



Python 应用编程丛书



# Python 实战编程： 从零学Python



黑马程序员 编著

中国铁道出版社  
CHINA RAILWAY PUBLISHING HOUSE

# Python 实战编程： 从零学 Python

黑马程序员 编著

图书在版编目(CIP)数据

Python 实战编程：从零学 Python / 黑马程序员编著. —北京：中国铁道出版社，2018.8

ISBN 978-7-113-23007-1

I. ①Py… II. ①黑… III. ①计算机—程序设计—教材 IV. ①TP311.567

中国版本图书馆CIP数据核字(2018)第169111号

定价：49.90元

ISBN 978-7-113-23007-1

9 787113 > 230071 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

01 >

中国铁道出版社  
CHINA RAILWAY PUBLISHING HOUSE

## 内 容 简 介

本书涵盖了 Python 开发的核心知识。其中，第 1 ~ 13 章主要讲解的是 Python 的核心语法，包括基本概念和语句、风格、数据类型、字符串、序列、流程控制、字典和集合、函数和函数式编程、文件和面向对象编程；第 14 ~ 20 章讲解进阶内容，如错误和异常的处理、模块、内存管理、正则表达式、图形界面、多任务编程、网络编辑、数据库、Web 编程等；第 21 ~ 25 章分别通过井字棋、贪吃蛇、飞机大战、多人聊天室、天天生鲜这 5 个项目，全面巩固前面所学知识。如果读者能结合本书数以百计的代码片段、交互实例，相信一定可以加强 Python 技能的实用练习。

本书适合 Python 初学者以及已经入门但想继续学习和提高自身 Python 技巧的程序员。

### 图书在版编目 (CIP) 数据

Python 实战编程：从零学 Python/ 黑马程序员编著. —  
北京：中国铁道出版社，2018. 8

(Python 应用编程丛书)

ISBN 978-7-113-24007-3

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

中国版本图书馆 CIP 数据核字 (2018) 第 166466 号

书 名：Python 实战编程：从零学 Python

作 者：黑马程序员 编著

策 划：秦绪好 翟玉峰

读者热线：(010) 63550836

责任编辑：翟玉峰 鲍 闻

封面设计：王 哲

封面制作：刘 颖

责任校对：张玉华

责任印制：郭向伟

出版发行：中国铁道出版社 (100054, 北京市西城区右安门西街 8 号)

网 址：<http://www.tdpress.com/5leds/>

印 刷：中煤 (北京) 印务有限公司

版 次：2018 年 8 月第 1 版 2018 年 8 月第 1 次印刷

开 本：787 mm×1 092 mm 1/16 印张：42.5 字数：976 千

书 号：ISBN 978-7-113-24007-3

定 价：108.00 元



版权所有 侵权必究

凡购买铁道版图书，如有印制质量问题，请与本社教材图书营销部联系调换。电话：(010) 63550836

打击盗版举报电话：(010) 51873659

人工智能（简称 AI）就其本质而言，是对人的思维的信息过程的模拟，它自诞生以来，相关的理论和技术日益成熟，其应用领域也在不断扩大。例如，机器人、语音识别领域中的智能音箱、图像识别领域中的智慧交通系统、自然语言处理领域中的自动翻译等都涉及人工智能，可以说，人工智能有助于提高我们的生活指数。

Python 在今天变得如此重要，一个重要的原因便是它能更方便地为我们的工作以及生活创造智能的特性。互联网飞速发展，积累了大量可供分析的数据，对这些数据进行处理、分析以及预测能力的要求显著提升，而 Python 正是以数据科学而闻名，它拥有着极其丰富且稳定的数据科学工具环境，从而助推其成为大数据和云计算中最流行的语言之一。而它的这种数据科学基因，也自然地延伸到了机器学习领域，今天，我们非常熟悉的众多机器学习库，如 scikit-learn、Tensorflow 等都基于或支持 Python 语言开发，我们可以很方便地使用它们，去构建自己的智能应用。

### 为什么学习本书

Python 是一种面向对象的解释型计算机程序设计语言，它作为人工智能的最佳语言，具有下列优势：

- （1）优质的文档。
- （2）与平台无关，Python 基本可以在任何平台上使用。
- （3）和其他面向对象编程语言相比，学习 Python 更加简单快捷。
- （4）Python 的设计非常好，快速、健壮、可移植、可扩展，很明显这些对于人工智能应用来说都是非常重要的因素。
- （5）Python 提供了丰富而功能强大的库，这些库可以帮助程序员实现各个领域的开发。
- （6）Python 是开源的，可以得到相同的社区支持。

本书涵盖了 Python 程序设计的方方面面，不仅是语法，还加入了很多高阶知识，通过数以百计的代码片段、交互实例和多个综合案例，让读者真正提高 Python 技能。

