工业和信息化普通高等教育“十三五”规划教材
21世纪高等教育计算机规划教材

Python 程序设计基础教程

Python Programming

王绍锋 李淑英 主编
李涛 曹琳琳 芦关山 副主编

- 针对全国计算机等级考试（二级）编写
- 案例丰富，涵盖多种开发场景
- 提供程序源代码、多媒体课件和习题答案

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

工业和信息化普通高等教育
21世纪高等教育计算机规划教材

Python 程序设计基础教程

Python Programming

王绍锋 李淑英 主编
李涛 曹琳琳 芦关山 副主编

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

Python程序设计基础教程 / 王绍锋，李淑英主编
. -- 北京 : 人民邮电出版社，2019.2
21世纪高等教育计算机规划教材
ISBN 978-7-115-50551-4

Ⅰ. ①P… Ⅱ. ①王… ②李… Ⅲ. ①软件工具－程序设计－高等学校－教材 Ⅳ. ①TP311.561

中国版本图书馆CIP数据核字(2018)第297684号

内 容 提 要

本书以全国计算机等级考试 Python 大纲为参考进行编写，共分为 10 章，内容包括 Python 概述、语法基础、程序控制结构、数据结构、函数与模块、面向对象程序设计、编程规范、错误和异常、文件操作及 Python 第三方库。

本书适合作为普通高等院校相关专业 Python 程序设计课程的教材和参考资料，也可作为全国计算机等级考试的培训用书。

◆ 主　　编　王绍锋　李淑英
副 主 编　李　涛　曹琳琳　芦关山
责任编辑　张　斌
责任印制　彭志环

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京圣夫亚美印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：9.5　　2019 年 2 月第 1 版
字数：228 千字　　2019 年 2 月北京第 1 次印刷

定价：39.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前言 FOREWORD

随着大数据、人工智能、机器学习的蓬勃发展，Python 编程语言得到了广泛应用。Python 是一种免费、开源的跨平台解释型高级编程语言，具有简单、易学习、开发效率高及调试运行方便等特点，深受广大编程爱好者的喜爱，被誉为最好的人工智能语言之一。

Python 近年来备受瞩目，因其具有丰富强大的库，所以用 Python 开发项目，编写的代码量少，代码简短，可读性强，便于团队协作开发。Python 开发人员可以把主要精力放在业务逻辑的设计与实现上。Python 应用领域主要包括 Web 开发、网络编程、爬虫开发、云计算开发、人工智能、自动化运维、金融分析、科学计算、游戏开发、桌面软件等。

本书基于 Python 的特点，本着简单明了的原则而编写。书中案例以图像化运行结果为特点，知识讲解采取循序渐进的方式，尽可能使学生学习的过程更平滑，从而提升学生的学习兴趣和编程能力。

本书由王绍锋、李淑英任主编，李涛、曹琳琳、芦关山任副主编，具体编写分工如下：第 1～3 章由王绍锋编写，第 4 章和第 10 章由李淑英编写，第 5 章和第 9 章由李涛编写，第 6～8 章由曹琳琳和芦关山编写，此外，崔金香、刘菊、周威、郑立平、卜伶俐也参与了本书编写和程序调试工作。本书的编写得到了哈尔滨远东理工学院领导的大力支持，在此一并表示衷心感谢。

本书的程序均在 Python 3.7.0 版本下调试运行。由于作者水平有限，且编写时间仓促，书中疏漏之处在所难免，恳请广大读者批评指正。本书作者的电子邮箱为 331382818@qq.com，欢迎读者来信交流。

编者

2018 年 10 月

目 录 CONTENTS

第 1 章 Python 概述 ····· 1

1.1 Python 语言简介 ····· 1
1.1.1 Python 的发展史 ····· 1
1.1.2 Python 的特点 ····· 2
1.1.3 Python 的应用领域 ····· 4
1.2 Python 开发环境 ····· 5
1.2.1 Python IDLE 简介 ····· 5
1.2.2 Python 开发环境安装 ····· 5
1.2.3 启动 Python ····· 7
1.2.4 运行 Python 程序 ····· 7
1.3 Python 其他开发环境 ····· 8
1.4 习题 ····· 9

