

巍巍交大 百年书香
www.jiaodapress.com.cn
bookinfo@sjtu.edu.cn



策划编辑 杨 洋
责任编辑 胡思佳
封面设计 刘文东



Python程序设计 及上机实验

Python CHENGXU SHEJI JI SHANGJI SHIYAN



扫描二维码
关注上海交通大学出版社
官方微信

ISBN 978-7-313-31909-8



定价:65.00元

程序设计系列精品教材

Python程序设计及上机实验

主编
杨 薇
杨天晴



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

程序设计系列精品教材



Python程序设计 及上机实验

主编 杨 薇 杨天晴



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

免费提供

精品教学资料包

服务热线: 400-615-1233
www.huatengzy.com

程序设计系列精品教材



Python程序设计 及上机实验

主 编 杨 薇 杨天晴
副主编 彭 革 钟雨静 莫淑怡
魏再超 罗 昊



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

内容提要

本书以 Windows 操作系统为平台,系统讲解 Python 的基础知识。全书共分 12 章,主要包括Python入门概述、Python 语言基础、Python 数据类型、Python 字符串类型、Python 序列结构、Python 流程控制、Python 函数、面向对象编程、模块、错误和异常处理、Python 中的文件操作、常用的标准库和第三方库。

本书既可作为高等院校各专业 Python 程序设计基础课程的教材,也可作为 Python 入门的培训教材和学习参考书。

图书在版编目(CIP)数据

Python 程序设计及上机实验 / 杨薇, 杨天晴主编.

上海 : 上海交通大学出版社, 2024. 12. (2025. 2 重印) -- ISBN 978-7-313-31909-8

I . TP312.8

中国国家版本馆 CIP 数据核字第 2024J0B549 号

Python 程序设计及上机实验

Python CHENGXU SHEJI JI SHANGJI SHIYAN

主 编:杨 薇 杨天晴

出版发行:上海交通大学出版社

邮政编码:200030

印 制:三河市骏杰印刷有限公司

开 本:850 mm×1 168 mm 1/16

字 数:517 千字

版 次:2024 年 12 月第 1 版

书 号:ISBN 978-7-313-31909-8

定 价:65.00 元

地 址:上海市番禺路 951 号

电 话:021-64071208

经 销:全国新华书店

印 张:16.75

印 次:2025 年 2 月第 2 次印刷

电子书号:ISBN 978-7-89564-197-6

版权所有 侵权必究

告读者:如发现本书有印装质量问题请与印刷厂质量科联系

联系电话:0316-3662258

前言

PREFACE

Python 是一门免费、开源的跨平台高级动态编程语言,支持命令式编程、函数式编程,完全支持面向对象程序设计,语法简洁清晰,并且拥有大量功能强大的标准库和扩展库以及众多支持者,可以帮助各领域的科研人员或策划师,甚至管理人员快速实现和验证自己的思路与创意。Python 用户可以把主要精力放在业务逻辑的设计与实现上,而不用过多考虑语言本身的细节,开发效率非常高。

Python 是一门快乐的语言,学习和使用 Python 也是一个快乐的过程。与 C 语言系列和 Java 语言等相比,学习和使用 Python 更加容易,但用户若想熟练掌握和运用 Python,则仍需要通过大量的练习来锻炼自己的思维和熟悉 Python 编程模式,同时还需要关注 Python 社区的优秀代码及各种扩展库的最新动态。当然,如果能够适当了解 Python 标准库及扩展库的内部工作原理,对于编写正确而优雅的 Python 程序无疑会有很大的帮助。

Python 语法简洁清晰,并且提供了大量的内置对象和内置函数,编程模式非常符合人类的思维方式和习惯。在有些编程语言中需要编写大量代码才能实现的功能,在 Python 中仅需要调用内置函数或内置对象即可实现。如果用户已有其他程序设计语言的基础,那么在学习和使用 Python 时,一定不要把其他语言的编程习惯和风格带到 Python 中来,因为这不仅可能会使代码变得冗余、烦琐,还可能会严重影响代码的运行效率。用户应该尝试从最自然、最简洁的角度出发来思考和解决问题,这样才能编写出更加优雅、更加纯正、更具 Python 特色的代码。

本书的编写原则如下。

(1)适应原则。Python 语言有自己独特的语法及编程方式,与传统的 Java 语言、C 语言等有不同之处,编者试着从一个软件开发者的角度在编程语言的大框架下分析这些编程语言的细节差异,以期使学生能够很好地适应 Python 的学习节奏。

(2)科学原则。本书既是知识产品的再生产、再创造,也是编者教学经验的总结和提高。其覆盖范围广、内容新,既有面的铺开,又有点的深化,举例符合题意,使学生学习起来可以达到事半功倍的效果。

(3)实用原则。本书采用的是 Python 3 版本,准确、及时地反映了这门语言发展的最新成果及趋势,使学生能够很好地学到前沿的新技术。

本书从基础和实践两个层面引导学生学习 Python 这门学科,系统、全面地讨论了 Python 编程的思想和方法。第 1 章至第 5 章主要介绍了 Python 的基础知识;第 6 章、第 7 章详细介绍了 Python 编程的核心技术,着眼于控制语句与函数的使用,每一章都搭配