### 如何学习本书

本书基于 Python 3，系统全面地讲解 Python 开发的核心知识。全书共有 25 章，大致可以分为 3 个部分，具体如下：

第 1 部分：核心语法（第 1 ~ 13 章）

这部分内容占据了大约二分之一的篇幅，讲解 Python 程序开发的核心内容，包括基本概念和语句、风格和语法、数据类型、序列类型、映射和集合类型、条件和循环、文件、错误和异常、函数和函数式编程、模块、面向对象编程等，这部分内容是 Python 开发的必备知识，通过该阶段的学习，读者将具备 Python 的基础知识，建立面向对象的编程思想。

### 第 2 部分：高阶知识（第 14 ~ 20 章）

高阶内容讲解的是诸如内存管理、正则表达式、网络、多线程、图形界面、数据库、Web 等知识。这部分内容旨在帮助读者提高 Python 编程水平，无论大家以后使用 Python 开发哪个领域，都会用到这部分内容。

### 第 3 部分：项目实战（第 21 ~ 25 章）

这个部分包含井字棋、贪吃蛇、飞机大战、多人聊天室、天天生鲜这 5 个项目。其中，井字棋、贪吃蛇、飞机大战这三个项目是对第 1 部分核心语法知识的综合应用，多人聊天室是对高阶知识的综合应用，天天生鲜这个项目是借用 Django 框架开发的一个电商项目，含金量非常高。在讲解项目实战的过程中，我们首先介绍的是项目背景和实现目标，然后阐述项目的分析方法和过程，最后按照项目的构建顺序完成项目开发，这样编排的目的，不仅是让读者增加项目经验，更是为了加深读者对项目开发的理解。

## 致谢

本书的编写和整理工作由传智播客教育科技有限公司完成，主要参与人员有吕春林、高美云、刘传梅、郑瑶瑶、王晓娟、邢文鹏、刘凡、郝跃新、孔德海、丁佼、孟宝亮等。全体人员在近一年的编写过程中付出了很多辛勤的汗水，在此一并表示衷心的感谢。

## 意见反馈

尽管我们付出了最大的努力，但书中难免会有不妥之处，欢迎各界专家和读者朋友们来信给予宝贵意见，我们将不胜感激。您在阅读本书时，如发现任何问题或有不认同之处可以通过电子邮件与我们取得联系。

请发送电子邮件至：[itcast\\_book@vip.sina.com](mailto:itcast_book@vip.sina.com)

黑马程序员

2018 年 5 月

# CONTENTS

# 目 录

## 第 1 章 开启 Python 学习之旅..... 1

1.1 Python 的由来.....	1
1.2 Python 的特点.....	2
1.2.1 简单易学.....	2
1.2.2 开源.....	2
1.2.3 高级语言.....	2
1.2.4 可移植性.....	2
1.2.5 解释型.....	2
1.2.6 面向对象.....	3
1.2.7 可扩展性和可嵌入性.....	3
1.2.8 丰富的库.....	3
1.2.9 规范的代码.....	3
1.3 Python 应用领域.....	3
1.3.1 Web 应用开发.....	3
1.3.2 管理操作系统、服务器自动化运行和维护.....	4
1.3.3 科学计算.....	4
1.3.4 桌面软件.....	4
1.3.5 服务器软件（网络软件）... ..	4
1.3.6 游戏.....	4
1.3.7 构思实现，产品早期原型和迭代.....	4
1.4 Python 版本及解释器.....	5
1.4.1 Python 的版本过渡.....	5
1.4.2 Python 2 和 Python 3 的区别.....	6
1.4.3 Python 解释器.....	9
1.5 下载和安装 Python.....	9
1.5.1 Linux 平台.....	9
1.5.2 Windows 平台.....	10
1.5.3 Mac 平台.....	13

1.6 第一个 Python 程序——HelloWorld.....	14
1.6.1 Python 命令行.....	14
1.6.2 写一段小程序.....	14
1.6.3 脚本.....	14
1.7 运行 Python.....	15
1.7.1 命令行上的交互式解释器.....	15
1.7.2 从命令行启动脚本.....	16
1.7.3 增强交互式 IPython.....	16
1.7.4 集成开发环境.....	17
1.7.5 PyCharm 的下载安装.....	18
1.7.6 PyCharm 的使用.....	20
1.8 Python 程序执行原理.....	24
1.8.1 编译型语言和解释型语言.....	24
1.8.2 Python 是一种解释型语言.....	24
1.9 Python 文档.....	25

## 第 2 章 Python 快速入门..... 26

2.1 程序的输出和输入.....	26
2.1.1 程序的输出.....	26
2.1.2 程序的输入.....	27
2.2 Python 注释.....	27
2.2.1 行注释和块注释.....	28
2.2.2 文档字符串.....	28
2.3 代码风格建议.....	28
2.3.1 使用缩进表示语句块.....	28
2.3.2 关于缩进风格.....	29
2.3.3 代码过长的换行.....	29
2.4 标识符和关键字.....	29
2.4.1 标识符.....	29