第 2 章 语法基础 ····· 10

2.1 基本数据类型 ····· 10
2.1.1 常量和变量 ····· 10
2.1.2 数字类型 ····· 14
2.1.3 布尔类型 ····· 17
2.1.4 字符串 ····· 18
2.1.5 数据类型转换 ····· 20
2.2 运算符与表达式 ····· 21
2.2.1 算术运算符 ····· 21
2.2.2 关系运算符 ····· 22
2.2.3 逻辑运算符 ····· 23
2.2.4 位运算符 ····· 25
2.2.5 赋值运算符 ····· 26
2.2.6 成员运算符 ····· 29
2.2.7 身份运算符 ····· 30
2.2.8 运算符优先级 ····· 31
2.3 习题 ····· 31

第 3 章 程序控制结构 ····· 32

3.1 海龟绘图模块 ····· 32
3.2 顺序结构 ····· 33
3.3 选择结构 ····· 35
3.3.1 单分支选择结构 ····· 35
3.3.2 双分支选择结构 ····· 36
3.3.3 多分支选择结构 ····· 37
3.3.4 选择结构嵌套 ····· 40
3.3.5 pass 语句 ····· 41
3.4 循环结构 ····· 42
3.4.1 for 循环 ····· 42
3.4.2 while 循环 ····· 44
3.4.3 break 和 continue 语句 ····· 46
3.5 习题 ····· 48

第 4 章 数据结构 ····· 49

4.1 列表 ····· 49
4.1.1 列表基本操作 ····· 49
4.1.2 列表常用方法 ····· 52
4.2 元组 ····· 54
4.2.1 元组基本操作 ····· 54
4.2.2 元组与列表 ····· 55
4.3 字典 ····· 56
4.3.1 字典基本操作 ····· 56
4.3.2 字典常用方法 ····· 59
4.4 集合 ····· 61
4.4.1 集合基本操作 ····· 61
4.4.2 集合运算 ····· 62
4.5 字符串 ····· 64
4.6 习题 ····· 67

第 5 章 函数与模块 ····· 68

5.1 函数定义与使用 ····· 68
5.2 函数的参数 ····· 69
5.2.1 必选参数 ····· 70
5.2.2 默认参数 ····· 70
5.2.3 可变参数 ····· 72
5.2.4 关键字参数 ····· 73

5.2.5 参数组合……73
5.3 函数的返回值……75
5.4 变量作用域……76
5.5 函数的嵌套……78
5.6 lambda 表达式……79
5.7 常用内置函数……80
5.8 模块……84
5.8.1 模块的使用……84
5.8.2 数学模块……85
5.8.3 随机模块……85
5.8.4 时间模块……86
5.9 习题……86

第 6 章 面向对象程序设计……87

6.1 面向对象程序设计简介……87
6.1.1 面向过程与面向对象……87
6.1.2 面向对象的主要特性……88
6.2 类的定义和实例化……88
6.3 数据成员与成员方法……90
6.3.1 私有成员与公有成员……90
6.3.2 数据成员……91
6.3.3 方法……93
6.4 属性……96
6.5 继承……101
6.5.1 类的简单继承……101
6.5.2 类的多重继承……102
6.6 多态……104
6.7 特殊方法和运算符重载……105
6.7.1 构造函数和析构函数……105
6.7.2 运算符重载……106
6.8 习题……107

第 7 章 编程规范……108

7.1 代码规范……108
7.2 注释规范……112
7.2.1 代码注释……112
7.2.2 文档注释……113
7.3 命名规范……114
7.4 习题……115

第 8 章 错误和异常……116

8.1 语法错误……116
8.2 异常……116
8.3 异常处理……118
8.4 抛出异常……121
8.5 用户自定义异常……122
8.6 定义清理行为……123
8.7 预定义清理行为……124
8.8 习题……125

第 9 章 文件操作……126

9.1 文件基础知识……126
9.2 文件基本操作……127
9.2.1 打开文件……127
9.2.2 关闭文件……129
9.3 文件读写操作……130
9.3.1 文件的读操作……130
9.3.2 文件的写操作……132
9.4 文件与目录操作……133
9.4.1 os……133
9.4.2 os.path……135
9.4.3 os.walk……135
9.5 数据维度……136
9.5.1 一维数据……136
9.5.2 二维数据……138
9.6 习题……139

