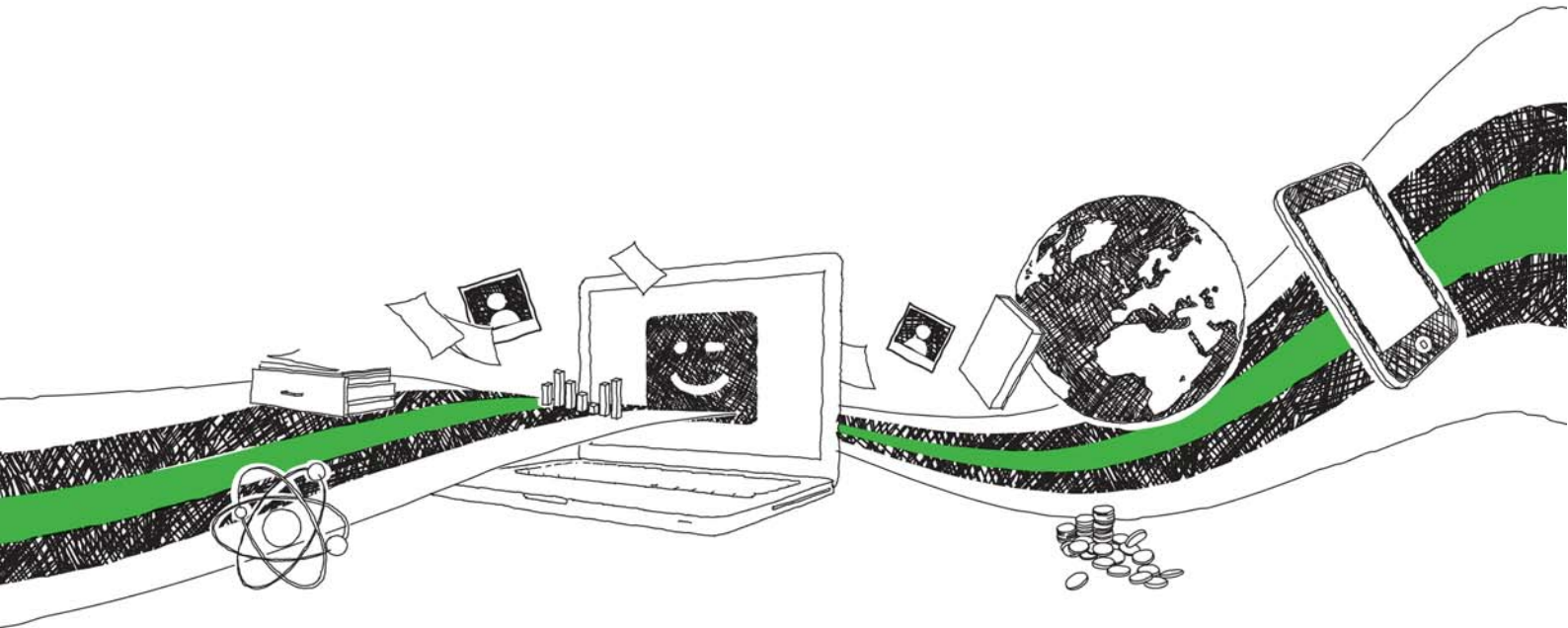


Palaniappan Sellappan

Learn Python. The Easy Way

Through Examples

YOUR KNOWLEDGE HAS VALUE



- We will publish your bachelor's and master's thesis, essays and papers
- Your own eBook and book - sold worldwide in all relevant shops
- Earn money with each sale

Upload your text at www.GRIN.com
and publish for free



Bibliographic information published by the German National Library:

The German National Library lists this publication in the National Bibliography; detailed bibliographic data are available on the Internet at <http://dnb.dnb.de> .