2.4.2 关键字	30	3.3.1 浮点型的表示方式	48
2.5 变量	30	3.3.2 浮点数的取值范围	49
2.5.1 变量和赋值	30	3.3.3 浮点数的存储	49
2.5.2 复用变量名	31	3.3.4 高精度浮点数	49
2.6 数字类型	32	3.4 复数类型	50
2.7 布尔类型	32	3.4.1 复数类型定义	50
2.8 字符串	33	3.4.2 复数的特点	50
2.9 元组和列表	33	3.4.3 创建复数	50
2.10 字典	34	3.4.4 获取复数的实部和虚部	51
2.11 集合 (set)	34	3.5 布尔类型	51
2.12 if 语句	35	3.6 数字运算	51
2.13 while 循环	36	3.6.1 算术运算符	51
2.14 for 循环	37	3.6.2 赋值运算符	53
2.15 可迭代对象	38	3.6.3 比较运算符	53
2.16 文件读写	38	3.6.4 逻辑运算符	54
2.17 错误和异常	39	3.7 运算符优先级	55
2.18 函数	40	3.8 类型转换	56
2.19 类和对象	40	3.8.1 类型转换函数	56
2.20 模块	41	3.8.2 类型转换注意事项	56
<b>第 3 章 数值类型</b>	<b>42</b>	3.8.3 类型转换示例	57
3.1 整型	42	3.9 对象和引用	57
3.1.1 早期的整型	42	3.9.1 对象	57
3.1.2 Python 3 中的整型	42	3.9.2 引用	58
3.1.3 整型的表示方法	43	3.9.3 身份运算符	59
3.1.4 进制转换	43	3.9.4 身份运算符的使用	59
3.2 位运算	44	<b>第 4 章 字符串</b>	<b>60</b>
3.2.1 整型存储方式	44	4.1 字符串介绍	60
3.2.2 按位取反	45	4.2 字符串存储方式	61
3.2.3 按位左移	45	4.2.1 字符串的索引	61
3.2.4 按位右移	46	4.2.2 根据索引访问值	62
3.2.5 按位与	47	4.2.3 字符串是不可变的	62
3.2.6 按位或	48	4.3 切片截取字符串	63
3.2.7 按位异或	48	4.3.1 切片和步长	63
3.3 浮点型	48	4.3.2 切片的默认值	64

4.3.3 切片的正反向索引混用.....	64	5.2.5 序列相加.....	85
4.4 特殊字符处理.....	65	5.2.6 序列与数字相乘.....	85
4.4.1 字符串的转义.....	65	5.2.7 计算序列长度.....	85
4.4.2 保持原始字符串.....	66	5.2.8 找出序列的最大元素和最小 元素.....	85
4.5 字符串的输出和输入.....	66	5.2.9 查找元素在序列中出现的 位置.....	86
4.5.1 字符串的格式化输出.....	66	5.2.10 统计元素在序列中出现的 次数.....	86
4.5.2 格式化操作的辅助指令.....	67	5.3 可变序列——列表 (list).....	86
4.5.3 字符串的输入.....	68	5.3.1 什么是列表.....	86
4.6 字符串模板.....	69	5.3.2 列表的创建方式.....	87
4.7 字符串的内建函数.....	70	5.3.3 列表操作符.....	88
4.7.1 序列类型操作相关函数.....	70	5.3.4 列表类型相关函数.....	90
4.7.2 字符串类型转换相关函数.....	71	5.3.5 列表类型相关方法.....	91
4.8 字符串的常见方法.....	72	5.4 不可变序列——元组.....	95
4.8.1 find 方法.....	74	5.4.1 元组的创建方式.....	96
4.8.2 index 方法.....	75	5.4.2 元组支持通用序列操作.....	96
4.8.3 replace 方法.....	75	5.4.3 单个元素的元组.....	97
4.8.4 split 方法.....	75	5.4.4 有时候元组也“可变”.....	97
4.8.5 lower 方法.....	76	5.5 元组是无符号对象的默认类型.....	98
4.8.6 strip 方法.....	76	5.6 不同序列的相互转换.....	98
4.8.7 format 方法.....	76	5.7 不可变序列——range.....	99
4.9 字符串操作符.....	78	5.7.1 range 介绍.....	99
4.9.1 使用比较运算符比较 字符串.....	78	5.7.2 range 支持通用序列操作.....	100
4.9.2 使用 + 运算符连接字符串.....	79	5.7.3 range 与列表和元组的 比较.....	101
4.9.3 使用成员运算符检查 字符串.....	80	<b>第 6 章 流程控制..... 102</b>	
<b>第 5 章 序列..... 81</b>		6.1 if 语句.....	102
5.1 认识什么是序列.....	81	6.1.1 简单 if 语句.....	102
5.2 通用序列操作.....	82	6.1.2 条件表达式.....	103
5.2.1 索引 ([]).....	82	6.1.3 多重条件表达式.....	103
5.2.2 切片 ([i:j]).....	82	6.1.4 if-else 语句.....	104
5.2.3 步长 ([i:j:k]).....	83		
5.2.4 判断某个元素是否属于 序列.....	84		



