

Python 入门很简单

洪锦魁◎著

约500个完整程序实例

使用tkinter开发GUI程序

列表、元组、字典、集合

函数与类设计

设计系统模块

使用系统与外部模块

文件压缩与解压缩

文档读写与目录管理

剪贴板的应用

程序排错与异常处理

海龟绘图

图像处理

QR code制作

动画、游戏设计

中文、英文图表绘制

非外借

清华大学出版社

Python

入门很简单

洪锦魁◎著



清华大学出版社
北京

内 容 简 介

本书是一本针对初级 Python 读者而作的 Python 入门书。Python 语法非常灵活，本书作者尝试将 Python 语法的各种用法用实例讲解，以帮助读者更好地使用 Python。本书以约 500 个程序实例讲解了以下知识：完整 Python 语法；列表、元组、字典、集合；函数与类；程序排错与异常处理；动画、游戏设计；图像处理；使用 tkinter 开发 GUI 程序等。

为了提升阅读体验，对图书结构、案例选择以及代码样式都进行了精心设计，力争呈现给读者一本与众不同的编程书。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目(CIP)数据

Python入门很简单 / 洪锦魁著. — 北京：清华大学出版社，2019
ISBN 978-7-302-53063-3

I. ①P… II. ①洪… III. ①软件工具—程序设计 IV. ①TP311.561

中国版本图书馆 CIP 数据核字 (2019) 第 098353 号

责任编辑：杨迪娜 薛 阳

封面设计：杨玉兰

责任校对：徐俊伟

责任印制：丛怀宇

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>, <http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社总机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn

印装者：涿州市京南印刷厂

经 销：全国新华书店

开 本：170mm×240mm 印 张：26.25 字 数：687 千字

版 次：2019 年 7 月第 1 版 印 次：2019 年 7 月第 1 次印刷

定 价：69.00 元

产品编号：081904-01

序

作者多次与教育界的朋友相聚，谈到计算机语言的发展趋势，大家一致认为 Python 已经是当今最重要的计算机语言了，几乎所有知名公司，例如 Google、Facebook 等都已经将此语言列为必备计算机语言。许多人想学 Python，市面上的书也不少，但是目前市面上仍欠缺一本用简单程序实例完整讲解 Python 语法的书籍，造成学习者学习上的困难，因此作者决定撰写一本可以用丰富实例完整讲解 Python 语法的入门书籍。

Python 语法非常灵活，本书作者尝试将 Python 语法的各种用法用实例进行讲解，以使读者更灵活地使用 Python。

本书以约 500 个程序实例讲解了下列知识。

- (1) 完整 Python 语法；
- (2) 列表 (list)、元组 (tuple)、字典 (dict)、集合 (set)；
- (3) 函数与类；
- (4) 使用系统与外部模块；
- (5) 设计自己的模块；
- (6) Python 的输入与输出；
- (7) 文件压缩与解压缩；
- (8) 程序排错与异常处理；
- (9) 文档读写与目录管理；
- (10) 剪贴板处理；
- (11) QR code 制作；
- (12) 动画、游戏设计；
- (13) 图像处理；
- (14) 使用 tkinter 开发 GUI 程序；
- (15) 中文、英文 matplotlib 图表设计。

作者曾编写过许多计算机书籍，本书沿袭作者以往著作的特色，程序实例丰富，相信读者只要遵循本书思路必定可以在最短时间内精通 Python 设计。本书内容虽力求完美，但是书中不足与疏漏在所难免，请读者不吝指正。