第 10 章 Python 第三方库…140

10.1 第三方库的安装……140
10.1.1 第三方库的安装方法……140
10.1.2 pip 工具使用……141
10.2 PyInstaller 库……142
10.3 jieba 库……143
10.4 wordcloud 库……144
10.5 Python 常用第三方库……145
10.6 习题……146

第1章 Python概述

Python 作为一门跨平台、开源、免费的解释型高级编程语言，得到了业界的广泛关注。本章将对 Python 语言进行简单介绍，包括 Python 的发展史、Python 的应用领域和 Python 开发环境等内容。

1.1 Python 语言简介

Python 是一种被广泛使用的优秀的编程语言，崇尚优美、清晰、简单。据统计，近年来 Python 的影响逐年扩大，2018 年 7 月的 TIOBE 排行榜显示，Python 已经在编程语言中排行第 4（见图 1-1），而且整体呈上升趋势，反映出 Python 应用越来越广泛，也越来越得到业内的认可。

Jul 2018	Jul 2017	Change	Programming Language	Ratings	Change
1	1		Java	16.139%	+2.37%
2	2		C	14.662%	+7.34%
3	3		C++	7.615%	+2.04%
4	4		Python	6.361%	+2.82%
5	7	^	Visual Basic .NET	4.247%	+1.20%
6	5	v	C#	3.795%	+0.28%
7	6	v	PHP	2.832%	-0.26%
8	8		JavaScript	2.831%	+0.22%
9	-	^	SQL	2.334%	+2.33%
10	18	^	Objective-C	1.453%	-0.44%

图 1-1 2018 年 7 月的 TIOBE 排行榜

1.1.1 Python 的发展史

Python 语言的创始人是吉多·范罗苏姆（Guido van Rossum）。1989 年，为了打发圣诞节假期，吉多·范罗苏姆开始开发一个新的脚本解释程序，作为 ABC 语言的一种继承，也就是 Python 语言的编译器。Python 这个名字，来自吉多所挚爱的电视剧 *Monty Python's Flying Circus*。吉多希望这个叫作 Python 的语言能符合他的理想：创造一种处于 C 和 Shell 之间、功能全面、易学易用、可拓展的语言。

1991 年，第一个 Python 编译器诞生。它是用 C 语言实现的，并能够调用 C 语言的库文件。从诞生开始，Python 就已经具有了类、函数、异常处理、包含列表和词典在内的核心数据类型，是以模块为基础的拓展系统。

2000 年 10 月 16 日，Python 2.0 发布，实现了完整的垃圾回收，并且支持 Unicode。同时，整个开发过程更加透明，在社区的影响也逐渐扩大。

2008 年 12 月 3 日，Python 3.0 发布，此版本不完全兼容之前的 Python 代码，不过，很多新特征后来也被移植到了 Python 2.x 版本。目前，Python 最新版本为 3.7，其下载界面如图 1-2 所示。

图 1-2　Python 3.7 版本下载界面

1.1.2　Python 的特点

Python 作为一门高级编程语言，它的诞生虽然很偶然，但是它得到程序员的喜爱却是必然的。

Python 的定位是“优雅”“明确”“简单”，所以 Python 程序看上去总是简单易懂，初学者学习 Python，不但入门容易，而且将来深入下去，可以编写一些功能非常复杂的程序。

1. Python 的优点

（1）简单

作为初学 Python 的人员，直接的感觉就是 Python 非常简单，非常适合阅读。阅读一个良好的 Python 程序就感觉像是在读英语文章一样，尽管这个“英语文章”的要求非常严格。Python 的这种伪代码本质是它最大的优点之一。它使你能够专注于解决问题而不是去搞明白语言本身。

（2）易学

Python 虽然是用 C 语言写的，但是它摈弃了 C 语言中非常复杂的指针，简化了 Python 的语法结构。

（3）免费开源

Python 是 FLOSS（自由/开放源码软件）之一。简单地说，用户可以自由地发布这个软件的备份、阅读它的源代码、对它做改动、把它的一部分用于新的自由软件中。Python 的开发者希望 Python 能得到更多优秀的人参与创造并经常改进。