6.1.5	elif 语句	104
6.1.6	if 嵌套语句	106
6.1.7	if 语句的缩进	107
6.1.8	判断表达式	107
6.2	循环语句	108
6.2.1	while 循环	108
6.2.2	while 循环嵌套	109
6.2.3	死循环	110
6.2.4	for 语句	110
6.2.5	range 函数用于 for 循环	111
6.2.6	循环技巧	111
6.2.7	迭代器和 iter() 函数	113
6.2.8	列表推导式	114
6.3	跳出循环	114
6.3.1	break 语句	114
6.3.2	continue 语句	115
6.3.3	pass 语句	116
6.3.4	else 语句	116

## 第 7 章 字典和集合..... 118

7.1	字典类型	118
7.1.1	字典的基本概念	118
7.1.2	字典是可变的容器	119
7.1.3	字典键和值的规范	119
7.2	字典的基本操作	120
7.2.1	创建字典并为元素赋值	120
7.2.2	访问字典中的元素	121
7.2.3	遍历字典中的元素	122
7.2.4	更新字典中的元素	123
7.2.5	删除字典元素和字典	123
7.3	字典常用的函数	124
7.3.1	通过 len 函数获取字典 元素的数量	124
7.3.2	通过 hash 函数判断某个 对象是否能为字典的键	124

7.4	字典的内建方法	124
7.4.1	copy 方法	125
7.4.2	get 方法	126
7.4.3	items 方法	127
7.4.4	keys 方法	127
7.4.5	values 方法	128
7.5	集合类型	128
7.5.1	集合的基本概念	128
7.5.2	集合的类型	129
7.6	集合的基本操作	130
7.6.1	集合的创建和赋值	130
7.6.2	访问集合中的元素	130
7.6.3	更新集合中的元素	131
7.6.4	删除集合本身	132
7.7	集合操作符	133
7.7.1	标准类型的操作符	133
7.7.2	集合类型的操作符	134
7.7.3	可变集合类型的操作符	135
7.8	集合内建方法	136

## 第 8 章 函数和函数式编程..... 138

8.1	认识函数	138
8.2	函数的定义和调用	139
8.2.1	函数的定义	139
8.2.2	函数的调用	139
8.2.3	函数文档说明	140
8.2.4	形参和实参	141
8.2.5	传递参数值的改变	141
8.2.6	return 语句	141
8.2.7	返回值的类型	142
8.3	函数的参数传递	143
8.3.1	位置参数传递	143
8.3.2	关键字参数传递	143
8.3.3	参数默认值	144
8.3.4	可变参数	144

8.3.5 解包裹 .....	145	8.12.1 反过头看看 .....	172
8.3.6 混合传递方式 .....	146	8.12.2 数字运算相关函数 .....	173
8.4 递归函数 .....	147	8.12.3 类型转换相关函数 .....	177
8.5 函数作为参数传递 .....	148	8.12.4 序列操作相关函数 .....	179
8.6 lambda 匿名函数 .....	148	8.12.5 类、对象、属性相关 函数 .....	180
8.7 常用函数 .....	149	8.12.6 编译, 执行相关函数 .....	183
8.7.1 map 函数 .....	149	8.12.7 其他函数 .....	186
8.7.2 filter 函数 .....	150	<b>第9章 文件..... 188</b>	
8.7.3 reduce 函数 .....	151	9.1 文件的打开和关闭 .....	188
8.8 变量作用域 .....	152	9.1.1 打开文件 .....	188
8.8.1 全局变量和局部变量 .....	152	9.1.2 关闭文件 .....	188
8.8.2 global 和 nonlocal 关键字 .....	153	9.2 文件概述 .....	189
8.9 闭包 .....	155	9.2.1 路径名 .....	189
8.9.1 函数的引用 .....	155	9.2.2 打开模式 .....	189
8.9.2 什么是闭包 .....	156	9.2.3 文件缓冲 .....	190
8.9.3 闭包的使用 .....	156	9.2.4 文件的分类 .....	191
8.10 装饰器 .....	157	9.2.5 标准文件 .....	191
8.10.1 什么是装饰器 .....	157	9.3 文件的读写 .....	192
8.10.2 多个装饰器 .....	159	9.3.1 写文件 .....	192
8.10.3 装饰器对有参数函数进行 装饰 .....	160	9.3.2 读文件 .....	192
8.10.4 装饰器对带有返回值的 函数进行装饰 .....	162	9.3.3 文件读写位置 .....	193
8.10.5 通用装饰器 .....	163	9.4 文件的方法和属性 .....	195
8.10.6 带有参数的装饰器 .....	164	9.5 文件遍历 .....	195
8.11 生成器 .....	165	9.5.1 read() 遍历文件 .....	195
8.11.1 什么是生成器 .....	165	9.5.2 readline() 遍历文件 .....	196
8.11.2 生成器运行机制 .....	166	9.5.3 readlines() 遍历文件 .....	197
8.11.3 yield 与 return .....	167	9.5.4 文件迭代器 .....	197
8.11.4 生成器支持的方法 .....	168	9.5.5 文件迭代器的特点 .....	198
8.11.5 生成器应用场景—— 协程 .....	170	9.6 文件备份 .....	198
8.11.6 生成器是一个迭代器 .....	171	9.7 文件重命名 .....	199
8.12 Python 标准库——内置函数 .....	172	9.8 删除文件 .....	199
		9.9 os 模块扩展 .....	200
		9.9.1 创建文件夹 .....	200