目 录

第 1 章 基本概念	
1-1 认识 Python	2
1-2 Python 的起源	2
1-3 Python 语言发展史	3
1-4 Python 的应用范围	4
1-5 静态语言与动态语言	4
1-6 跨平台的程序语言	5
1-7 系统的安装与执行	5
1-7-1 在 Windows 中启动与执行 Python	5
1-7-2 在 Mac OS 中启动与执行 Python	5
1-8 文档的建立、存储、执行与打开	6
1-8-1 文档的建立	6
1-8-2 文档的存储	7
1-8-3 文档的执行	7
1-8-4 打开文档	8
1-9 程序注释	8
1-9-1 注释符号 #	8
1-9-2 三个单引号或双引号	8
习题	9
第 2 章 认识变量与基本数学运算	
2-1 用 Python 做计算	11
2-2 了解变量	11
2-3 了解程序的意义	12
2-4 了解注释的意义	13
2-5 Python 变量与其他程序语言的差异	13
2-6 变量的命名原则	14
2-7 基本数学运算	15
2-7-1 四则运算	15
2-7-2 余数和整除	15
2-7-3 次方	16
2-7-4 Python 语言控制运算的优先级	16
2-8 赋值运算符	16
2-9 Python 等号的多重指定使用	17
2-10 删除变量	18
2-11 Python 的断行	18
2-11-1 一行有多条命令	18
2-11-2 将一条命令分成多行	19
习题	19
第 3 章 Python 的基本数据类型	
3-1 type() 函数	22
3-2 数值数据类型	22
3-2-1 整数与浮点数的运算	23
3-2-2 二进制整数与函数 bin()	24
3-2-3 八进制整数	24
3-2-4 十六进制整数	24
3-2-5 强制数据类型的转换	25
3-2-6 数值运算常用的函数	25
3-2-7 eval() 函数	26
3-3 布尔值数据类型	27
3-4 字符串数据类型	27
3-4-1 字符串的连接	28
3-4-2 处理多行字符串	29
3-4-3 转义字符	29
3-4-4 强制转换为字符串	30
3-4-5 将字符串转换为整数	30
3-4-6 字符串数据的转换	31
3-4-7 字符串与整数相乘产生字符串复制效果	31
3-4-8 灵活使用字符串加法和换行字符 \n	32
3-4-9 字符串前加 r	32
习题	32

第 4 章 基本输入与输出

4-1 Python 的辅助说明 help()	35
4-2 格式化输出数据使用 print()	35
4-2-1 函数 print() 的基本语法	35
4-2-2 格式化 print() 输出	36
4-2-3 精准控制格式化的输出	38
4-2-4 format() 函数	39
4-3 输出数据到文档	40
4-3-1 打开一个文档 open()	40
4-3-2 使用 print() 函数输出数据到 文档	41
4-4 数据输入 input()	41
4-5 列出所有内建函数 dir()	42
习题	43

第 5 章 程序的流程控制

5-1 关系运算符	46
5-2 逻辑运算符	47
5-3 if 语句	48
5-4 if ... else 语句	50
5-5 if ... elif ... else 语句	51
5-6 嵌套的 if 语句	54
习题	55

第 6 章 列表

6-1 认识列表	58
6-1-1 列表基本定义	58
6-1-2 读取列表元素	59
6-1-3 列表切片	60
6-1-4 列表索引值是 -1	61
6-1-5 统计列表最大值 max()、 最小值 min()、总和 sum()	62
6-1-6 列表个数 len()	62
6-1-7 更改列表元素的内容	63
6-1-8 列表的相加	63
6-1-9 列表乘以一个数字	64
6-1-10 列表元素的加法运算	64
6-1-11 删除列表元素	65
6-1-12 空列表的判断	66
6-1-13 删除列表	67

6-2 Python 简单的面向对象概念	67
6-2-1 字符串的方法	67
6-2-2 更改字符串大小写	68
6-2-3 dir() 获得系统内部对象的方法	69
6-3 获得列表的方法	70
6-4 增加与删除列表元素	70
6-4-1 在列表末端增加元素 append()	70
6-4-2 插入列表元素 insert()	71
6-4-3 删除列表元素 pop()	72
6-4-4 删除指定的元素 remove()	72
6-5 列表的排序	73
6-5-1 颠倒排序 reverse()	73
6-5-2 sort() 排序	74
6-5-3 sorted() 排序	75
6-6 列表操作进阶	76
6-6-1 index()	76
6-6-2 count()	77
6-6-3 列表元素的组合 join()	77
6-7 嵌套列表	78
6-7-1 再谈 append()	79
6-7-2 extend()	80
6-8 列表的复制	80
6-8-1 列表复制 deep copy	80
6-8-2 地址的概念	81
6-8-3 列表的复制 shallow copy	82
6-9 再谈字符串	83
6-9-1 字符串的索引	83
6-9-2 字符串切片	83
6-9-3 函数或方法	84
6-9-4 将字符串转成列表	84
6-9-5 切片赋值应用	85
6-9-6 使用 split() 处理字符串	85
6-9-7 字符串的其他方法	85
6-10 in 和 not in 表达式	86
6-11 is 或 is not 表达式	87
6-11-1 整数变量在内存中的地址	87
6-11-2 将 is 和 is not 表达式应用于 整数变量	88