（4）移植性强

由于 Python 具有开源的本质，它已经被移植到许多平台上（它经过改动能够工作在不同平台上）。如果开发者能小心地避免使用 Python 依赖于系统的特性，那么几乎所有 Python 程序无需修改就可以在下述任何平台上面运行，包括 Linux、Windows、FreeBSD、Macintosh、Solaris、OS/2、

Amiga、AROS、AS/400、BeOS、OS/390、z/OS、Palm OS、QNX、VMS、Psion、Acom RISC OS、VxWorks、PlayStation、Sharp Zaurus、Windows CE，甚至还有PocketPC、Symbian以及Google基于Linux开发的Android平台。

（5）解释性编程语言

在计算机内部，Python解释器把源代码转换成称为字节码的中间形式，然后再把它翻译成计算机使用的机器语言并运行。事实上，由于用户不再需要担心如何编译程序、如何确保连接转载正确的库等，所以这一切使应用Python更加简单。而且，Python程序直接复制到另外一台计算机上就可以工作，这也使Python程序更加易于移植。

（6）面向对象性

Python既支持面向过程的函数编程，也支持面向对象的抽象编程。在面向过程的语言中，程序是由过程或仅仅是可重用代码的函数构建起来的；在面向对象的语言中，程序是由数据和功能组合而成的对象构建起来的。与其他主要的语言（如C++和Java）相比，Python以一种非常强大又简单的方式实现面向对象编程。

（7）可扩展性和可嵌入性

如果需要一段关键代码运行得更快或者希望某些算法不公开，用户可以把部分程序用C或C++编写，然后在Python程序中使用它们。也可以把Python嵌入C/C++程序，从而向使用程序的用户提供脚本功能。

（8）丰富的库

Python有丰富的标准库和第三方库可以使用。它可以帮助用户处理各种工作，包括正则表达式、文档生成、单元测试、线程、数据库、网页浏览器、CGI、FTP、电子邮件、XML、XML-RPC、HTML、WAV文件、密码系统、GUI（图形用户界面）、Tk和其他与系统有关的操作。只要安装了Python，以上所有这些功能都是可用的。这被称作Python的“功能齐全”理念。除了标准库以外，Python还有许多其他高质量的库，如wxPython、Twisted和Python图像库等。

（9）功能强大

Python确实是一种十分精彩而又强大的语言，它合理地结合了高性能与编写程序简单有趣的特色。

（10）规范的代码

Python采用强制缩进的方式使代码具有极佳的可读性。

2. Python的缺点

（1）运行速度慢

如果用户有速度要求的话，可以用C++改写关键部分，以提高运行速度。不过对一般用户而言，机器上运行速度的因素是可以忽略的，因为用户几乎感觉不到这种速度的差异。

（2）不能加密

不能加密既是优点也是缺点。Python的开源性使Python语言不能加密，但是目前国内市场纯粹靠编写软件卖给客户的情况越来越少，网站和移动应用不需要给客户源代码，所以这个问题也就不算是问题了。

（3）构架选择太多

Python 没有像 C#这样的官方.NET 构架，也没有像 Ruby 开发的相对集中的构架（Ruby on Rails 构架开发中小型 Web 程序首选）。不过这也从另一个侧面说明，Python 比较优秀，吸引的开发人才多，项目也多。

1.1.3 Python 的应用领域

Python 作为一个整体可以用于任何软件开发领域，下面介绍 Python 主要应用的领域。

1. Web 开发

目前最流行的 Python Web 框架 Django，支持异步高并发的 Tornado 框架，短小精悍的 Flask 和 Bottle。Django 官方的标语把 Django 定义为 the framework for perfectionist with deadlines（为完美主义者开发的高效率框架）。

2. 网络编程

Python 支持高并发的 Twisted 网络框架，Python 3 引入的 asyncio 使异步编程变得非常简单。

3. 网络爬虫

在爬虫领域，Python 几乎是霸主地位，包括 Scrapy、Request、BeautifuSoap、urllib 等，用户需要爬取什么内容几乎都可以爬取到。

4. 云计算

目前最流行、最知名的云计算框架是 OpenStack，它正是由 Python 开发的。Python 现在的流行，很大一部分原因就是云计算的发展。

5. 人工智能

谁会成为 AI 和大数据时代的第一开发语言？这本已是一个不需要争论的问题。如果说三年前，Matlab、Scala、R、Java 和 Python 还各有机会，局面尚且不清楚，那么在 Facebook 开源了 PyTorch 之后，Python 作为 AI 时代头牌语言的位置基本确立，未来的悬念仅仅是谁能坐稳第二把交椅。