9.9.2 删除文件夹 ..... 200

9.9.3 获取当前目录 ..... 200

9.9.4 更改默认目录 ..... 200

9.9.5 获取目录列表 ..... 201

9.10 文件操作案例——用户登录 ..... 201

9.10.1 业务逻辑 ..... 201

9.10.2 相关文件 ..... 202

9.10.3 模块功能 ..... 202

9.10.4 功能实现 ..... 202

9.10.5 功能演示 ..... 206

**第 10 章 面向对象（上） ..... 208**

10.1 面向对象的概念 ..... 208

10.2 类和对象 ..... 210

10.3 类的定义和使用 ..... 210

10.3.1 类的定义 ..... 210

10.3.2 创建对象 ..... 211

10.3.3 新式类和经典类 ..... 211

10.4 属性和方法 ..... 211

10.4.1 类属性 ..... 211

10.4.2 对象属性 ..... 212

10.4.3 构造方法和析构方法 ..... 214

10.4.4 对象方法 ..... 215

10.4.5 类方法 ..... 217

10.4.6 静态方法 ..... 218

10.4.7 保护对象的私有属性 ..... 219

10.4.8 更便捷地访问私有属性  
——使用 @property ..... 220

**第 11 章 面向对象（下） ..... 223**

11.1 面向对象的三大特征 ..... 223

11.2 封装 ..... 224

11.3 继承 ..... 224

11.3.1 单继承 ..... 224

11.3.2 isinstance() 和 issubclass()  
函数 ..... 226

11.3.3 多继承 ..... 227

11.3.4 重写 ..... 229

11.3.5 super 关键字 ..... 229

11.4 多态 ..... 230

11.5 运算符重载 ..... 231

11.5.1 四则运算重载 ..... 232

11.5.2 索引和分片重载 ..... 233

11.5.3 定制对象的字符串形式 ..... 234

11.6 \_\_new\_\_ 方法 ..... 235

11.7 单例模式 ..... 236

11.8 工厂模式 ..... 238

**第 12 章 错误和异常 ..... 240**

12.1 错误和异常简介 ..... 240

12.1.1 异常 ..... 241

12.1.2 Python 常见异常 ..... 241

12.2 捕获异常 ..... 243

12.2.1 try-except 语句 ..... 243

12.2.2 获取异常提示信息 ..... 244

12.2.3 捕获多个异常 ..... 244

12.2.4 捕获所有异常 ..... 245

12.2.5 else 子句 ..... 246

12.2.6 finally 子句 ..... 246

12.3 抛出异常 ..... 247

12.3.1 raise 语句 ..... 247

12.3.2 异常的传递 ..... 249

12.3.3 assert 断言语句 ..... 250

12.4 自定义异常 ..... 251

12.5 with 语句处理异常 ..... 253

12.5.1 with 语句 ..... 253

12.5.2 上下文管理器 ..... 254

12.5.3 自定义上下文管理器 ..... 255

**第 13 章 模块 ..... 257**

13.1 什么叫模块 ..... 257

13.2 常用标准模块 ..... 257

13.2.1	sys 模块	257
13.2.2	os 模块	258
13.2.3	os.path 模块	259
13.2.4	random 模块	260
13.2.5	time 模块	260
13.3	导入模块	262
13.3.1	import	262
13.3.2	from...import...	262
13.3.3	from...import *	263
13.4	导入方法比较	263
13.5	自制模块	263
13.5.1	制作自定义模块	263
13.5.2	自定义模块的使用	263
13.5.3	模块搜索路径	264
13.6	模块属性	264
13.6.1	__all__ 属性	264
13.6.2	__name__ 属性	266
13.7	模块导入特性	266
13.7.1	导入特性概述	266
13.7.2	导入特性详解	267
13.8	模块缓存	267
13.9	Python 中的包	268
13.10	包的导入	268
13.10.1	import 导入	268
13.10.2	from...import...导入	268
13.10.3	__init__.py 文件	269
13.11	模块的打包发布	269
13.11.1	待发布包结构	270
13.11.2	setup.py 文件	270
13.11.3	模块构建	270
13.11.4	模块打包	271
13.12	模块安装	272