This book is copyright material and must not be copied, reproduced, transferred, distributed, leased, licensed or publicly performed or used in any way except as specifically permitted in writing by the publishers, as allowed under the terms and conditions under which it was purchased or as strictly permitted by applicable copyright law. Any unauthorized distribution or use of this text may be a direct infringement of the author's and publisher's rights and those responsible may be liable in law accordingly.

Imprint:

Copyright © 2018 GRIN Verlag
ISBN: 9783668771123

This book at GRIN:

<https://www.grin.com/document/435037>

Palaniappan Sellappan

Learn Python. The Easy Way

Through Examples

GRIN - Your knowledge has value

Since its foundation in 1998, GRIN has specialized in publishing academic texts by students, college teachers and other academics as e-book and printed book. The website www.grin.com is an ideal platform for presenting term papers, final papers, scientific essays, dissertations and specialist books.

Visit us on the internet:

<http://www.grin.com/>

<http://www.facebook.com/grincom>

http://www.twitter.com/grin_com

LEARN PYTHON

THE EASY WAY - THROUGH EXAMPLES

PROF. DR. P. SELLAPPAN

MALAYSIA UNIVERSITY
of Science and Technology

Preface

Python, developed by Guido van Rossum of Netherlands in the late 80s, and named after the BBC TV show Monty Python's Flying Circus, is one of the most user-friendly and powerful general-purpose computer programming languages available today.

Its English-like syntax makes it a great language for teaching and learning computer programming. Python's powerful data structures/types such as lists, tuples, dictionaries, sets, and arrays make coding simple. It also comes with an extensive collection of built-in/library functions that allows users to develop software applications with relative ease. Besides, users can freely import external modules to help them develop all sorts of applications.

Python's interactive and interpreted mode makes coding and testing software easy. Python also doubles us as a powerful and sophisticated calculator.

You can use Python to develop all sorts of applications ranging from simple Mathematical and Text processing to Database, Web, Graphical User Interface, Network, Games, Data Mining, Artificial Intelligence, Machine Learning and Deep Learning.

This book is intended for beginners who have little or no knowledge of programming. It is also suitable for intermediate programmers who already have some knowledge of programming.

This text is suitable for secondary school, college and university students irrespective of their field of study – be it Arts, Business, Science, Engineering, Life Sciences or Medicine. It starts with the basics, but progresses rapidly to the advanced topics such as lists, tuples, dictionaries, arrays, functions, classes, files and databases. So whether you are a beginner or an intermediate programmer, this book will help you master the essentials of Python programming very quickly.

The book is written in a simple, easy-to-read style and contains numerous examples to illustrate the programming concepts presented. It also contains exercises to test the reader's grasp of the material presented in each chapter.

Acknowledgements

I would like to gratefully acknowledge the contributions of several people who have in one or another assisted me in the preparation of this book. I would like to thank all my IT students and colleagues for their valuable input and feedback in the preparation of this manuscript.

My grateful thanks also go to Professor Dr. Premkumar Rajagopal, President of the Malaysia University of Science and Technology for giving me the opportunity, freedom, encouragement and support that I needed in the preparation of this manuscript. I would like to especially thank him for creating and nurturing an environment that actively promotes learning, research, teamwork and personal development. His dynamic leadership is greatly appreciated.

Last but first I would like to thank God for giving me the desire, motivation, interest, passion, strength and guidance to successfully complete this manuscript.

Dr. P. Sellappan
Professor of Information Technology
Dean of School of Science and Engineering
Provost of Malaysia University of Science and Technology

About the Author

Dr. P. Sellappan is currently Professor of Information Technology, Dean of School of Science and Engineering, and Provost of the Malaysia University of Science and Technology. Prior to joining Malaysia University of Science and Technology, he held a similar academic position in the Faculty of Computer Science and Information Technology, University of Malaya, Malaysia.

He holds a Bachelor in Economics degree with a Statistics major from the University of Malaya, a Master in Computer Science from the University of London (UK), and a PhD in Interdisciplinary Information Science from the University of Pittsburgh (USA).