6. 自动化运维

如果问问运维人员，运维人员必须掌握的语言是什么？绝大多数的人会给出相同的答案——Python。

7. 金融分析

目前，Python 是金融分析、量化交易领域里使用最多的开发语言。

8. 科学运算

从 1997 年开始，美国国家航空航天局（National Aeronautics and Space Administration，NASA）就大量使用 Python 进行各种复杂的科学运算，随着 NumPy、SciPy、Matplotlib 和 Enthought librarys 等众多程序库的开发，使 Python 越来越适合于做科学计算、绘制高质量的 2D 和 3D 图像。与科学计算领域最流行的商业软件 Matlab 相比，Python 是一门通用的程序设计语言，比 Matlab 所采用的脚本语言的应用范围更广泛。

9. 游戏开发

Python在网络游戏开发中也有很多应用。Python比Lua有更高阶的抽象能力，可以用更少的代码描述游戏业务逻辑，与Lua相比，Python更适合作为一种Host语言，即程序的入口点在Python那一端会比较好，然后用C/C++在非常必要的时候写一些扩展。Python非常适合编写1万行以上的项目，而且能够很好地把网游项目的规模控制在10万行代码以内。

1.2 Python开发环境

1.2.1 Python IDLE简介

IDLE是开发Python程序的基本IDE（集成开发环境），具备基本的IDE的功能，是非商业Python开发的不错的选择。当安装好Python以后，IDLE就自动安装好了，不需要另外安装。同时，使用Eclipse这个强大的框架时，IDLE也可以非常方便地调试Python程序。IDLE包括语法加亮、段落缩进、基本文本编辑、TABLE键控制和调试程序等基本功能。

IDLE是标准的Python发行版，甚至是由创始人吉多亲自编写（至少最初的绝大部分）的，开发者可以在能运行Python和Tk的任何环境下运行IDLE。打开IDLE后出现一个增强的交互命令行解释器窗口（具有比基本的交互命令提示符更好的复制、粘贴和回行等功能）。除此之外，IDLE还有一个针对Python的编辑器（无代码合并，但有语法标签高亮和代码自动完成功能）、类浏览器和调试器。菜单为Tk“剥离”式，也就是单击顶部任意下拉菜单的虚线会将该菜单提升到它自己的永久窗口中去。特别是“Edit”菜单，将其“停靠”在桌面一角非常实用。IDLE的调试器提供断点、步进和变量监视功能，以及内存地址和变量内存数或进行同步和其他分析功能等一些更受用户欢迎的功能。Python 3.7.0 IDLE界面如图1-3所示。

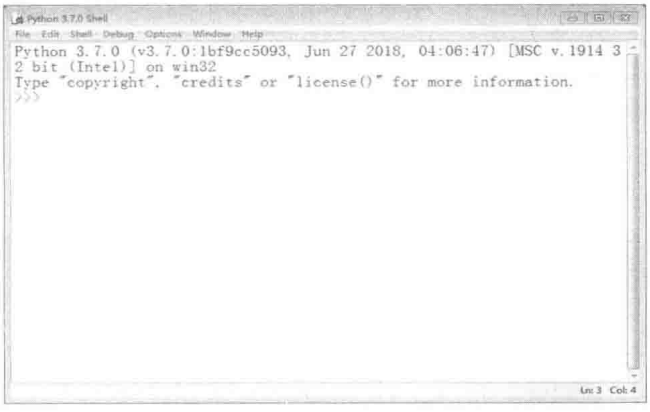

图1-3 Python 3.7.0 IDLE界面

1.2.2 Python开发环境安装

学习Python首先需要安装开发环境。安装后会得到Python解释器，它负责运行Python程序。Python可以在命令行交互环境下或简单的集成开发环境下运行。

目前，Python有两个版本，分别是2.x版本和3.x版本，这两个版本并不兼容。由于3.x版本

越来越普及，本书以最新的 Python 3.7 版本为基础。

安装前要确定 Windows 操作系统的版本（32 位或 64 位），然后从 Python 官网下载对应的 Python 安装程序并安装，安装界面如图 1-4 所示。

