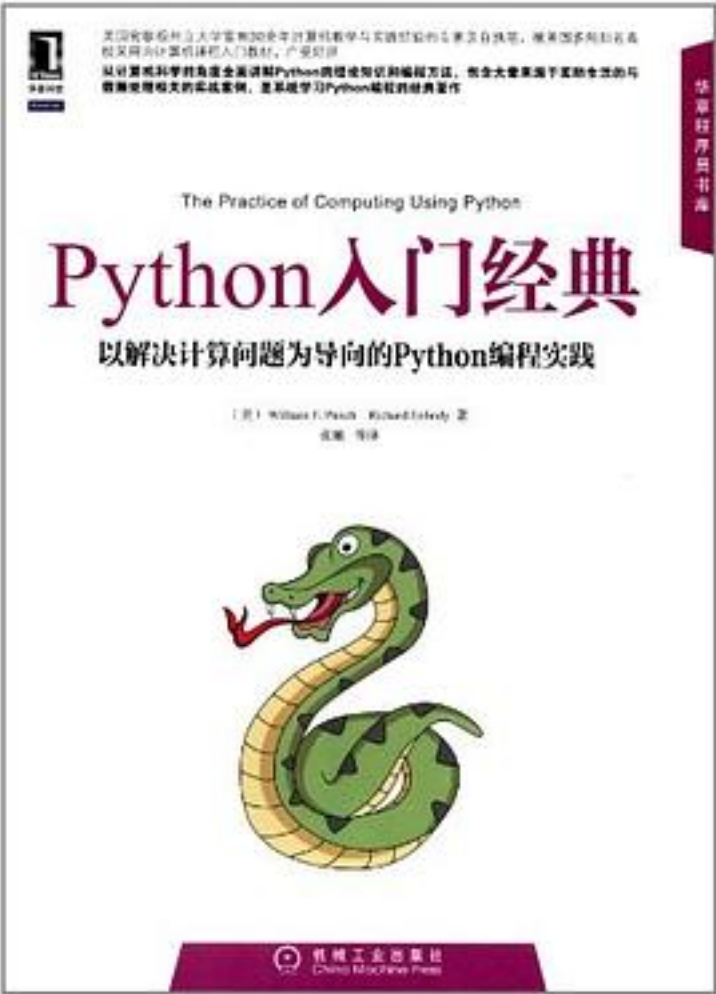


Python入门经典



[Python入门经典_下载链接1](#)

著者:(美)William F. Punch/Richard Enbody

出版者:机械工业出版社

出版时间:2012-8-1

装帧:平装

isbn:9787111394136

《Python入门经典:以解决计算问题为导向的Python编程实践》是一本系统而科学的Python入门教程，美国密歇根州立大学等多所美国知名高校采用其作为编程语言的入门教

材，被奉为经典。它不仅从计算机教学和计算机科学的角度讲解了初学者如何才能更有效地去学习Python，而且特别强调用Python解决生活中的实际问题，精心组织了大量来源于生活中不同领域的与数值计算和数据处理相关的案例。通过《Python入门经典:以解决计算问题为导向的Python编程实践》，读者不仅能系统掌握Python编程相关的知识，而且还能掌握利用Python处理各种与数据相关的问题。《Python入门经典:以解决计算问题为导向的Python编程实践》分为五部分，第一部分介绍计算机设备的一般概念和一些计算机术语；第二部分开始介绍编程的基本知识，包括入门知识和控制语句，为后续内容做铺垫；第三部分讲述数据结构和函数，包括字符串、列表和元组、字典和集合、文件、函数、算法和程序等进阶内容，有助于读者提升编程技能；第四部分重点介绍如何使用类定义数据结构和算法、开发程序等，培养读者运用Python语言来实现基本的计算思想和策略的能力；第五部分主要介绍异常、测试和递归，旨在使读者成为更好的程序员。《Python入门经典:以解决计算问题为导向的Python编程实践》深入浅出，每一章节均给出了大量的实例、示范代码和自测练习，便于读者理解和掌握相关知识。读者通过学习《Python入门经典:以解决计算问题为导向的Python编程实践》，不仅能掌握Python语言的基本知识，还能学习如何在实践中运用该语言解决问题。

作者介绍:

目录: 译者序

前言

第一部分 关于计算机的思考

第0章 计算机科学研究

0.1为什么要研究计算机科学

0.1.1计算机科学的重要性

0.1.2计算机“科学”