了详细的 Python 程序,让学生可以全面理解 Python 编程;第 8 章是程序开发的进阶,着重介绍了类、继承、封装、多态等知识点,并针对每一个知识点给出了详细的例子;第 9 章介绍了模块的操作;第 10 章具体介绍了开发语言中的调试及异常处理;第 11 章介绍了文件和对象序列化的重点知识;第 12 章介绍了标准库和第三方库的基本使用方法。

本书由杨薇、杨天晴担任主编,彭革、钟雨静、莫淑怡、魏再超和罗昊担任副主编。

为提升学习效果,书中结合实际应用提供了大量案例进行说明,并配以完善的学习资料,包括教学 PPT、案例素材、源码等,为学生带来全方位的学习体验。

编 者

目录

CONTENTS

第 1 章 Python 入门概述 1

1.1 Python 语言简介..... 1

- 1.1.1 什么是 Python 语言 1
- 1.1.2 Python 语言的特点 2
- 1.1.3 Python 语言的应用 3
- 1.1.4 Python 语言的发展前景 4
- 1.1.5 Python 语言的版本 5
- 1.1.6 Python 语言的集成开发环境 5

1.2 Python 环境的准备..... 6

- 1.2.1 安装 Python 6
- 1.2.2 使用交互式执行环境 9
- 1.2.3 使用 IDLE 编写代码 9
- 1.2.4 安装和管理 Python 第三方库 10
- 1.2.5 安装 Anaconda 13
- 1.2.6 使用 Spyder 编写并运行一个程序 15
- 1.2.7 使用 Jupyter Notebook 编写并运行一个程序 15
- 1.2.8 在 Anaconda 中安装和管理第三方库 16

第 2 章 Python 语言基础 18

2.1 Python 程序概述 18

- 2.1.1 引例 18
- 2.1.2 Python 程序的构成 19

2.2 Python 规范 19

- 2.2.1 语句规则 19
- 2.2.2 缩进规则 20
- 2.2.3 注释规则 21
- 2.2.4 标识符及其命名规则 23
- 2.2.5 关键字 25

2.3 数据类型 27

2.4 变量 27

- 2.4.1 对象 27
- 2.4.2 变量的创建 29
- 2.4.3 变量的删除 30

2.5 表达式和运算符 31

- 2.5.1 表达式的组成 31
- 2.5.2 表达式的书写规则 31
- 2.5.3 运算符概述 32
- 2.5.4 算术运算符 32
- 2.5.5 比较运算符 34
- 2.5.6 逻辑运算符 35
- 2.5.7 成员运算符 37
- 2.5.8 身份运算符 38
- 2.5.9 位运算符 38
- 2.5.10 赋值运算符 40

2.6 基本输入、输出 41

- 2.6.1 使用 input() 函数输入 41
- 2.6.2 使用 print() 函数输出 41

2.7 常用函数 42

- 2.7.1 内置函数 42
- 2.7.2 模块函数 46
- 2.7.3 内置标准数学函数和 math 库函数 47



2.8 案例实战	50
----------------	----

第 3 章 Python 数据类型 54

3.1 数字类型	54
----------------	----

3.1.1 整型	54
3.1.2 浮点型	56
3.1.3 布尔型	57
3.1.4 复数型	57

3.2 数字类型之间的转换	57
---------------------	----

3.2.1 隐式转换	57
3.2.2 显式转换	58

第 4 章 Python 字符串类型 60

4.1 字符串的编码方式	60
--------------------	----

4.2 字符串的表示形式	62
--------------------	----

4.3 特殊字符和字符转义	63
---------------------	----

4.4 原始字符串	64
-----------------	----

4.5 字符串的索引和切片	65
---------------------	----

4.5.1 字符串的索引	65
4.5.2 字符串的切片	66

4.6 字符串的拼接	68
------------------	----

4.7 字符串的基本操作符	69
---------------------	----



4.8 字符串处理的常用操作 70

4.8.1 表达式转换	70
4.8.2 字符串长度计算	70
4.8.3 字符串的类型判断	71
4.8.4 字符串的大小写转换	72
4.8.5 字符串的填充、空白和对齐	73
4.8.6 字符串的测试、查找和替换	74
4.8.7 字符串的拆分和组合	77
4.8.8 字符串常量	78

4.9 字符串的格式化 78

4.9.1 格式化表达式	79
4.9.2 format()函数	80

4.10 Python 的内置字符串函数..... 81

4.11 案例实战 83

第 5 章 Python 序列结构 87

5.1 序列概述 87

5.2 列表 88

5.2.1 列表的创建和删除	88
5.2.2 列表的赋值、浅复制和深复制	90
5.2.3 列表元素的访问	93
5.2.4 列表元素的添加、删除和修改	95
5.2.5 对列表进行统计	99
5.2.6 对列表进行排序	100
5.2.7 列表的比较	102
5.2.8 成员资格判断	102
5.2.9 切片操作	103
5.2.10 列表推导式	105
5.2.11 常用列表对象函数	106
5.2.12 列表综合案例	106