6-11-3 将 is 和 is not 表达式应用于 列表变量.....	89	8-4 修改元组内容产生错误的实例.....	122
6-12 enumerate 对象.....	89	8-5 修改元组元素.....	123
习题.....	90	8-6 元组切片.....	123
第 7 章 循环设计		8-7 方法与函数.....	124
7-1 基本 for 循环.....	96	8-8 列表与元组数据互换.....	125
7-1-1 for 循环基本流程.....	96	8-9 其他常用的元组方法.....	125
7-1-2 单行程序代码区块.....	97	8-10 在元组中使用 enumerate 对象.....	126
7-1-3 多行程序代码区块.....	98	8-11 元组的功能.....	126
7-1-4 将 for 循环应用在列表区间 元素.....	98	习题.....	127
7-2 range() 函数.....	99	第 9 章 字典	
7-2-1 只有一个参数的 range() 函数.....	99	9-1 字典的基本操作.....	129
7-2-2 有两个参数的 range() 函数.....	100	9-1-1 定义字典.....	129
7-2-3 有三个参数的 range() 函数.....	101	9-1-2 列出字典元素的值.....	130
7-2-4 一般应用.....	102	9-1-3 增加字典元素.....	130
7-2-5 进阶列表产生的应用.....	103	9-1-4 更改字典元素内容.....	131
7-3 for 循环应用进阶.....	104	9-1-5 删除字典特定元素.....	132
7-3-1 嵌套 for 循环.....	104	9-1-6 删除字典所有元素.....	132
7-3-2 强制离开 for 循环——break 命令... ..	105	9-1-7 删除字典.....	132
7-3-3 for 循环暂时停止不往下 执行——continue 命令.....	106	9-1-8 建立一个空字典.....	133
7-3-4 for ... else 语句.....	108	9-1-9 字典的复制.....	133
7-4 while 循环.....	109	9-1-10 取得字典元素数量.....	134
7-4-1 基本 while 循环.....	110	9-1-11 验证元素是否存在.....	134
7-4-2 嵌套 while 循环.....	112	9-1-12 设计字典的可读性技巧.....	135
7-4-3 强制离开 while 循环——break 命令.....	112	9-2 遍历字典.....	135
7-4-4 while 循环暂时停止不往下 执行——continue 命令.....	113	9-2-1 遍历字典的键 - 值.....	136
7-4-5 while 循环条件表达式.....	114	9-2-2 遍历字典的键.....	136
7-4-6 pass.....	115	9-2-3 排序与遍历字典.....	137
7-5 enumerate 对象使用 for 循环.... ..	115	9-2-4 遍历字典的值.....	138
习题.....	116	9-3 建立字典列表.....	139
第 8 章 元组		9-4 字典内含列表元素.....	140
8-1 元组的定义.....	121	9-5 字典内含字典.....	141
8-2 读取元组元素.....	121	9-6 while 循环在字典中的应用.....	142
8-3 遍历所有元组元素.....	122	9-7 字典常用的函数和方法.....	143
		9-7-1 len().....	143
		9-7-2 fromkeys().....	143
		9-7-3 get().....	144
		习题.....	145

第 10 章 集合

10-1 建立集合	148
10-1-1 使用大括号建立集合	148
10-1-2 使用 set() 函数定义集合	149
10-1-3 海量数据与集合的应用	150
10-2 集合的操作	151
10-2-1 交集	151
10-2-2 并集	152
10-2-3 差集	153
10-2-4 对称差集	154
10-2-5 等于	155
10-2-6 不等于	155
10-2-7 是成员 in	156
10-2-8 不是成员 not in	156
习题	156

第 11 章 函数设计

11-1 Python 函数基本概念	160
11-1-1 函数的定义	160
11-1-2 没有调用参数也没有 返回值的参数	161
11-1-3 在 Python Shell 中执行函数	162
11-2 函数的参数设计	162
11-2-1 传递一个参数	163
11-2-2 多个参数传递	163
11-2-3 关键词参数: 参数名称 = 值	165
11-2-4 参数默认值的处理	165
11-3 函数返回值	166
11-3-1 返回 None	166
11-3-2 简单返回数值数据	167
11-3-3 返回多条数据的应用	168
11-3-4 简单返回字符串数据	169
11-3-5 再谈参数默认值	169
11-3-6 函数返回字典数据	170
11-3-7 将循环应用于建立 VIP 会员 字典	171
11-4 调用函数时参数是列表	172
11-4-1 基本传递列表参数的应用	172
11-4-2 在函数内修改列表的内容	172

11-4-3 使用副本传递列表	174
11-5 传递任意数量的参数	176
11-5-1 传递任意数量的参数	176
11-5-2 设计含有一般参数与任意 数量参数的函数	176
11-5-3 设计含有一般参数与任意 数量的关键词参数	177
11-6 递归式函数设计	177
11-7 局部变量与全局变量	179
11-7-1 全局变量可以在所有 函数中使用	179
11-7-2 局部变量与全局变量使用 相同的名称	179
11-7-3 程序设计需注意事项	180
11-8 匿名函数 lambda	181
11-8-1 匿名函数 lambda 的语法	181
11-8-2 匿名函数的使用与 filter()	182
11-8-3 匿名函数的使用与 map()	184
11-9 pass 与函数	184
11-10 type 关键词应用于函数	184
习题	185