Working in the academia for more than 30 years, he has taught a wide range of courses both at undergraduate and postgraduate levels: Principles of Programming, Advanced Programming, Programming Languages, Data Structures and Algorithms, System Analysis and Design, Software Engineering, Human Computer Interaction, Database Systems, Data Mining, Health Informatics, Web Applications, E-Commerce, Operating Systems, Management Information Systems, Research Methods, Mathematics and Statistics.

Professor Sellappan is an active researcher. He has received several national research grants from the Ministry of Science and Technology and Innovation under E-Science and FRGS to undertake IT-related research projects. Arising from these projects, he has published numerous research papers in reputable international journals and conference proceedings. Besides, he has also authored over a dozen college- and university-level IT text books.

As a thesis supervisor, he has supervised more than 70 Master and PhD theses. He also serves in editorial/review boards of several international journals and conferences. He is also chief editor of the Journal of Advanced Applied Sciences and the Plain Truth magazine. He is a certified trainer, external examiner, moderator and program assessor for IT programs for several local and international universities.

Together with other international experts, he has also served as an IT Consultant for several local and international agencies such as the Asian Development Bank, the United Nations Development Program, the World Bank, and the Government of Malaysia. His professional affiliation includes membership in the Chartered Engineering Council (UK), the British Computer Society (UK), the Institute of Statisticians (UK), and the Malaysian National Computer Confederation (MNCC).

Contents

Chapter 1	About Python.....	1
	1.1 Python Features.....	1
	1.2 Python Environment.....	2
	1.3 Installing Python on Windows.....	2
	1.4 Running Python Code.....	3
	1.5 Sample Runs.....	4
	1.6 Parts of a Python Program.....	5
	Exercise.....	8
Chapter 2	Python Basics.....	9
	2.1 Identifiers.....	9
	2.2 Keywords.....	10
	2.3 Variables.....	10
	2.4 Comments.....	11
	2.5 Quotes.....	11
	2.6 Blank Lines and Indentation.....	12
	2.7 Multiline Statements.....	12
	2.8 Operators.....	13
	2.9 Expressions.....	17
	2.10 Assignment Statements.....	17
	2.11 Multiple Assignments in Single Statements.....	17
	2.12 Multiple Statements on Single Lines.....	18
	2.13 Code Blocks.....	18
	2.14 Data Types.....	18
	2.15 Input-Output Statements.....	22
	2.16 Importing Modules.....	23
	Exercise.....	23
Chapter 3	Control Structures.....	26
	3.1 Sequence	26
	3.2 Decision Making.....	26
	3.3 Loops.....	31
	3.4 Other Control Flows.....	39
	3.5 Sample Programs.....	41
	Exercise.....	46
Chapter 4	Advanced Data Types.....	48
	4.1 Lists.....	48
	4.2 Tuples.....	50
	4.3 Dictionaries.....	50
	4.4 Sets.....	52
	4.5 Arrays.....	53
	Exercise.....	63
Chapter 5	Advanced Input/Output.....	65
	5.1 Reading Data from Keyboard.....	65
	5.2 Sending Output to Monitor.....	67
	5.3 Formatting Output.....	69
	5.4 Reading Input from Text File.....	76
	5.5 Writing Output to Text File.....	80
	5.6 Reading from and Writing to Text Files.....	81