**5.3 元组 107**

5.3.1 元组的创建和删除	107
5.3.2 访问元组	109
5.3.3 元组的长度、最大值、最小值及求和	109
5.3.4 修改元组	110
5.3.5 元组推导式	110
5.3.6 元组与列表的区别	111
5.3.7 序列封包和序列解包	111
5.3.8 常用元组对象函数	112

5.4 字典 112

5.4.1 字典的创建和删除	113
5.4.2 字典的访问	115
5.4.3 添加和修改字典元素	117
5.4.4 字典元素的排序	119
5.4.5 字典推导式	119
5.4.6 常用字典对象函数	120

5.5 集合 120

5.5.1 集合的创建和删除	121
5.5.2 集合元素的添加与删除	121
5.5.3 集合的并集、交集、差集与对称差集操作	123
5.5.4 集合比较	123
5.5.5 常用集合对象的函数	124

5.6 案例实战 125**第6章 Python 流程控制 129****6.1 程序控制结构 129**

6.1.1 Python 程序的三种控制结构	129
6.1.2 程序流程图	130
6.1.3 条件表达式	130



6.2	顺序结构	131
-----	------------	-----

6.3	选择结构	132
-----	------------	-----

6.4	循环结构	136
-----	------------	-----

6.5	跳转语句	141
-----	------------	-----

6.6	案例实战	143
-----	------------	-----

第 7 章 Python 函数 145

7.1	函数概述	145
-----	------------	-----

7.1.1	函数的基本概念	145
-------	---------------	-----

7.1.2	函数的功能	145
-------	-------------	-----

7.1.3	Python 中函数的分类	146
-------	---------------------	-----

7.2	函数的定义和函数的参数	146
-----	-------------------	-----

7.2.1	无参函数	147
-------	------------	-----

7.2.2	有参函数	147
-------	------------	-----

7.3	函数的返回值	152
-----	--------------	-----

7.4	函数的调用	155
-----	-------------	-----

7.5	匿名函数	157
-----	------------	-----

7.6	函数中的装饰器	158
-----	---------------	-----

7.7	函数中变量的作用域	159
-----	-----------------	-----

7.8	函数的递归	160
-----	-------------	-----

7.9	案例实战	162
-----	------------	-----

第 8 章 面向对象编程 166**8.1 面向对象编程概述 166****8.2 面向对象的概念 167****8.3 Python 中的面向对象 167****8.4 自定义类 169**

8.4.1 类的定义 169

8.4.2 使用类创建实例对象 170

8.4.3 为类添加方法 170

8.4.4 为类添加属性 174

8.5 继承 183

8.5.1 继承的概念 183

8.5.2 子类中的变量和方法 186

8.5.3 多继承 187

8.6 封装 188**8.7 多态 189**

8.7.1 方法重写 190

8.7.2 实现多态 190

8.8 案例实战 191**第 9 章 模块 197****9.1 模块化程序设计 197**

9.1.1 模块化程序设计的定义 197

9.1.2 模块的 API 197

9.1.3 模块化程序设计的优越性 198

9.2 模块的设计和实现 198

9.2.1 模块设计的一般原则 198

9.2.2 API 设计 198

9.2.3 模块的创建 198

9.3 模块的导入和使用 199

9.4 包 200

9.4.1 包的概念 200

9.4.2 创建包 200

9.4.3 包的导入和使用 200

第 10 章 错误和异常处理 203

10.1 程序的错误 203

10.2 程序的基本调试方法 204

10.3 异常 204

10.3.1 异常的概念 204

10.3.2 异常的角色 204

10.3.3 内置的异常类 205

10.3.4 捕获指定异常 205

10.3.5 捕获多个异常 206

10.3.6 未捕获到异常 208

10.3.7 try...except...finally 语句 209

10.3.8 自定义异常和抛出异常 209

10.4 断言处理 210

10.5 案例实战 211

第 11 章 Python 中的文件操作 215

11.1 文件操作相关模块概述 215

11.2 字符流和字节流的概念 216



11.3	文本文件的写入与读取	216
11.3.1	文本文件的写入	217
11.3.2	文本文件的读取	219
11.4	二进制文件的写入与读取	221
11.4.1	二进制文件的写入	221
11.4.2	二进制文件的读取	222
11.5	文件对象的常用方法	222
11.6	CSV 文件的写入与读取	223
11.6.1	CSV 文件的写入	223
11.6.2	CSV 文件的读取	224
11.7	对象序列化	225
11.7.1	对象序列化的概念	225
11.7.2	pickle 模块和对象序列化	225
11.7.3	json 模块和 JSON 格式数据	226
11.8	文件的几个常规操作	226
11.9	Python 对文件目录的操作	228
11.10	案例实战	229
第 12 章	常用的标准库和第三方库	232
12.1	turtle 库	232
12.1.1	turtle 库中的函数	232
12.1.2	绘图实例	235
12.2	random 库	236
12.3	time 库	237