图 1-4　Python 安装界面

安装前要注意把“Add Python 3.7 to PATH”选上，这样省去了手动配置环境变量的麻烦。选中后单击“Install Now”按钮开始默认安装，安装的过程界面如图 1-5 所示。

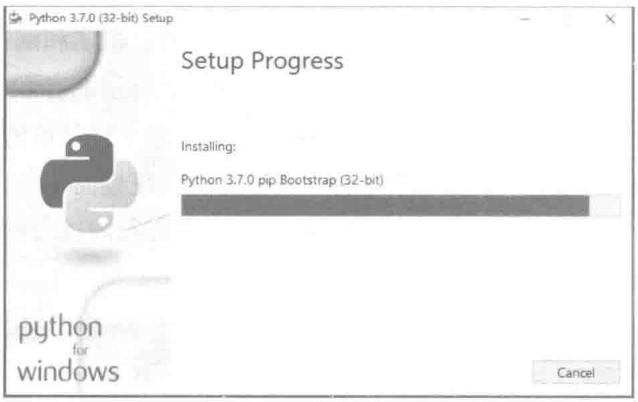

图 1-5　Python 安装过程界面

安装程序会自动安装，直到程序安装完成，Python 安装成功界面如图 1-6 所示。

图 1-6　Python 安装成功界面

1.2.3 启动 Python

Python 安装完成后，通过 cmd 打开命令提示符窗口，输入“Python”后回车，出现图 1-7 所示的界面，表明开发环境安装配置成功。

图 1-7 测试 Python 安装及配置是否成功

假如得到一个错误：'Python' 不是内部或外部命令，也不是可运行的程序或批处理文件。这是因为 Windows 会根据 Path 环境变量设定的路径去查找 Python.exe，如果没找到就会报错，这也是为什么安装时强调把“Add Python 3.7 to PATH”选项选上的原因，选上后安装程序自动为我们配置了 Python 运行所需要的环境变量。

1.2.4 运行 Python 程序

1. 交互式编程

交互式编程不需要创建脚本文件，是通过 Python 解释器的交互模式来编写代码的。在 Windows 操作系统中，打开默认的交互式 IDE - IDLE。

进入交互式环境后，输入以下文本信息，然后按 Enter 键，运行效果如下所示。

```
>>> print("Hello World!")
Hello World!
>>>
```

2. 脚本式编程

通过脚本参数调用解释器开始执行脚本，直到脚本执行完毕。当脚本执行完成后，解释器不再有效。

下面写一个简单的 Python 脚本程序，所有 Python 文件将以.py 为扩展名。将以下源代码输入 firstproc.py 文件中。

```
print("Hello World!")
```

打开 cmd 命令提示符，切换至 firstproc.py 所在目录（例如：E:\），使用以下命令执行脚本：

```
Python firstproc.py
```

运行结果如图 1-8 所示。

图 1-8　firstproc.py 运行结果

3. Python IDLE

（1）新建文件

单击“File”→“New File”打开一个新的窗口，并输入程序。

```
print("Hello World!")
```

（2）保存程序

在 IDLE 编写完程序后，在菜单里依次选择“File”→“Save”（或者用 Ctrl+S 组合键）来进行保存，首次保存时会弹出文件对话框，要求用户输入保存的文件名。此时保存的文件名为 firstproc.py。

（3）运行程序

文件编辑完成后，可以按 F5 键运行程序，或单击“Run”→“Run Module”菜单项。

1.3　Python 其他开发环境

1. Anaconda 简介

Anaconda 是一个用于科学计算的 Python 发行版，支持 Linux、Mac、Windows 系统，包含了众多流行的科学计算、数据分析的 Python 包。此外，Anaconda 提供了包管理与环境管理的功能，可以很方便地解决多版本 Python 并存、切换以及各种第三方包的安装问题。Anaconda 利用工具/命令 conda 来进行 package 和 environment 的管理，并且已经包含了 Python 及其相关的配套工具。

与其说 Anaconda 是一个 IDE，还不如说它是一个 Python 环境。Anaconda 中包含 Numpy、Pandas、Matplotlib 等库，所以说利用 Anaconda 可以避免让用户将过多的精力花在环境搭建上，从而快速进入 Python、数据分析、机器学习等领域的探索当中。