5.7	Sending Output to Printer.....	82
	Exercise.....	83
Chapter 6	Built-in Functions.....	84
6.1	What are Functions?.....	84
6.2	Built-in Functions.....	84
6.3	Math Functions.....	86
6.4	String Functions.....	87
6.5	Data Conversion Functions.....	89
6.6	Input/Output Functions.....	90
6.7	Date Functions.....	91
6.8	List Functions.....	93
6.9	Tuple Functions.....	95
6.10	Dictionary Functions.....	95
6.11	Set Functions.....	96
6.12	Sample Programs.....	97
	Exercise.....	102
Chapter 7	User-defined Functions.....	104
7.1	Why User-defined Functions.....	104
7.2	Defining Functions.....	104
7.3	Calling Functions.....	105
7.4	Call by Reference vs. Value.....	107
7.5	Function Overloading.....	110
7.6	Scope of Variables.....	111
7.7	Recursive Functions.....	112
7.8	Anonymous Functions.....	115
7.9	Sample Programs.....	116
	Exercise.....	120
Chapter 8	Classes.....	121
8.1	Object-oriented Concepts.....	122
8.2	Defining Classes.....	125
8.3	Class Inheritance.....	125
8.4	Abstract Classes.....	130
8.5	Overloading and Overriding Methods.....	131
8.6	Operator Overloading.....	134
8.7	Object Interaction.....	135
8.8	Sample programs.....	136
	Exercise.....	144
Chapter 9	Threading.....	146
9.1	What are Threads?.....	146
9.2	Importing Thread Modules.....	146
9.3	Thread Methods.....	147
9.4	Creating Threads With Functions.....	147
9.5	Creating Threads With Classes.....	153
9.6	Synchronizing Threads.....	156
9.7	Accessing Shared Resources.....	158
9.8	Daemon vs. Non-Daemon Threads.....	160
9.9	Enumerating Threads.....	162
9.10	Sample Programs.....	164
	Exercise.....	166
Chapter 10	Database.....	168
10.1	What is a Database?.....	168
10.2	Database Concepts.....	168

10.3	SQL Commands.....	169
10.4	Creating Database.....	171
10.5	Inserting Records.....	172
10.6	Fetching Records.....	174
10.7	Displaying Records in Different Formats.....	176
10.8	Inserting Records from Lists.....	176
10.9	Deleting Records.....	177
10.10	Updating Records.....	179
10.11	Aggregate Functions.....	180
10.12	Multiple Databases.....	182
10.13	MySQL Database.....	186
	Exercise.....	187
References		189

Chapter 1

About Python

Learning Outcomes:

After completing this chapter, the student will be able to

- *Describe the main features of Python.*
- *Explain the Python environment.*
- *Run Python in calculator, interpreted, and script mode.*
- *Describe the different parts of a Python program.*

1.1 Python Features

Python was developed by Guido van Rossum of Netherlands in the late 80s. It was named after the BBC TV show [Monty Python's Flying Circus](#).

Python is one of the popular general-purpose programming languages available today. Its English-like syntax makes it easier to learn. It is a great language for learning and teaching computer programming.

With Python you can develop all kinds of applications ranging from simple Business, Mathematical and Text processing to Database, Web, GUI (Graphical User Interface), Network, Games, Data Mining and Machine Learning applications.

Python has rich features which includes the following:

- Python can be used as a powerful and sophisticated calculator.
 - Python is interactive – you can type your code at the Command prompt and view the results immediately.
 - Python has an interpreter that lets you execute code as you type. You don't have to compile the whole program to run it. This is very useful for testing and debugging code.
 - Python is object-oriented – you can define classes by bundling data attributes (variables) and functionality (methods) and create objects and work with them.
- 1) It has extensive libraries with built-in functions/methods that greatly simplify programming.
 - 2) It is portable – you can run Python code on several hardware and software platforms (Linux and Windows).
 - 3) It provides interfaces to major database software like MS Access, SQL Server, SQL Lite, MySQL and Oracle.
 - 4) It supports Graphical User Interface (GUI).

- 5) It provides advanced data structures/types like lists, tuples, dictionaries and sets that make programming easy.
- 6) It supports automatic garbage collection to reclaim and reuse unused memory previously allocated to objects that no longer exist.
- 7) It can be used as a scripting language – you can write programs and store them in files for later use or embedding them in other programs.

1.2 Python Environment

You can run Python on several platforms like Windows and Linux. You can also incorporate Python code in programming languages like C, C++, Java and R.