12.4	PyInstaller 库	240
------	---------------------	-----

12.5	jieba 库	241
------	---------------	-----

12.5.1	jieba 库简介	241
--------	-----------------	-----

12.5.2	jieba 库应用实例	242
--------	-------------------	-----

12.6	wordcloud 库	243
------	-------------------	-----

12.6.1	wordcloud 库简介	243
--------	---------------------	-----

12.6.2	wordcloud 库应用实例	243
--------	-----------------------	-----

12.7	Matplotlib 库	244
------	--------------------	-----

12.7.1	Matplotlib 库简介	244
--------	----------------------	-----

12.7.2	使用 Matplotlib 库绘制图表	245
--------	---------------------------	-----

第 1 章

Python 入门概述

本章重点

- ◎ Python 语言的特点、应用及发展前景。
- ◎ 常用的 Python 语言集成开发环境。
- ◎ Python 软件的下载、安装及使用。
- ◎ Python 语言第三方库的安装和管理。

本章难点

- ◎ Python 软件及第三方库的安装。

Python 是一门面向对象的、解释型的编程语言,具有语法简洁、易读等特点,对于初学者来说,非常容易上手。同时,它也是一种功能强大且完善的通用型语言,已有 30 多年的发展历史,因此已经非常成熟和稳定。它的语法简洁而清晰,兼容性非常强,几乎可以在所有的操作系统中运行,能够支持绝大多数应用系统的构建。它拥有非常丰富和强大的第三方库,其应用范围从 Web 开发,到运维开发、搜索引擎、机器学习、游戏开发等。

在当前云计算、大数据、物联网、人工智能、区块链等新兴技术发展的新时代,Python 正扮演着越来越重要的角色。对于编程初学者而言,Python 是一个理想的选择。

1.1 Python 语言简介

1.1.1 什么是 Python 语言

Python 翻译成中文是巨蟒的意思,它是一种编程语言,于 1991 年正式发布。它的创造者吉多·范·罗苏姆(Guido van Rossum)是一名荷兰计算机程序员,选中 Python 作为编程语言的名字,是因为他喜爱的一个电视节目中有 Python 这个单词。Python 的定位是简单、明确、优雅,所以对初学者而言,Python 简单易懂、便于学习,而且将来深入学习后,还可以编写一些非常复杂的程序。

Python 语言是少有的一种可以称得上既简单又功能强大的编程语言。Python 编程的重点是解决问题,而不是语法与结构,非常适合完全没有编程经验的人学习。无论是想进入数据分析、人工智能、网站开发等专业领域,还是仅仅希望掌握一门编程语言,都可以从 Python 开始。

1.1.2 Python 语言的特点

Python 语言的特点如下。

1. 简单易读

Python 没有开始、结束、分号等标记,代码简洁,循环语句简化,程序结构清晰,易于阅读和理解,其保留字也很少,目前只有 35 个。

2. 跨平台可移植性强

Python 的源代码是开放的,基于这一特性,Python 可以被移植到其他平台。

Python 程序稍作修改甚至不修改便可在 Linux、Windows、FreeBSD、Solaris、OS/2、Amiga、AROS、AS/400、BeOS、OS/390、z/OS、Palm OS、QNX、Acorn RISC OS、VxWorks、Sharp Zaurus、Windows CE、Pocket PC 等平台上运行。

3. 动态性

Python 与 PHP、Ruby、ECMAScript(JavaScript)一样都属于动态语言。所谓动态语言就是在执行时能够改变其结构的语言,是高级程序设计语言的一个类别。Python 灵活性高,不需要声明变量,直接赋值即可创建新变量。例如,定义一个 Student(学生)类,初始属性有 name(姓名)和 age(年龄),代码如下。

```
class Student():
    def __init__(self, name = None, age = None):
        self.name = name
        self.age = age
s = student("Jack", "12")
```

学生属性除了 name(姓名)和 age(年龄)外,应该还有 sex(性别),如果是动态语言,直接给 s 对象的 sex 属性赋值“male”就可以得到想要的结果,如果是其他语言,这一步可能会报错。

```
s.sex = "male"
```

运行上面的代码,输出 s.sex 的值,结果为 male,这就是给实例动态绑定属性。

所以动态语言可以在运行时引进新的函数(function)、对象(object)甚至代码,还可以删除已有的函数等。

4. 健全的异常处理机制

Python 的异常处理机制能捕获程序异常情况,且堆栈跟踪对象能找到出错的地方和出错的原因,帮助人们调试程序。

5. 面向对象特性

Python 既支持面向过程的编程,又支持面向对象的编程。

Python 执行面向对象编程(object oriented programming, OOP)的方式既强大又简洁,它简化了面向对象的实现,消除了保护类型、抽象类、接口等面向对象元素,使面向对象的概念更容易理解。

6. 可扩展性

Python 由 C 语言开发,可以使用 C 或 C++ 完成新模块和类的添加,而 Python 程序可以完全调

用(call)它们,同时还可以嵌入 C 或 C++ 程序中。

Python 的扩展性非常好。例如,如果需要一段运行很快的代码,或想写一段不愿公开的代码,那么可以用 C 或 C++ 写这段代码,然后从 Python 程序中进行调用。