## 第 14 章 内存管理 ..... 273

14.1	内存管理概述	273
------	--------	-----

14.1.1	为什么要进行内存管理	273
14.1.2	内存管理的常用机制	273
14.2	引用计数机制	274
14.2.1	引用计数机制概述	274
14.2.2	增加对象的引用	275
14.2.3	减少对象的引用	276
14.2.4	释放对象占用的内存	276
14.3	容器对象引起的循环引用	277
14.3.1	容器对象引用对象	277
14.3.2	对象之间的循环引用	277
14.4	弱引用解决自定义对象的循环引用	279
14.4.1	弱引用概述	279
14.4.2	弱引用处理循环引用	281
14.5	垃圾回收机制	281
14.5.1	垃圾回收机制概述	282
14.5.2	标记清除技术	282
14.5.3	分代回收技术	283
14.5.4	gc 模块概述	284
14.5.5	启动垃圾回收的场景	284
14.5.6	gc 模块自动回收垃圾	285
14.6	内存池机制	286
14.6.1	小整数对象池	286
14.6.2	字符串的 intern 机制	287

## 第 15 章 正则表达式 ..... 288

15.1	起源与发展	288
15.2	正则表达式的定义	289
15.3	字符分类	289
15.3.1	普通字符	289
15.3.2	元字符	289
15.3.3	预定义字符集	290
15.4	点字符	290
15.5	位置匹配	290
15.5.1	行头行尾	291

15.5.2	单词边界	291	16.3.1	按钮组件 (Button)	309
15.6	多选结构	291	16.3.2	标签组件 (Label)	310
15.7	字符组	291	16.3.3	文本框 (Entry)	310
15.7.1	连字符	292	16.3.4	多行文本框 (Text)	311
15.7.2	排除型字符组	292	16.3.5	单选按钮 (Radiobutton)	312
15.7.3	元字符与字符组	292	16.3.6	复选框 (Checkbutton)	314
15.8	重复匹配	293	16.3.7	列表框 (Listbox)	315
15.8.1	可选模式	293	16.4	菜单	316
15.8.2	重复模式	293	16.4.1	主菜单	316
15.9	转义字符	293	16.4.2	下拉菜单	316
15.10	子组	294	16.4.3	弹出式菜单	317
15.10.1	子组的意义	294	16.5	对话框	318
15.10.2	非捕获子组	295	16.5.1	消息对话框 (messagebox)	318
15.11	re 模块	295	16.5.2	文件对话框 (filedialog)	318
15.12	预编译	296	16.5.3	颜色选择对话框 (colorchooser)	319
15.13	匹配与搜索	296	16.6	画布绘图	320
15.13.1	match()	296	16.7	Tkinter 几何布局管理器	322
15.13.2	search()	297	16.7.1	pack 几何布局管理器	322
15.14	匹配对象	298	16.7.2	grid 几何布局管理器	323
15.15	全文匹配	299	16.7.3	place 几何布局管理器	323
15.15.1	findall()	299	16.8	事件处理	324
15.15.2	finditer()	300	16.8.1	Tkinter 中的事件	324
15.16	检索替换	300	16.8.2	事件对象及属性	325
15.17	文本分割	301	16.9	其他 GUI	327
15.18	贪婪匹配	301	16.9.1	wxPython	327
<b>第 16 章 图形用户界面编程..... 303</b>			16.9.2	PyGTK	327
16.1	Tkinter 概述	303	16.9.3	PyQt	328
16.1.1	认识 Tkinter	303	16.9.4	JPython	328
16.1.2	开发第一个 Tkinter 程序	303	<b>第 17 章 多任务编程 ..... 329</b>		
16.2	Tkinter 组件概述	304	17.1	多任务的实现原理	329
16.2.1	Tkinter 核心组件	304	17.2	进程介绍	330
16.2.2	标准组件属性	305	17.2.1	什么是进程	330
16.2.3	组件的属性设置	308			
16.3	常用组件介绍	309			