You can execute Python code in three modes:

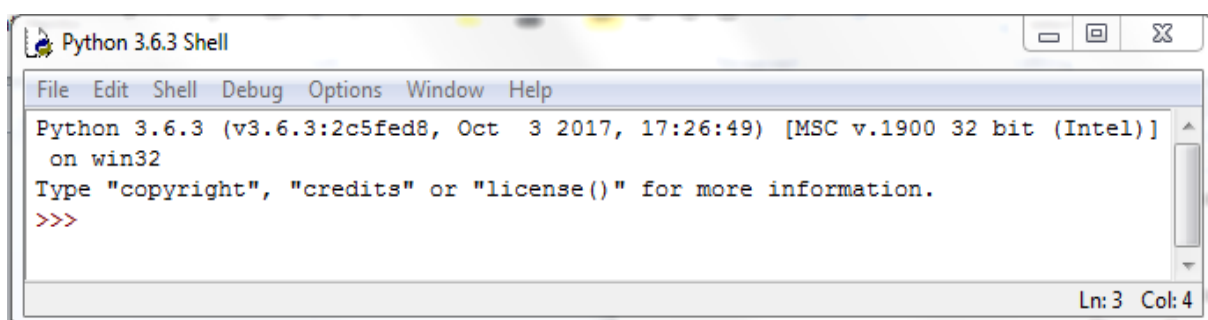
- **Calculator mode** – execute one statement or line of code at a time. It will automatically remember the previous results stored in the memory (provided they are not overwritten).
- **Interpreted mode** - you can execute statements and view the results immediately. You don't have to have the full program to execute it. This feature is very useful for testing and debugging code.
- **Script mode** – after you have debugged a program, you can store it in a file and later execute it as a script. This will improve performance, especially if it is a large or computationally-intensive program.

Thus Python provides a user-friendly environment for both novice, intermediate, and expert programmers.

1.3 Installing Python on Windows

This text is based on Python 3.6.x. for Windows. You can download and install Python as follows:

- Open a Web browser and go to <https://www.anaconda.com/download/>.
- Click on Download Python 3.6.x. for Windows.
- Run the downloaded file by accepting all the default settings.
- Run Python and you will see the Command window like the one shown below.



1.4 Running Python Code

You can run Python in three modes as follows:

▪ Interactive Interpreter

Enter `python` at the command line as follows:

```
C:> python
```

You can include options at the Command prompt such as

```
-d    provides debug output  
file  runs script from file
```