2. Eclipse+PyDev

Eclipse是一款基于Java的可扩展开发平台，其官方下载中包括Java SE、Java EE、Java ME等诸多版本。除此之外，Eclipse还可以通过安装插件的方式进行诸如Python、Android、PHP等语言的开发。

PyDev是一个功能强大的Eclipse插件，用户可以完全利用Eclipse来进行Python应用程序的开发和调试。它能够将Eclipse当作Python IDE。PyDev插件的出现方便了众多的Python开发人员，它提供了一些很好的功能，如：语法错误提示、源代码编辑助手、Quick Outline、Globals Browser、Hierarchy View、运行和调试等。PyDev基于Eclipse平台，拥有诸多强大的功能，同时也非常易于使用，它的这些特性使其越来越受到人们的关注。

Eclipse+PyDev对IDLE进行了封装，提供了强大的功能，非常适合开发大型项目。

1.4 习题

1. 简述Python语言的特点。
2. 简述Python语言的应用领域。
3. 编写一个简单的Python程序，输出本书的相关信息（包括书名、出版社、作者等信息）。

02 第2章 语法基础

Python 作为一门编程语言，开发过程中离不开常量和变量定义、数据类型及数据类型转换、运算符和表达式等内容，本章将从 Python 的基本数据类型、运算符和表达式等方面讲解 Python 的基本语法。

2.1 基本数据类型

2.1.1 常量和变量

常量一般是指不需要改变也不能改变的字面值，常量一旦初始化之后就不能修改。例如：数字 5、字符串 abc 都是常量。Python 中并没有提供定义常量的保留字。

变量是计算机内存中的一块区域，变量可以存储规定范围内的值，而且值可以改变。基于变量的数据类型，解释器会分配指定内存，并决定什么数据可以被存储在内存中。与变量对比，常量是一块只读的内存区域，所以，常量一旦被初始化就不能被改变。

Python 中的变量不需要声明，变量的赋值操作即变量的声明和定义的过程。每个变量在内存中创建都包括变量的标识、名称和数据等信息，如图 2-1 所示。

图 2-1 变量的内部结构

变量操作可参见例 2-1。

【例 2-1】变量操作。具体代码如下。

程序代码：

```
# 例 2-1 变量操作
num = 1
# 输出变量 num 的值
print("num = " + str(num))
# 输出表达式（num + 1）的值
print("num + 1 = " + str(num + 1))
```

运行结果：

```
num = 1
num + 1 = 2
```

注：（1）#为行注释符号，#后的内容为代码注释，方便读者阅读和理解代码。

（2）在 print("num = " + str(x))代码语句中，+为字符串连接符，str()函数一般把数值转换为字符串。

上例中有一条赋值语句 num = 1，在 Python 中一次新的赋值，将创建一个新的变量。即使变量的名称相同，变量的标识（变量的内存地址）并不同。

变量赋值可参见例 2-2。

【例 2-2】变量赋值。具体代码如下。

程序代码：

```
# 例 2-2 变量赋值
print("========第一次赋值========")
# 变量 num 第一次赋值
num = 1
# 输出变量 num 的值
print("num = " + str(num))
# 打印变量 num 的标识（地址）
print("id(num) = " + str(id(num)))

print("========第二次赋值========")
# 变量 num 再次赋值，定义一个新变量 num
num = 2
# 输出变量 num 的值
print("num = " + str(num))
# 此时的变量 num 已经是一个新的变量
print("id(num) = " + str(id(num)))
```

运行结果：

```
========第一次赋值========
num = 1
id(num) = 2013018368
========第二次赋值========
num = 2
id(num) = 2013018384
```

注：id(object)，返回对象 object 的内存地址。

变量不仅能重新赋相同类型的值，还可以赋新类型的值。把上例中的变量 num 赋值为字符串，此时 num 又将成为一个新的变量，而且变量类型也由于所赋值的数据类型改变而改变。

【例 2-3】变量赋值不同类型。具体代码如下。

程序代码：

```
# 例 2-3 变量赋值不同类型
print("========第一次赋值========")

# 变量 var 第一次赋值
var = 1
```