17.2.2	进程的状态 .....	331	17.9	创建线程.....	347
17.3	通过 fork() 函数创建进程.....	331	17.9.1	使用 Thread 类实现	
17.3.1	使用 fork() 函数创建子			多线程.....	347
	进程 .....	332	17.9.2	使用 Thread 子类实现	
17.3.2	使用 fork() 函数创建多个			多线程.....	349
	子进程.....	333	17.9.3	调用 join() 方法阻塞	
17.4	通过 Process 类创建进程			线程 .....	349
	(跨平台) .....	334	17.10	解决线程共享资源产生的	
17.4.1	通过 Process 类实例创建			问题.....	350
	进程 .....	334	17.10.1	线程共享全局变量.....	350
17.4.2	使用 join() 方法同步		17.10.2	访问全局变量出现数据	
	进程 .....	335		不同步的问题.....	351
17.4.3	通过 Process 子类创建子		17.10.3	通过互斥锁 (Lock) 解决	
	进程 .....	337		数据不同步的问题 .....	352
17.5	进程池批量创建进程.....	338	17.10.4	持有多个锁造成死锁的	
17.5.1	进程池非阻塞式添加			问题.....	353
	任务 .....	338	17.10.5	可重入锁 (RLock)	
17.5.2	进程池阻塞式添加任务...340			的使用.....	355
17.6	通过 subprocess 模块创建		17.11	线程的同步应用 .....	356
	进程.....	341	17.11.1	通过条件变量 (Condition)	
17.6.1	调用 call() 函数创建子			实现线程同步 .....	357
	进程 .....	342	17.11.2	使用队列实现线程同步...358	
17.6.2	通过 Popen 类实例创建子		17.12	使用事件 (Event) 实现线程	
	进程 .....	342		通信 .....	361
17.7	通过 Queue 实现进程间的		17.13	后台线程.....	362
	通信.....	344	17.14	协程.....	363
17.7.1	全局变量在多个进程中不		17.14.1	协程的优势 .....	363
	共享.....	344	17.14.2	Python 2.x 的协程 .....	363
17.7.2	使用 Queue 队列实现进程		17.14.3	Python 3.x 的协程 .....	364
	间通信.....	344	<b>第 18 章 网络编程 .....</b>	<b>366</b>	
17.8	线程介绍.....	345	18.1	计算机网络概述.....	366
17.8.1	什么是线程.....	346	18.2	协议与体系结构.....	367
17.8.2	线程的状态.....	346	18.2.1	体系结构.....	367
17.8.3	线程类型.....	346			

18.2.2	协议.....	368	18.12.1	select 并发服务器.....	390
18.2.3	数据传输流程.....	369	18.12.2	epoll 并发服务器.....	393
18.3	网络架构.....	370	<b>第 19 章 数据库编程 ..... 397</b>		
18.4	IP 地址.....	370	19.1	MySQL 数据库.....	397
18.4.1	IPv4 的分类.....	371	19.1.1	下载和安装.....	398
18.4.2	IPv6 地址.....	372	19.1.2	使用 MySQL Workbench.....	408
18.4.3	子网掩码.....	373	19.1.3	卸载 MySQL.....	409
18.5	端口号.....	373	19.2	SQL 语言.....	412
18.6	网络编程概述.....	374	19.3	数据库基本操作.....	412
18.7	套接字.....	374	19.3.1	创建数据库.....	413
18.8	socket 通信流程.....	375	19.3.2	删除数据库.....	413
18.8.1	面向连接的通信.....	375	19.4	表的基本操作.....	414
18.8.2	面向非连接的通信.....	376	19.4.1	字段类型.....	414
18.9	socket 内建方法.....	376	19.4.2	约束.....	415
18.9.1	bind——绑定地址到套 接字.....	376	19.4.3	创建表.....	415
18.9.2	listen——服务器监听客户端 的请求.....	377	19.4.4	删除表.....	416
18.9.3	connect——建立与服务器 的连接.....	377	19.5	数据的操作.....	417
18.9.4	accept——接受客户端 的连接.....	377	19.5.1	数据的添加.....	417
18.9.5	send/sendto—— 发送数据.....	378	19.5.2	数据的更新.....	417
18.9.6	recv/recvfrom—— 接收数据.....	378	19.5.3	数据的删除.....	418
18.9.7	close——关闭套接字.....	379	19.5.4	事务.....	418
18.10	网络编程实例.....	379	19.6	单表查询.....	419
18.10.1	UDP 聊天室.....	379	19.6.1	数据查询.....	419
18.10.2	TCP 数据转换.....	382	19.6.2	最简单查询.....	420
18.11	TCP 并发服务器.....	384	19.6.3	条件查询.....	420
18.11.1	单进程非阻塞服务器.....	384	19.6.4	查询不重复的记录.....	424
18.11.2	多进程并发服务器.....	387	19.6.5	聚合查询.....	424
18.11.3	多线程并发服务器.....	389	19.6.6	分组查询.....	425
18.12	I/O 多路转接服务器.....	390	19.6.7	排序.....	426
			19.6.8	限制记录数量.....	426
			19.6.9	完整的 SQL 语句.....	427
			19.7	多表查询.....	427
			19.7.1	内连接 (inner join).....	428

19.7.2	外连接 (outer join)	428
19.8	MySQL 的内置函数	429
19.8.1	字符串函数	429
19.8.2	数学函数	431
19.8.3	日期时间函数	432
19.8.4	类型转换函数	434
19.9	MySQL 与 Python 交互	434
19.9.1	安装 PyMySQL	435
19.9.2	PyMySQL 的常用对象	435
19.9.3	PyMySQL 示例	436