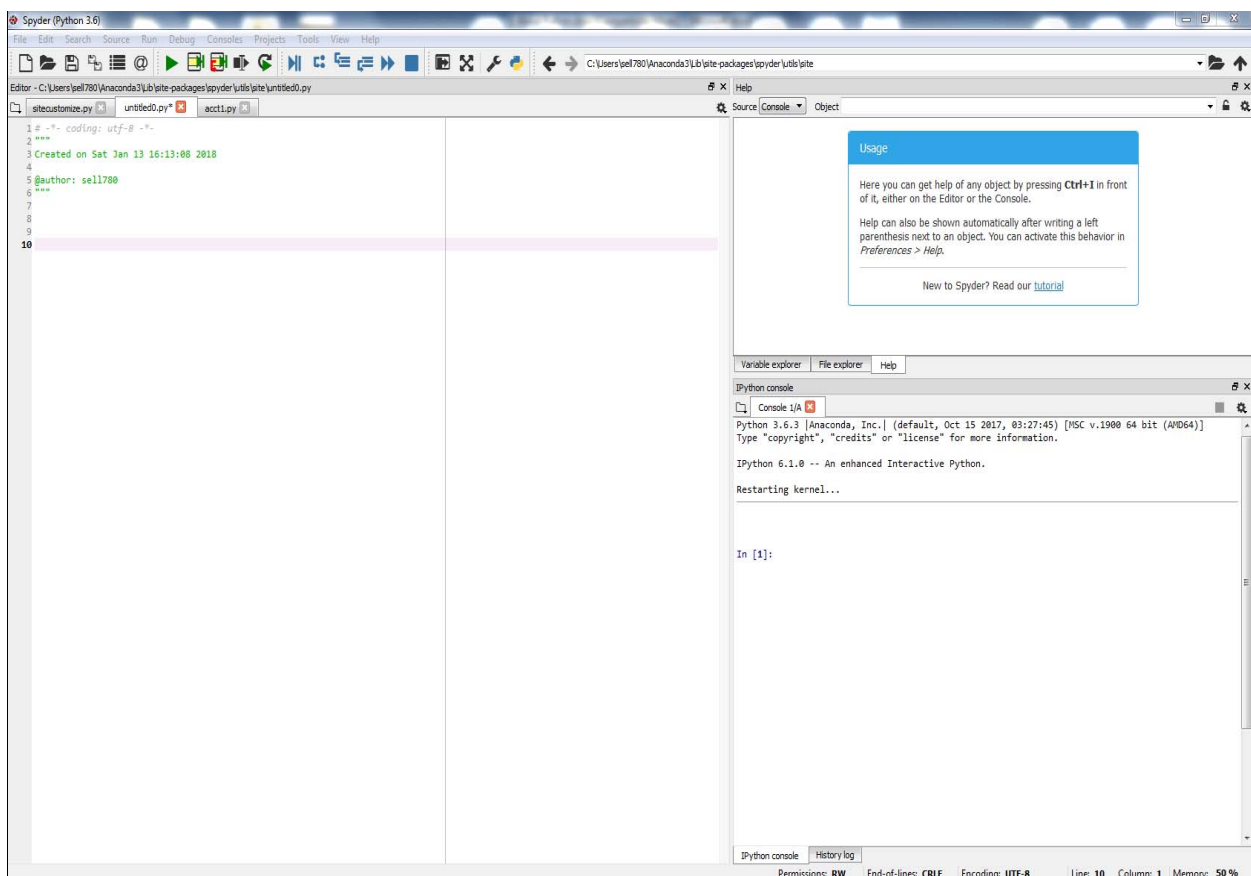
▪ Script at Command-line

You can execute a Python script at the Command line as follows:

```
C:> python script.py
```

▪ Integrated Development Environment

You can also run Python in the Integrated Development Environment (IDE) that looks like the one shown below:



The IDE provides several windows to help you code, see output/results and consult related documentation if you need them. It also provides several menus and menu icons (Save, Run, Debug, Stop Debugging, etc.) at the top to help you write, test and debug code.

1.5 Sample Runs

Let's now run some simple code in the different modes.

Calculator Mode

At the `>>>` prompt, type the code below and press Enter.

```
>>> print ('Hello world!')
```

You will get the following output:

```
Hello world!
```

Here are more examples with their output.

```
>>> print ('John ' + 'Mark') # join the two names
John Mark

>>> 5 + 7 * 7
54

>>> 15/3 + 5 * 3 - 12
8.0

>>> math.sqrt(25) # take the square root of 25
5.0

>>> math.sqrt(25+24)
7.0


>>> print (5 + math.sqrt(49))
12.0

>>> v = ['Wong', 'Sally', 'Ali', 'Sam'] # a list of names
>>> print (v)
['Wong', 'Sally', 'Ali', 'Sam']

>>> t = ('red', 'yellow', 'green') # a tuple of colors
>>> print (t)
('red', 'yellow', 'green')

>>> d = {'Sam': 111, 'Sally': 222} # a dictionary of paired items
>>> print (d)
{'Sam': 111, 'Sally': 222}
```

Interpreted Mode

Type the following code and run by clicking the arrow button on the menu bar (.

```
import math
name = 'Sally'
age = 25
print ('\nYour name is {} and you are {} years old.'.format(name, age))
```



```
x = 5
y = 44
print ('\nSquare root of x+y = ', math.sqrt(x+y))
```

You will get the following output:

```
Your name is Sally and you are 25 years old.
```

```
Square root of x+y = 7.0
```

Script Mode

Write a Python script (like the above) and store it in the default directory as `test.py`. (Python files have extension `.py`).