第 12 章 类——面向对象的程序设计

12-1 类的定义与使用	188
12-1-1 定义类	188
12-1-2 操作类的属性与方法	188
12-1-3 类的构建方法	189
12-1-4 属性初始值的设置	191
12-2 类的访问权限	192
12-2-1 私有属性	192
12-2-2 私有方法	193
12-3 类的继承	194
12-3-1 衍生类继承基类的实例应用	195
12-3-2 如何取得基类的私有属性	195
12-3-3 衍生类与基类有相同 名称的属性	196
12-3-4 衍生类与基类有相同 名称的方法	196
12-3-5 衍生类引用基类的方法	197

12-3-6 “三代同堂”的类与取得 基类的属性 <code>super()</code>	198
12-3-7 兄弟类属性的取得	199
12-4 多态	200
12-5 多重继承	202
习题	204
第 13 章 设计与应用模块	
13-1 将自定义的函数存储在模块中 ...	207
13-1-1 准备工作	207
13-1-2 建立函数内容的模块	208
13-2 应用自定义函数模块	208
13-2-1 <code>import</code> 模块名称	208
13-2-2 导入模块内特定函数	209
13-2-3 导入模块内多个函数	209
13-2-4 导入模块所有函数	209
13-2-5 使用 <code>as</code> 给函数指定替代名称 ...	210
13-2-6 使用 <code>as</code> 给模块指定替代名称 ...	210
13-3 将自定义类存储在模块内	211
13-3-1 准备工作	211
13-3-2 建立类内容的模块	212
13-4 应用自定义类模块	212
13-4-1 导入模块的单一类	212
13-4-2 导入模块的多个类	213
13-4-3 导入模块内所有类	213
13-4-4 <code>import</code> 模块名称	214
13-4-5 模块内导入另一个模块的类	214
13-5 随机数 <code>random</code> 模块	215
13-5-1 <code>randint()</code>	215
13-5-2 <code>choice()</code>	217
13-5-3 <code>shuffle()</code>	218
13-5-4 <code>sample()</code>	218
13-6 时间 <code>time</code> 模块	219
13-6-1 <code>time()</code>	219
13-6-2 <code>sleep()</code>	220
13-6-3 <code>asctime()</code>	220
13-6-4 <code>localtime()</code>	221
13-7 系统 <code>sys</code> 模块	221
13-7-1 <code>version</code> 属性	222

13-7-2 <code>stdin</code> 对象	222
13-7-3 <code>stdout</code> 对象	223
13-8 日期 <code>calendar</code> 模块	223
13-8-1 列出某年是否闰年 <code>isleap()</code>	223
13-8-2 输出月历 <code>month()</code>	223
13-8-3 输出年历 <code>calendar()</code>	224
习题	225
第 14 章 文档的读取与写入	
14-1 文件夹与文档路径	228
14-1-1 绝对路径与相对路径	228
14-1-2 <code>os</code> 模块与 <code>os.path</code> 模块	228
14-1-3 取得目前工作目录 <code>os.getcwd()</code>	229
14-1-4 取得绝对路径 <code>os.path.abspath()</code>	229
14-1-5 返回特定相对路径 <code>os.path.relpath()</code>	229
14-1-6 检查路径方法 <code>exist/isabs/isdir/isfile</code>	230
14-1-7 文档与目录的操作 <code>mkdir/rmdir/remove/chdir</code>	230
14-1-8 返回文档路径 <code>os.path.join()</code>	232
14-1-9 获得特定文档的大小 <code>os.path.getsize()</code>	233
14-1-10 获得特定工作 目录的内容 <code>os.listdir()</code>	233
14-1-11 获得特定工作目录内容 <code>glob</code> ...	234
14-1-12 遍历目录树 <code>os.walk()</code>	235
14-2 读取文档	236
14-2-1 读取整个文档 <code>read()</code>	237
14-2-2 <code>with</code> 关键词	237
14-2-3 逐行读取文档内容	238
14-2-4 逐行读取使用 <code>readlines()</code>	239
14-2-5 数据组合	240
14-2-6 字符串的替换	240
14-2-7 数据的查找	241
14-2-8 数据查找函数 <code>find()</code>	241
14-2-9 数据查找函数 <code>rfind()</code>	242