7. 丰富的库文件

Python 最大的优势之一是拥有丰富的库文件。

Python 有非常完善的基础代码库,覆盖了网络、数据库、文件、文本等大量内容,被形象地称为内置电池(batteries included)。Python 标准库很大,它能够帮助用户完成许多工作,如文件传输协议(file transfer protocol,FTP)、数据库、正则表达式、单元测试、线程、网页浏览器、公共网关接口(common gateway interface,CGI)、电子邮件、可扩展标记语言(extensible markup language,XML)、图形用户界面(graphical user interface,GUI)等。除了这些基础代码库之外,Python 还有大量高质量的第三方库,是其他人编写贡献出来的,用户可以通过 Python 包索引找到它们。

8. 易于维护

Python 成功的原因之一在于它的源代码是相当容易维护的。

1.1.3 Python 语言的应用

Python 应用范围很广,常用的应用场景如下。

1. 操作系统管理

Python 作为一种解释型的脚本语言,特别适合于编写操作系统管理脚本,使用 Python 编写的系统管理脚本在可读性、源代码重用度、扩展性等方面都优于普通的 shell 脚本。

2. 科学计算

Python 程序员可以使用 NumPy、SciPy、Matplotlib 等模块编写科学计算程序,用 sklearn 等模块完成数据挖掘及机器学习等编程。众多开源的科学计算软件包均提供了 Python 的调用接口,如著名的计算机视觉库 OpenCV、三维可视化库 VTK、医学图像处理库 ITK 等。

3. Web 应用

Python 经常被用于 Web 开发。例如,通过 mod_wsgi 模块可以运行使用 Python 编写的 Web 程序;基于 Python 的 Web 应用框架 Django 主要用来快速搭建高性能网站;许多大型的网站,如 YouTube、Dropbox、Douban 等都是用 Python 开发的。

4. 图形用户界面开发

Python 支持图形用户界面开发,使用 Tkinter、wxPython 或 PyQt 库可以开发跨平台的桌面软件。

5. 云计算

Python 是云计算最热门的语言之一,其典型应用为 OpenStack。

6. 金融

在金融领域,Python 不仅使用最多,而且其重要性逐年增加,如进行定量交易、金融分析等都可以通过 Python 来完成。

7. 其他

很多游戏使用 C++ 编写图形显示等高性能模块,而使用 Python 编写游戏的逻辑。

在国外,Python 有着广泛的应用范围,如谷歌的 Google.com、Google 爬虫、Google 广告和其他项

目正在广泛使用 Python;美国中央情报局(CIA)网站是用 Python 开发的;美国航空航天局(NASA)广泛使用 Python 进行数据分析和计算;世界上最大的视频网站 YouTube 是用 Python 开发的;美国最大的在线云存储网站 Dropbox 全部用 Python 实现;美国最大的照片共享社交网站 Instagram 每天有 3 000 多万张照片被共享,所有这些都是用 Python 开发的;世界上最流行的 Linux 发行版中的 Yum 包管理工具是用 Python 开发的;Douban 公司几乎所有的业务都是通过 Python 开发的。

在国内,Python 同样有着广泛的应用范围,如中国的 Q&A(问答)社区知乎就是用 Python 开发的,除此之外,还有搜狐、金山、腾讯、盛大、网易、百度、阿里、淘宝、土豆、新浪、果壳等公司正在使用 Python 来完成各种任务。

1.1.4 Python 语言的发展前景

虽然 Java 有众多使用者,但是 Python 的语法特点使程序设计更轻松,编写的代码比 Java 可读性更强。

2017 年,国务院印发《新一代人工智能发展规划》。规划中指出,支持开展形式多样的人工智能科普活动,在中小学设置人工智能相关课程。而 Python 是人工智能与大数据分析的基础工具,各地、市教育局可将 Python 列入小学、中学课程体系中。2017 年年底,山东省的小学信息技术六年级教材加入了 Python 内容,小学生开始接触 Python 语言。从 2018 年起,浙江省的信息技术教材不再使用 VB 语言,也改用 Python 语言。从 2018 年起,Python 被列入全国计算机等级考试范围。而 2022 年 PYPL 发布的编程语言指数榜单显示 Python 占据榜首,如图 1-1 所示。

Worldwide, Oct 2022 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	28.3 %	-1.8 %
2		Java	17.2 %	-0.9 %
3		JavaScript	9.69 %	+0.4 %
4		C#	7.2 %	-0.2 %
5		C/C++	6.45 %	-0.5 %
6		PHP	5.39 %	-0.9 %
7		R	4.03 %	+0.3 %
8	↑↑↑	TypeScript	2.71 %	+1.1 %
9	↓	Objective-C	2.16 %	+0.2 %
10	↑↑	Go	2.08 %	+0.5 %
11	↓	Swift	2.05 %	+0.4 %
12	↓↓↓	Kotlin	1.81 %	+0.1 %
13	↑↑↑↑	Rust	1.57 %	+0.8 %
14	↓	Matlab	1.52 %	+0.1 %
15		Ruby	1.13 %	+0.1 %
16	↓↓	VBA	0.96 %	-0.3 %
17	↑↑↑↑↑	Scala	0.78 %	+0.3 %
18	↑	Ada	0.76 %	+0.2 %
19	↓	Dart	0.75 %	+0.1 %
20	↓↓↓↓	Visual Basic	0.64 %	-0.2 %