You can now run the stored script by typing

```
$ python test.py
```

The script will produce the same output as above.

1.6 Parts of a Python Program

A Python program has many parts/sections (see the below sample program): comments, expressions, statements, input, output, functions, import, etc. We will explain the different parts of a Python program using a simple sample program given below.

Note: The line numberings on the left, generated by the system, is for reference purpose only – they are not part of the code. Also, for easy reading, Python uses different colors for different parts of a program: green for comments, blue for keywords, pink for functions, etc.

Comments

Inserting comments in a program makes the code more readable. You can insert comments on a single line, on several lines, or inline after a statement.

A single line comment starts with a sharp character (`#`) as in lines 7, 10.... in the sample code. Everything on that after `#` is treated as a comment. You can create a *multiline* comment by starting *each* line with a `#`.

An *inline* comment starts with a `#` *after* a statement as in lines 12, 17.... in the sample code.

You can also insert a multiline comment by starting and ending the comment with triple apostrophes (`' '`). Such a comment can span several lines as in lines 2 to 6.

Line spacing

Inserting blank lines to separate code blocks makes a program more readable as in lines 9, 14....

Indentation

Code blocks in Python are *indented* by a fixed number of spaces (typically 4 spaces) as in lines 12 and 13. All statements in the code block must follow the *same* indentation.

Sample Python code

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Jan 13 16:13:08 2018
4
5 @author: sell780
6 """
7 # import math functions from system library
8 import math
9
10 # define function to calculate area of a circle given its radius
11 def cir_area(radius):
12     area = math.pi * radius * radius # pi is from math library
13     return area
14
15 # input radius from keyboard, then convert the value to float
16 r = float(input('Enter radius: '))
17 a = cir_area(r) # call cir_area function with radius r
18 # print/display output
19 print ('\nArea of circle with radius {:.2f} = {:.2f}'.format(r, a))
20
21 # calculate area of triangle given its base and height
22 base = 5.8
23 height = 7.2
24 tri_area = 0.5 * base * height
25 print ('\nArea of triangle with base {:.2f} and height {:.2f} = {:.2f}'.\
26         format(base, height, tri_area))
27
28 name = ['Joe', 'May', 'Wong']
29 print ('\nName list:', name)
30
31 color = ('red', 'yellow', 'green')
32 print ('\nTraffic light colors: ', color)
33
34 tel = {'Joe': 1234567, 'May': 2345678, 'Wong': 3456789}
35 print ('\nTel. list: ', tel)
36
```

Sample output

```
Enter radius: 5.85

Area of circle with radius 5.85 = 107.51

Area of triangle with base 5.80 and height 7.20 = 20.88

Name list: ['Joe', 'May', 'Wong']

Traffic light colors: ('red', 'yellow', 'green')

Tel. list: {'Joe': 1234567, 'May': 2345678, 'Wong': 3456789}
```

Keywords

Keywords (also called reserved words) have special meaning in Python. You cannot use these keywords for any other purpose such as for naming identifiers (variables, functions or classes). For example, `import` (line 8), `define` (line 11) and `return` (line 13) in the code are keywords.

Case sensitivity

Python identifiers for naming variables, functions and classes are *case-sensitive*. It treats uppercase and lowercase letters as *different* characters. That means, `name`, `Name` and `NAME` are all different identifiers.

Parameters

Parameters (arguments) in functions (methods) are enclosed between a pair of curved brackets `()`. The brackets are still needed even if a function has no parameters.

Input statement

The `input` statement (function) is used for reading data from the Keyboard (or Console). It takes the form