14-3 写入文档	243	15-2-5 捕捉所有异常	265
14-3-1 将执行结果写入空的文件内	243	15-3 抛出异常	266
14-3-2 写入数值数据	243	15-4 记录 Traceback 字符串	267
14-3-3 输出多行数据的实例	244	15-5 finally 语句	269
14-3-4 追加文件	245	15-6 程序排错的典故	269
14-4 shutil 模块	245	习题	270
14-4-1 文档的复制 copy()	246	第 16 章 海龟绘图	
14-4-2 目录的复制 copytree()	246	16-1 基本概念与模块安装	273
14-4-3 文档的移动 move()	246	16-2 绘图初体验	273
14-4-4 文件名的更改 move()	247	16-3 绘图基本练习	274
14-4-5 目录的移动 move()	247	16-4 控制画笔色彩与线条粗细	277
14-4-6 更改目录名称 move()	248	16-5 绘制圆、弧形或多边形	278
14-4-7 删除包含数据文件的目录		16-5-1 绘制圆或弧形	278
rmtree()	248	16-5-2 绘制多边形	281
14-4-8 安全删除文档或目录		16-6 认识与操作海龟图像	281
send2trash()	248	16-6-1 隐藏与显示海龟	282
14-5 文件压缩与解压缩 zipfile	249	16-6-2 认识所有的海龟游标	283
14-5-1 执行文档或目录的压缩	249	16-7 填充颜色	283
14-5-2 读取 zip 文档	250	16-8 颜色动画的设计	285
14-5-3 解压缩 zip 文档	250	16-9 绘图窗口的相关知识	286
14-6 剪贴板的应用	251	16-9-1 更改海龟窗口标题与背景颜色	286
14-7 综合应用	252	16-9-2 取得 / 更改窗口宽度与高度	287
习题	253	16-9-3 重设世界坐标	287
第 15 章 程序排错与异常处理		16-10 文字的输出	288
15-1 程序异常	257	习题	289
15-1-1 一个除数为 0 的错误	257	第 17 章 用 Python 处理图像文件	
15-1-2 撰写异常处理程序 try - except	257	17-1 认识 Pillow 模块的 RGBA	292
15-1-3 try - except - else	259	17-1-1 getrgb()	292
15-1-4 找不到文档错误		17-1-2 getcolor()	292
FileNotFoundError	259	17-2 Pillow 模块的盒子元组	293
15-1-5 分析单一文件的字数	260	17-3 图像的基本操作	294
15-1-6 分析多个文件的字数	261	17-3-1 打开图像对象	294
15-2 设计多组异常处理程序	262	17-3-2 图像大小属性	294
15-2-1 常见的异常对象	262	17-3-3 取得图像对象文件名	294
15-2-2 设计捕捉多个异常	263	17-3-4 取得图像对象的文件格式	295
15-2-3 使用一个 except 捕捉多个异常	264	17-3-5 存储文档	295
15-2-4 处理异常但是使用 Python		17-3-6 屏幕显示图像	295
内建的错误消息	264	17-3-7 建立新的图像对象	296

17-4 图像的编辑	297	18-12-2 图像与功能按钮的应用	333
17-4-1 更改图像大小	297	18-13 尺度的控制	334
17-4-2 图像的旋转	297	18-14 菜单设计	336
17-4-3 图像的翻转	299	习题	337
17-4-4 图像像素的编辑	299	第 19 章 动画与游戏	
17-5 裁切、复制与图像合成	300	19-1 绘图功能	340
17-5-1 裁切图像	300	19-1-1 建立画布	340
17-5-2 复制图像	301	19-1-2 绘制线条 create_line()	340
17-5-3 图像合成	301	19-1-3 绘制矩形 create_rectangle()	341
17-5-4 将裁切图片填满图像区间	302	19-1-4 绘制圆弧 create_arc()	342
17-6 图像滤镜	303	19-1-5 绘制圆或椭圆 create_oval()	343
17-7 在图像内绘制图案	304	19-1-6 绘制多边形 create_polygon()	343
17-7-1 绘制点	304	19-1-7 输出文字 create_text()	344
17-7-2 绘制线条	304	19-1-8 更改画布背景颜色	344
17-7-3 绘制圆或椭圆	305	19-2 尺度控制画布背景颜色	345
17-7-4 绘制矩形	305	19-3 动画设计	346
17-7-5 绘制多边形	305	19-3-1 基本动画	346
17-8 在图像内填写文字	306	19-3-2 多个球移动的设计	347
17-9 建立 QR code	307	19-3-3 将随机数应用于多个 球体的移动	347
习题	308	19-3-4 消息绑定	348
第 18 章 使用 tkinter 开发 GUI 程序		19-4 反弹球游戏设计	349
18-1 建立窗口	310	19-4-1 设计球往下移动	349
18-2 标签 Label	311	19-4-2 设计让球上下反弹	351
18-3 窗口组件配置管理员	313	19-4-3 设计让球在画布四面反弹	352
18-3-1 pack() 方法	313	19-4-4 建立球拍	352
18-3-2 grid() 方法	315	19-4-5 设计球拍移动	353
18-3-3 place() 方法	316	19-4-6 球拍与球碰撞的处理	354
18-3-4 窗口组件位置的总结	316	19-4-7 完整的游戏	356
18-4 功能按钮 Button	317	习题	358
18-5 变量类型	318	第 20 章 数据图表的设计	
18-6 文本框 Entry	319	20-1 绘制简单的折线图	360
18-7 文字区域 Text	323	20-1-1 显示绘制的图形 show()	360
18-8 滚动条 Scrollbar	324	20-1-2 画线 plot()	360
18-9 选项按钮 Radiobutton	325	20-1-3 线条宽度 linewidth	361
18-10 复选框 Checkbutton	328	20-1-4 标题的显示	362
18-11 对话框 messagebox	330	20-1-5 坐标轴刻度的设置	363
18-12 图像 PhotoImage	332	20-1-6 修改图表的起始值	363
18-12-1 图像与标签的应用	333		