图 1-1 2022 年 PYPL 发布的编程语言指数榜

1.1.5 Python 语言的版本

Python 已经历了 30 多个版本的发展。2008 年年底,Python 3.0 版本发布,目前已更新至 3.12 版本,中间还发布了 Python 2.5、Python 2.7 等版本,本书使用 Python 3.7 版本进行讲解。

这些版本之间存在着一定的联系,也有差异。例如,Python 2.x 和 Python 3.x 就有明显的差异。Python 3.x 由于不想带入过多的累赘,所以没有考虑向下兼容的问题,Python 3.0 以下版本的程序都无法在 Python 3.0 以上的版本中运行。为了解决这个问题,实现 Python 3.0 以下版本的程序向 Python 3.x 的迁移,中间又开发了几个过渡版本,其中,Python 2.7 是比较经典的版本。建议初学者直接安装 Python 3.0 以上版本。

1.1.6 Python 语言的集成开发环境

Python 是一种跨平台的脚本语言,在不同平台上提供了众多的集成开发环境(integrated development environment, IDE),可以提高用户的编程效率,常用的集成开发环境如下。

1. IDLE

IDLE 是 Python 内置的集成开发环境,在操作系统中安装好 Python 软件后就自动安装该编辑器。初学者一般建议使用 IDLE,该编辑器容易上手,且具备语法高亮功能,同时可以运行程序。

2. Spyder

Spyder 和其他 Python IDE 相比有一个很大的特点,可以用表格的形式查看数据,其针对数据科学做了一定的优化。使用 IDLE 交互式命令行确实很方便,但是当代码越来越多或越复杂时,就显得力不从心,这时选择一款适合的代码编辑器就显得很重要。Spyder 是一款不错的编辑器,在安装好 Anaconda 后就会自动被安装好,启动运行即可。

3. Jupyter Notebook

Jupyter Notebook(此前被称为 IPython Notebook)是一个交互式笔记本,支持运行 40 多种编程语言。它本质上是一个支持实时代码、数学方程、可视化和 Markdown 的 Web 应用程序。在安装好 Anaconda 后就会自动被安装好,启动运行即可。

4. PyCharm

PyCharm 是由 JetBrains 公司开发的商业 Python IDE,支持企业级的开发。PyCharm 有一整套可以帮助开发人员提高开发效率的工具,如调试、语法高亮、Project 管理、代码跳转、智能提示、自动完成、单元测试、版本控制等。

5. Eclipse + Pydev 插件

在通用集成开发环境 Eclipse 中安装 Pydev 插件,可以实现 Python 集成开发环境,方便调试程序。

6. PythonWin

PythonWin 是 Python 在 Windows 系统下的扩展包,使用 PythonWin 可以让 Python 使用 Windows 系统的特性。PythonWin 官网为 <https://sourceforge.net/projects/pywin32/>,下载对应的版本安装即可。

7. Vim

Vim(Vi iMproved)是由 Bram Moolenaar 编写的功能强大的文本编辑器。Vim 和其他开源软件

一样,具有良好的可移植性。Vim 提供了强大的程序代码编辑功能,如自动缩进、代码折叠、语法高亮等。Vim 需要单独下载安装,官网为 <http://www.vim.org/>,下载对应的版本安装即可。

8. Emacs

Emacs 被设计得“无所不能”,号称“世界上最强大的文本编辑器”,使用 Emacs 就像使用 Windows 的记事本一样,但 Emacs 比 Windows 记事本的功能强大得多。

可以从 <https://mirrors.ustc.edu.cn/gnu/emacs/windows/> 网站上下载编译好的 Emacs 安装程序,然后在 Windows 系统下进行安装。Emacs 安装程序是一个自解压的压缩文件,只需要选择解压目录进行解压即可。解压完成后,运行 Emacs 所在目录下 bin 目录的 runemacs.exe 文件,即可启动 Emacs。

9. Sublime Text

Sublime Text 也比较适合 Python 新手使用,Sublime Text 支持跨平台,而且提供丰富的插件和主题。Sublime Text 具备各种语法高亮和代码补全功能,视觉感受更舒适。

10. Visual Studio + Python Tools for Visual Studio

在 Visual Studio 基础上安装 Python Tools for Visual Studio,可以使用功能完善的 Visual Studio 开发 Python 程序。

1.2 Python 环境的准备

用户可以在多个平台上安装和开发 Python,如 Linux、UNIX、Windows、macOS 等。本节重点介绍在 Windows 平台上安装部署 Python。

在 Windows 平台上安装 Python 开发环境最受欢迎的方法有两种。其中,第一种是通过 Python 官网下载 Windows 对应系统版本的 Python 安装程序,第二种是通过 Anaconda 安装。