```
variable = input ('prompt')
```

where `variable` stores the data entered, and `prompt` prompts the user to enter the data.

Output statement

The `print` function is used for sending output (results) to the monitor (Console). It takes the form

```
print (output list)
```

where `output list` is a list of variables or expressions that you want to print or display.

Assignment statement

An assignment statement takes the form

```
variable = expression
```

where `variable` stores the result of the `expression`, and `=` is an assignment operator (not an equality operator). In an assignment statement, the expression on the right-hand side of `=` is first evaluated and the result is assigned to the `variable` on the left-hand side of `=`. The `expression` can be a number, a constant, a literal, a variable, an algebraic expression, or a function or method call.

Import statement

The `import` statement is used to import modules (classes) stored in external files. It begins with the keyword `import` followed by the name of the module (e.g. `math`) as in line 8.

Lists, tuples and dictionaries

Python variables include *grouped data*: lists, tuples and dictionaries. Lists are enclosed between square brackets `[]` as in line 28; tuples are enclosed between curved brackets `()` as in line 31; and dictionaries (containing paired items) are enclosed between braces or curly brackets `{ }` as in line 34.

Note: We will discuss all these and topics in greater detail in the rest of the chapters. So don't worry if you haven't fully grasped the material presented in this chapter. The sample code is meant to give you a flavor of how a Python program looks like. As you progress through the chapters, you will learn how to write Python programs.

Exercise

1. Run Python in calculator mode to evaluate the following:
 - a) $23 - (4 * 15) / 3 + 73$
 - b) Concatenate (join) the strings "Good" and "day"
 - c) Concatenate the strings "time", "and", "chance"
2. Run Python in calculator mode to evaluate the following:
 - a) Square of 547.25
 - b) Square root of 973.16
 - c) Square root of $(973.75 + \text{square of } (45))$
3. Run Python in interpreted mode to execute the following code:

```
x = (55 + 25) * 2
y = 546.95 + sqrt(25)
z = x + y
print ('Sum of x and y = ', z)
```
4. Execute the code in question (3) in script mode.
5. Run Python in interpreted mode to execute the following code:

```
hours = 98
rate = 35.70
gross_pay = hours * rate
deduction = 10 * pay
net_pay = pay - deduction
print ('Take home pay = $', netpay)
```
6. Execute the code in question (5) in script mode.

Chapter 2

Python Basics

Learning Outcomes:

After completing this chapter, the student will be able to

- *Form identifiers and variables.*
- *Use arithmetic, logical and relational operators.*
- *Form expressions.*
- *Write assignment statements.*
- *Identify different data types.*
- *Perform input/output.*
- *Write simple Python code.*

2.1 Identifiers

Identifiers are used to name variables, functions, classes, objects and modules. They are simply references (pointers) to memory locations for storing objects such as numbers, strings, dates or Boolean values.

An identifier consists of one or more lower and upper letters (a–z, A–Z), digits (0–9) and the underscore (_). Identifiers must start with an underscore or a letter followed by zero or more letters, digits or underscores. Special characters such as #, &, @, \$ and % are not allowed in an identifier. The blank or space character is also not allowed in an identifier.

Python identifiers are *case-sensitive*, meaning it treats uppercase and lowercase letters as different characters. That means, NAME, Name and name are all different identifiers.

The following naming convention is used for identifiers:

- All class names start with uppercase letters.
- All other identifiers start with lowercase letters.
- An identifier with a single leading underscore indicates that it is private.
- An identifier with two leading underscores indicates that it is strongly private.
- An identifier that ends with two trailing underscores indicates that it is a special language-defined name.

The following are examples of valid identifiers.

k	rate_of_return	Employee	x
count	total_salary	suffix_	match
name	__init__	Account	found
_prefix	sum_of_x2	tax_rate_1	email
age	interest_rate	city	myList

The following are examples of invalid identifiers.