20-1-7	多组数据的应用	364	20-7	圆饼图的制作	385
20-1-8	线条色彩与样式	365	20-8	图表显示中文	386
20-1-9	刻度设计	367		习题	387
20-1-10	图例 legend()	369	附录 A 安装 Python		
20-1-11	保存图表	372	A-1	Windows 操作系统中安装 Python	390
20-2	绘制散点图	372	A-2	Mac OS 操作系统中安装 Python	392
20-2-1	基本散点图的绘制	372	附录 B 安装第三方模块		
20-2-2	绘制系列点	373	B-1	pip 工具	395
20-2-3	设置绘图区间	374	B-1-1	Windows 系统 Python 3.7 安装在 C 盘	395
20-3	Numpy 模块	375	B-1-2	Python 3.7 安装在硬盘 更深层	395
20-3-1	建立一个简单的数组 linspace() 和 arange()	375	B-2	启动 DOS 与安装模块	396
20-3-2	绘制波形	376	B-2-1	DOS 环境	396
20-3-3	建立不等宽度的散点图	377	B-2-2	DOS 命令提示符	396
20-4	随机数的应用	377	B-3	导入模块安装更新版模块	397
20-4-1	一个简单的应用	378	B-4	安装更新版模块	397
20-4-2	随机数的移动	378	附录 C RGB 色彩表		
20-4-3	隐藏坐标	379	附录 D 本书习题解答		
20-5	绘制多个图表	380			
20-5-1	一个程序有多个图表	380			
20-5-2	含有子图的图表	381			
20-6	直方图的制作	383			



第 1 章

基本概念

本章摘要

- 1-1 认识 Python
- 1-2 Python 的起源
- 1-3 Python 语言发展史
- 1-4 Python 的应用范围
- 1-5 静态语言与动态语言
- 1-6 跨平台的程序语言
- 1-7 系统的安装与执行
- 1-8 文档的建立、存储、执行与打开
- 1-9 程序注释

1-1 认识 Python

Python 是一种直译式 (Interpreted)、面向对象 (Object-Oriented) 的程序语言, 它拥有完整的函数库, 可以协助用户轻松地完成许多常见的工作。

所谓的直译式语言是指, 直译器 (Interpreter) 会将程序代码一句一句直接执行, 不需要经过编译动作, 将语言先转换成机器码, 再予以执行。目前它的直译器是 CPython, 这是由 C 语言编写的一个直译程序, 与 Python 一样目前是由 Python 基金会管理使用的。

Python 也算是一个动态的高级语言, 具有垃圾回收功能。垃圾回收是指程序执行时, 直译程序会主动收回不再需要的动态内存空间, 将内存集中管理, 这种机制可以减轻程序设计师的负担, 当然也就减少了程序设计师犯错的机会。这种垃圾回收功能最早是 LISP, 后来的 Java、C# 等著名的程序语言也都支持这个功能。

由于 Python 开放源码 (Open Source), 每个人都可免费使用或为它贡献, 除了它本身有许多内建的包 (package) 或称模块 (module), 许多单位也为它开发了更多的包, 促使它的功能可以持续扩充, 因此 Python 目前已经是全球最热门的程序语言之一, 这也是本书的主题。

1-2 Python 的起源

Python 的最初设计者是吉多·范罗姆苏 (Guido van Rossum), 他于 1956 年出生于荷兰哈勒姆, 1982 年毕业于阿姆斯特丹大学的数学和计算机系, 获得硕士学位。