1.2.1 安装 Python

1. 下载安装程序

进入 Python 官网 <https://www.python.org/>,按以下步骤下载所需版本的 Python 安装程序。
步骤 01:在“Downloads”下拉列表中选择“Windows”选项,如图 1-2 所示。

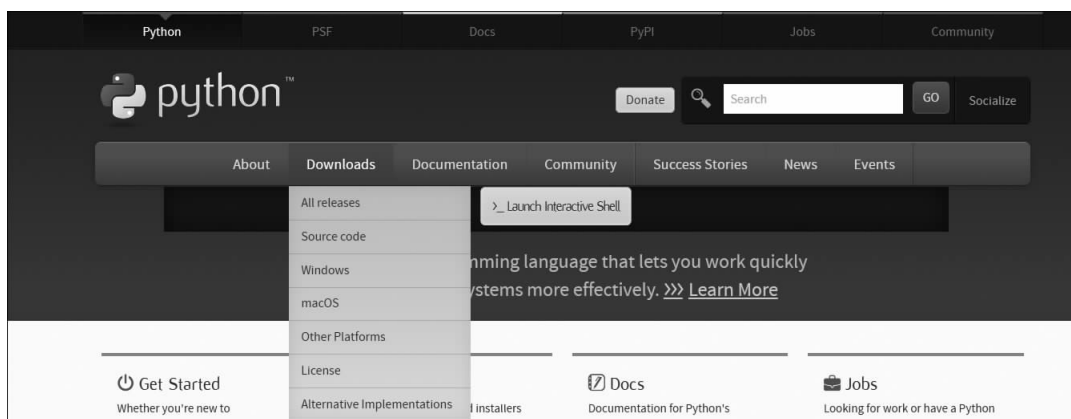


图 1-2 选择“Windows”选项

步骤 02: 打开 Windows 各版本下载页面, 如图 1-3 所示。

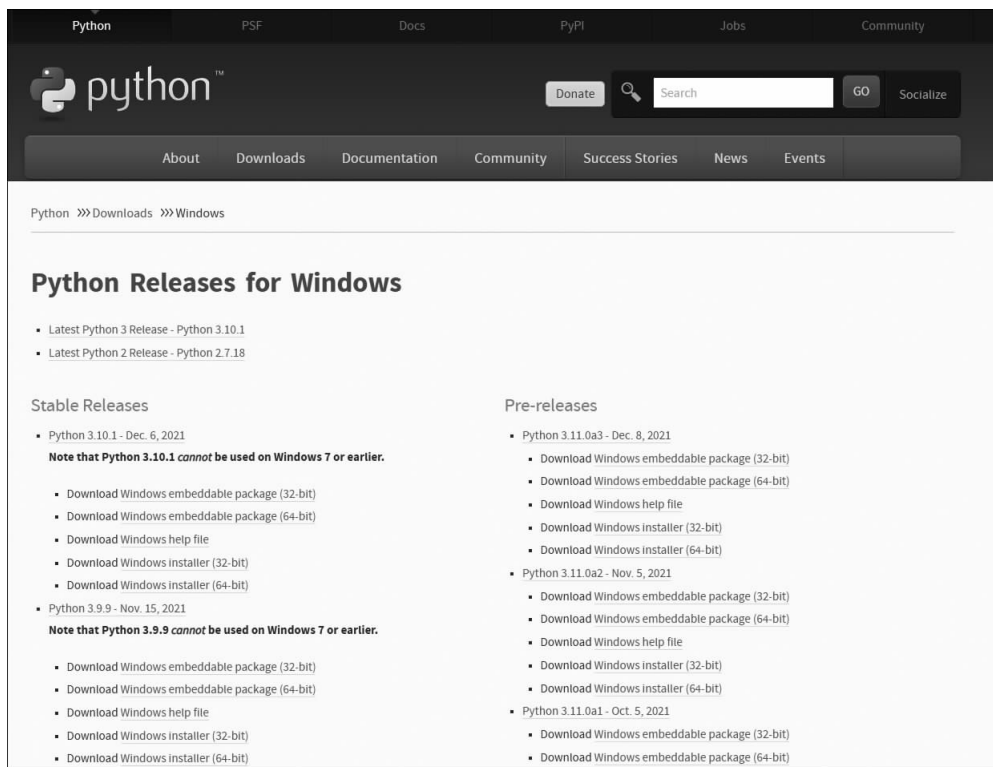


图 1-3 Python 的 Windows 各版本下载页面

从图 1-3 中可以看出, Python 的版本非常多。

步骤 03: 先确认自己的系统是 32 位还是 64 位, 再选择相应的 Python 版本下载, 在此选择 Python-3.7.8-amd64 版本。

2. 安装 Python 软件

下载好 Python 安装程序文件后, 按以下步骤安装即可。

步骤 01: 双击 Python 安装程序文件, 打开“Python 3.7.8(64-bit) Setup”窗口, 选择“Install Now”选项, 并选择“Add Python 3.7 to PATH”复选框, 如图 1-4 所示。