## 第 20 章 Web 编程 ..... 438

20.1	什么是 Web 服务器	438
20.2	第一个 Web 服务器	438
20.3	统一资源定位符	439
20.4	HTTP 概述	440
20.5	HTTP 消息	440
20.5.1	请求结构	440
20.5.2	响应结构	441
20.5.3	HTTP 请求方法	441
20.5.4	状态码	442
20.6	HTML 概述	442
20.6.1	静态网页	443
20.6.2	CSS 简介	443
20.6.3	JavaScript 简介	444
20.7	静态 Web 服务器	445
20.8	WSGI 规范	448
20.9	WSGI 服务器	449
20.10	Web 服务器	452
20.11	基于框架的服务器	454
20.11.1	Web 应用框架	455
20.11.2	基于框架的服务器	456
20.11.3	流程分析	458
20.11.4	功能展示	459

## 第 21 章 井字棋 ..... 460

21.1	游戏简介	460
21.1.1	游戏规则	460
21.1.2	游戏策略	460
21.2	分析与设计	461
21.2.1	流程分析	462
21.2.2	类的设计	463
21.2.3	开发阶段设计	463
21.3	新建项目及文件准备	464
21.4	棋盘功能实现	464
21.4.1	棋盘的数据结构	465
21.4.2	初始化棋盘	466
21.4.3	显示棋盘	466
21.4.4	在棋盘上落子	467
21.4.5	判断平局	468
21.4.6	判断胜利	469
21.4.7	重置棋盘	469
21.5	玩家功能实现	470
21.5.1	初始化玩家	470
21.5.2	玩家在棋盘落子	471
21.6	人人对战功能实现	472
21.6.1	游戏的初始化	472
21.6.2	随机决定先手玩家	472
21.6.3	一轮完整对局	473
21.6.4	循环对局	474
21.7	人机对战——计算机随机落子	475
21.7.1	派生 AIPlayer 子类	475
21.7.2	修改游戏类中计算机玩家的类型	476
21.8	人机对战——计算机智能落子	476
21.8.1	计算机落子策略	476
21.8.2	预判胜利	476



21.8.3	必胜落子和必救落子.....	478	<b>第 23 章 飞机大战 .....</b>	<b>518</b>
21.8.4	评估子力价值 .....	478	23.1 游戏简介.....	518
21.9	项目体会.....	480	23.1.1 经典飞机大战 .....	518
<b>第 22 章 贪吃蛇 .....</b>	<b>481</b>		23.1.2 增强版飞机大战游戏 特点 .....	519
22.1 游戏简介.....	481		23.1.3 增强版飞机大战游戏 设计 .....	519
22.1.1 简易版贪吃蛇 .....	481		23.1.4 增强版飞机大战典型 场景 .....	522
22.1.2 游戏规则 .....	482		23.2 项目准备.....	524
22.2 类和模块的设计.....	483		23.2.1 类设计 .....	524
22.2.1 类的设计 .....	483		23.2.2 模块设计 .....	525
22.2.2 模块的设计 .....	484		23.2.3 创建项目和项目文件 结构 .....	526
22.3 游戏准备工作.....	484		23.3 游戏框架的设计与搭建.....	526
22.3.1 新建项目 .....	484		23.3.1 游戏类的设计 .....	526
22.3.2 安装 pygame 库 .....	485		23.3.2 游戏框架搭建 .....	528
22.4 游戏框架搭建.....	487		23.4 精灵和精灵组.....	530
22.4.1 pygame 初始化和退出.....	487		23.4.1 介绍 Sprite 和 Group.....	530
22.4.2 创建游戏主窗口 .....	488		23.4.2 派生游戏精灵子类.....	532
22.4.3 增加游戏循环和游戏 时钟 .....	489		23.4.3 绘制游戏背景和英雄 飞机 .....	533
22.4.4 填充窗口的背景颜色.....	490		23.4.4 实现游戏背景连续滚动.....	535
22.4.5 绘制得分文本 .....	492		23.5 指示器面板.....	537
22.4.6 重置得分文本的位置.....	494		23.5.1 指示器面板类的设计.....	538
22.4.7 游戏事件监听 .....	498		23.5.2 指示器面板类的准备.....	539
22.4.8 绘制食物图形 .....	501		23.5.3 使用精灵实现文本标签.....	542
22.4.9 定时更新食物的位置.....	504		23.5.4 显示和修改游戏数据.....	544
22.5 贪吃蛇功能实现.....	505		23.5.5 保存和显示最好成绩.....	548
22.5.1 贪吃蛇类的设计 .....	505		23.5.6 显示游戏状态 .....	550
22.5.2 定义贪吃蛇类 .....	507		23.5.7 游戏结束后重置面板.....	552
22.5.3 添加蛇身体 .....	508		23.6 逐帧动画和飞机类.....	553
22.5.4 绘制和移动身体 .....	510		23.6.1 逐帧动画的基本实现.....	554
22.5.5 改变贪吃蛇的方向.....	512		23.6.2 飞机类的设计与实现.....	557
22.5.6 贪吃蛇吃食物 .....	513			
22.5.7 贪吃蛇的死亡 .....	515			
22.6 项目体会.....	517			