吉多·范罗姆苏 (Guido van Rossum) 在 1996 年由 O' Reilly 出版社作者 Mark Lutz 所著的 *Programming Python* 的序言中表示: 6 年前, 1989 年我想在圣诞节期间思考设计一种程序语言打发时间, 当时我正在构思一个新的脚本语言的解释器, 它是 ABC 语言的后代, 期待这个程序语言对 UNIX C 的程序语言设计师会有吸引力。基于我是蒙提派森飞行马戏团 (Monty Python's Flying Circus) 的疯狂爱好者, 所以就以 Python 为名当作这个程序的标题名称。

在一些 Python 的文件或有些书的封面上喜欢用蟒蛇代表 Python，从吉多·范罗姆苏的上述序言可知，Python 灵感的来源是**马戏团名称**而非**蟒蛇**。

1999年，他向美国国防部下的国防高等研究计划署（Defense Advanced Research Projects Agency, DARPA）提出 Computer Programming for Everybody 的研发经费申请，他提出了下列 Python 的目标。

- (1) 这是一个简单直觉式的程序语言，可以和主要程序语言一样强大。
- (2) 开放源码（Open Source），每个人都可自由使用与贡献。
- (3) 程序代码像英语一样容易理解与使用。
- (4) 可在短期内开发一些常用功能。

现在上述目标都已经实现了，Python 已经与 C/C++、Java 一样成为程序设计师必备的程序语言，然而它却比 C/C++ 和 Java 更容易学习。

目前 Python 语言是由 Python 软件基金会管理的，有关新版软件下载相关信息可以在这个基金会获得，可参考附录 A。

1-3 Python 语言发展史

1991年 Python 正式诞生，当时的操作系统平台是 Mac。尽管吉多·范罗姆苏坦承 Python 是构思于 ABC 语言，但是 ABC 语言并没有成功。吉多·范罗姆苏本人认为 ABC 语言并不是一个开放的程序语言，这是主要原因。因此，在 Python 的推广中，他避开了这个错误，将 Python 推向开放式系统，而获得了很大的成功。

1. Python 2.0 发表

2000年10月16日，Python 2.0 正式发表，主要是增加了**垃圾回收**的功能，同时支持 **Unicode**。

Unicode 是一种适合多语系的编码规则，主要思想是使用可变长度字节方式存储字符，以节省内存空间。例如，对于英文字母而言是使用 1B 空间存储即可，对于含有附加符号的希腊文、拉丁文或阿拉伯文等则用 2B 空间存储字符，中文则是以 3B 空间存储字符，只有极少数的平面辅助文字需要 4B 空间存储字符。也就是说，这种编码规则已经包含全球所有语言的字符了，所以采用这种编码方式设计程序时，其他语系的程序只要支持 Unicode 编码都可显示。例如，法国人即使使用法文版的程序，也可以正常显示中文。

2. Python 3.0 发表

2008年12月3日，Python 3.0 正式发表。一般程序语言的发展会考虑到兼容特性，但是 Python 3 在开发时为了不受到先前 2.x 版本的束缚，因此没有考虑兼容特性，所以许多早期版本开发的程序无法在 Python 3.x 版上执行。

不过为了解决这个问题，尽管发表了 Python 3.x 版本，后来陆续将 3.x 版的特性移植到 Python 2.6/2.7x 版上，所以现在进入 Python 基金会网站时，可以发现 2.7x 版和 3.7x 版的软件可以下载。

作者经验提醒：有一些早期开发的冒险游戏软件只支持 Python 2.7x 版，目前尚未支持 Python

3.7x 版。不过相信这些软件未来也将朝向支持 Python 3.7x 版的路迈进。

Python 基金会提醒：Python 2.7x 已经被确定为最后一个 Python 2.x 的版本。

作者在撰写此书时，基本上所有程序是以 Python 3.x 版本作为撰写此书的主要依据。

1-4 Python 的应用范围

尽管 Python 是一个非常适合初学者学习的程序语言，在国外有许多儿童程序语言教学也是以 Python 为工具，然而它却是一个功能强大的程序语言，下面是它的部分应用。

- (1) 设计动画游戏。
- (2) 支持图形接口 (Graphical User Interface, GUI) 开发。
- (3) 开发与管理网站。
- (4) 执行大数据分析。
- (5) 机器学习 (Machine Learning)、深度学习 (Deep Learning)、人工智能 (Artificial Intelligence)。
- (6) Google、Yahoo!、YouTube、NASA、Dropbox (文档分享服务)、Reddit (社交网站) 在内部都大量使用 Python 作为开发工具。
- (7) 黑客攻防。