图 1-4 选择 Python 安装模式

步骤 02:系统开始自动安装,如图 1-5 所示。安装完成后单击“Close”按钮关闭窗口,如图 1-6 所示。

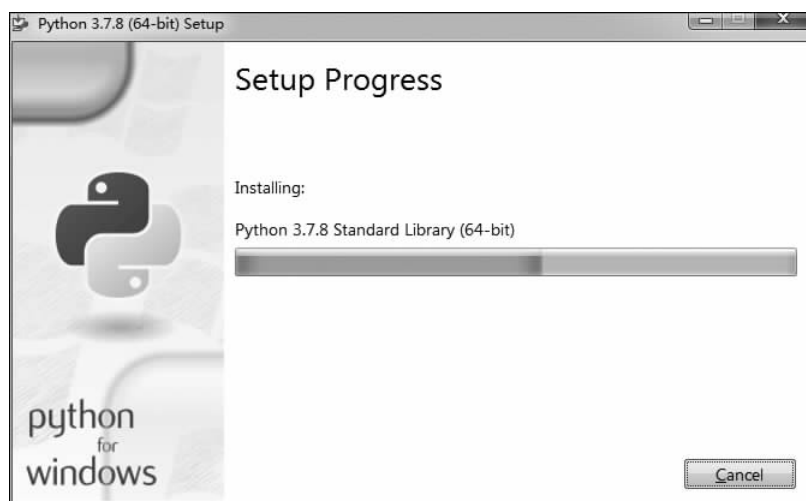


图 1-5 正在安装



图 1-6 安装成功

到此为止,Python 3.7 在 Windows 操作系统下的安装全部完成。

3. Python 菜单命令功能说明

执行“开始”→“所有程序”→“Python 3.7”命令,即可看到已经安装好的 Python 程序共包含 4 个菜单命令,如图 1-7 所示。

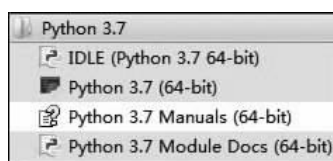


图 1-7 Python 3.7 菜单命令

各命令功能的含义如下。

- (1) IDLE(Python 3.7 64-bit): Python 集成开发环境。
- (2) Python 3.7(64-bit): Python 的命令提示符窗口。
- (3) Python 3.7 Manuals(64-bit): Python 的全部文档。
- (4) Python 3.7 Module Docs(64-bit): Python 系统中可用的所有已安装模块的文档。

1.2.2 使用交互式执行环境

使用交互式执行环境,即运行 Python 解释器。用户在 Windows 操作系统中可以使用命令提示符窗口运行 `python.exe`,也可以执行“开始”→“所有程序”→“Python 3.7”→“Python 3.7(64-bit)”命令,启动后的命令提示符窗口如图 1-8 所示。

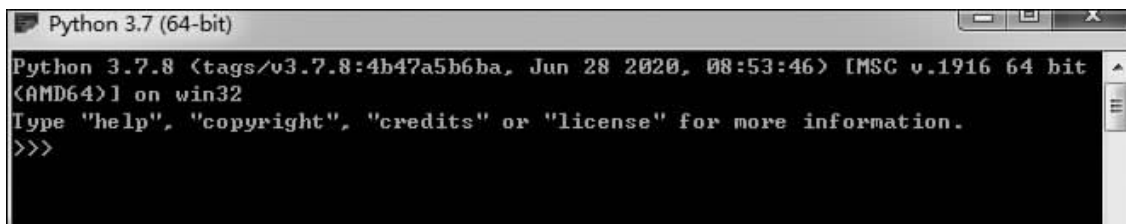


图 1-8 Python 命令提示符窗口

在命令提示符“>>>”后输入各种 Python 代码,按“Enter”键后就会立即看到执行结果。

例 1-1 使用 Python 解释器进行数学运算,运算的式子为 $12+36-78 \times 56$ 和 $(1+0.01) \times \times 365$ 。代码及执行结果如下。

```
>>> 12 + 36 - 78 * 56
- 4320

>>> (1 + 0.01) * * 365
37.78343433288716
```

命令行中两个式子的代码及执行结果如图 1-9 所示。

```
>>> 12+36-78*56
-4320

>>> (1+0.01)**365
37.78343433288716
```

图 1-9 在命令行中进行数学运算

1.2.3 使用 IDLE 编写代码

Python 安装成功后,会自带一个集成式开发环境 IDLE,它是一个 Python Shell,程序开发人员可以利用 Python Shell 与 Python 交互。

在 Windows 操作系统的开始菜单中找到“IDLE(Python 3.7 64-bit)”,单击进入 IDLE 主窗口,如图 1-10 所示,窗口左侧会显示命令提示符“>>>”,在该提示符后面输入 Python 代码,按“Enter”键后就会立即执行并返回执行结果。

如果要创建一个代码文件,可以在 IDLE 主窗口菜单栏中选择“File”→“New File”选项,就会弹出图 1-11 所示的文件窗口,可以在文件窗口中输入 Python 代码,然后选择保存并输入文件名。如果要运行该文件,在菜单栏中选择“Run”→“Run Module”选项(或按“F5”键)即可。