1-5 静态语言与动态语言

变量是一个语言的核心，由变量的设置可以知道这个程序所要完成的工作。

有些程序语言的**变量**在使用前需要先声明它的数据类型，这样**编译程序**可以在内存内预留空间给这个变量。这个变量的数据类型经过声明后，无法再改变，这类程序语言称为**静态语言**，例如，C、C++、Java 等。其实声明变量可以协助计算机捕捉可能的错误，同时也可以让程序执行速度更快，但是程序设计员需要花更多的时间打字与思考程序的规划。

有些程序语言的变量在使用前不必声明它的数据类型，这样可以用比较少的程序代码完成更多工作，增加程序设计的便利性。这类程序在执行前不必经过**编译过程**，而是使用**直译器**直接**直译与执行**，这类程序语言称为**动态语言**，有时也可称这类语言是**脚本语言**，例如，Python、Perl、Ruby。动态语言执行速度比经过编译后的静态语言执行速度慢，所以有相当长的时间动态语言只适合进行短程序的设计，或是将它作为准备数据供静态语言处理，在这种状况下也有人将这种动态语言称为**胶水码 (glue code)**，但是随着软件技术的进步直译器执行速度越来越快，已经可以用它执行复杂的工作了。如果读者懂 Java、C、C++，可以发现，Python 相较于这些语言除了便利性，程序设计效率已经远远超过这些语言了，这也是 Python 成为目前最热门程序语言的原因。

Python 语言使用时可以直接在提示消息下 (>>>)，输入程序代码执行工作，可参考 1-7 节。也可以将程序代码存储成文档然后再执行，可参考 1-9 节。

1-6 跨平台的程序语言

Python 是一种跨平台的程序语言，主要的操作系统，例如，Windows、Mac OS、UNIX/Linux 等，都可以安装和使用，本书所有程序实例都在 Windows 和 Mac OS 下测试完成。

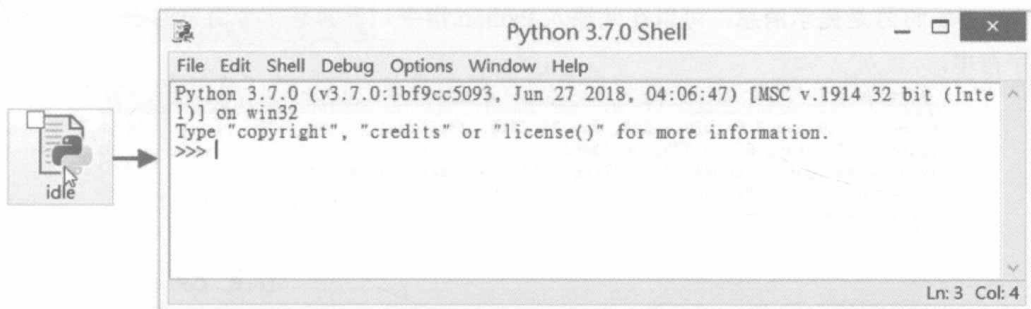
跨平台的程序语言意味着，你可以在某一个平台上使用 Python 设计一个程序，将来这个程序也可以在其他平台上顺利运行。

1-7 系统的安装与执行

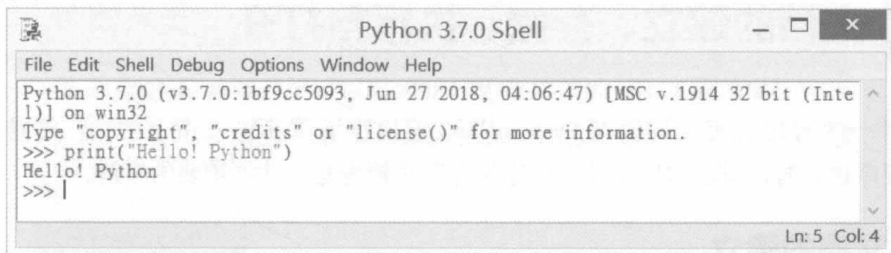
有关安装 Python 的步骤请参考附录 A。下面将以 Python 3.6x 版为例进行说明。

1-7-1 在 Windows 中启动与执行 Python

双击 Windows 桌面上的 idle 图标，将看到下列 Python Shell 窗口。



上图中 >>> 符号是提示消息，可以在此输入 Python 指令。下面是一个简单的 print() 函数，目的是输出字符串。



由上图可以确定成功执行了第一个 Python 的程序实例。

1-7-2 在 Mac OS 中启动与执行 Python

双击应用程序文件夹中的 Python 3.7 图标，将看到下列窗口。