0.1.3通过编程学习计算机科学

0.2编程的困难和使命

0.2.1困难1：同时做两件事

0.2.2困难2：什么是好程序

0.2.3程序的使命

0.3选择一种计算机语言

0.3.1各种计算机语言

0.3.2为什么选Python

0.3.3Python是最好的程序语言吗

0.4什么是计算

0.5什么是计算机

0.5.1自然界中的计算

0.5.2人类制造的计算机

0.6现代电子计算机

0.6.1就是开关

0.6.2晶体管

0.7从更高层面来了解现代计算机

0.8数据表示

0.8.1二进制数据

0.8.2使用二进制

0.8.3局限性

0.8.4字符表示

0.8.5其他数据表示

0.8.6数字代表什么

0.8.7数据量

0.8.8数据量有多大

0.9后续章节概述

0.10总结

第二部分 开始编程

第1章 入门

1.1练习，练习，再练习

1.2快速入门——计算圆周长的程序

1.3交互式会话

1.4程序组成部分

1.4.1模块

1.4.2表达式和语句

1.4.3空白

1.4.4注释

1.4.5Python 的特殊元素：标记

1.4.6对象命名

1.5变量

1.6对象和类型

1.6.1数字

1.6.2其他内置类型

1.6.3对象类型：非变量类型

1.6.4创建新值

1.7运算符

1.7.1整数运算符

1.7.2浮点运算符

1.7.3混合运算符

1.7.4运算符顺序和圆括号

1.7.5增强的赋值运算符：快捷方式

1.8第一个模块：math模块

1.9开发算法

1.10总结

1.11视觉场景：海龟绘图

习题

编程项目

第2章 控制语句

2.1选择语句：if

2.1.1利用布尔值做决定

2.1.2if 语句

2.1.3示例：在篮球运动中，领先多少分才安全

2.1.4循环

2.1.5例子：寻找完全数

2.1.6例子：对数字分类

2.2深入控制语句

2.2.1真与假：布尔值

2.2.2布尔变量

2.2.3关系运算符

2.2.4布尔运算符

2.2.5优先级

2.2.6布尔运算符示例

2.2.7另一种赋值方式

2.2.8用于判定的选择语句

2.2.9Python判定语句进阶

2.2.10循环：while语句

2.2.11信号量循环

2.2.12循环总结

2.2.13for语句进阶

2.2.14嵌套

- 2.2.15冰雹序列示例
- 2.3视觉场景：用pylab对数据绘图
 - 2.3.1使用列表和第一次绘制
 - 2.3.2更有趣的绘图：正弦波
- 2.4计算机科学观点：最小的通用计算
- 2.5总结

习题

编程项目

第3章 算法和程序开发

- 3.1什么是算法
- 3.2算法特征
 - 3.2.1算法和程序
 - 3.2.2细化
 - 3.2.3有效性
 - 3.2.4指定行为
 - 3.2.5通用算法
 - 3.2.6真的可以实现一切吗
- 3.3程序是什么
 - 3.3.1可读性
 - 3.3.2鲁棒性
 - 3.3.3正确性
- 3.4程序设计策略
 - 3.4.1参与并提交
 - 3.4.2了解，然后想象
 - 3.4.3编程之前先思考
 - 3.4.4实验
 - 3.4.5简化
 - 3.4.6停下来思考
 - 3.4.7放松：让自己休息一下
- 3.5简单示例
 - 3.5.1搭建框架
 - 3.5.2输出
 - 3.5.3输入
 - 3.5.4计算
- 3.6总结

习题

第三部分 组织：数据结构和函数

第4章 字符串

- 4.1字符串类型
 - 4.1.1三重引号字符串
 - 4.1.2非显示字符
 - 4.1.3字符串表示形式
 - 4.1.4字符序列
 - 4.1.5索引和分片
- 4.2字符串操作
 - 4.2.1连接 (+) 和重复 (*)
 - 4.2.2 “+” 什么时候表示加法运算，什么时候表示连接运算
 - 4.2.3比较运算符
 - 4.2.4in运算符
 - 4.2.5字符串集合是不可变的
- 4.3函数和方法预览
 - 4.3.1第一步：什么是函数
 - 4.3.2选择方法的名字和参数
 - 4.3.3字符串方法

- 4.3.4字符串函数
- 4.4字符串的格式化输出
 - 4.4.1描述符码
 - 4.4.2宽度描述符
 - 4.4.3浮点数精度描述符
- 4.5字符串与控制语句
- 4.6处理字符串
 - 4.6.1示例：记录人名
 - 4.6.2回文
- 4.7示例：计算扑克牌
- 4.8总结
- 习题
- 编程项目
- 第5章 函数快速入门
 - 5.1函数是什么
 - 5.2Python 函数
 - 5.3函数控制语句
 - 5.3.1函数控制语句详解
 - 5.3.2另一个函数示例
 - 5.3.3函数示例：猜词
 - 5.3.4函数调用函数
 - 5.3.5什么时候使用函数
 - 5.3.6如果没有return语句会如何
 - 5.3.7如果有多条return语句会如何
 - 5.4视觉场景：用海龟绘图法绘制美国国旗
 - 5.5总结
 - 习题
 - 编程项目
- 第6章 列表和元组
 - 6.1什么是列表
 - 6.2操作列表
 - 6.2.1索引和分片
 - 6.2.2运算符
 - 6.2.3函数
 - 6.2.4列表循环
 - 6.3列表新内容
 - 6.3.1列表可变性
 - 6.3.2列表方法
 - 6.4range、split及其他函数和方法
 - 6.4.1range、split和多重赋值
 - 6.4.2使用join在列表和字符串之间转换
 - 6.4.3sorted 函数
 - 6.5示例
 - 6.5.1字谜
 - 6.5.2示例：文件分析
 - 6.6可变对象及其引用
 - 6.6.1深拷贝与浅拷贝
 - 6.6.2可变与不可变
 - 6.7元组
 - 6.7.1从列表到元组
 - 6.7.2为什么需要元组
 - 6.8列表：数据结构
 - 6.8.1数据结构示例
 - 6.8.2数据结构的另一个示例

6.9算法示例：美国环境保护署通车里程数据

6.10列表解析

6.11视觉场景：更多绘制任务

6.11.1numpy阵列

6.11.2绘制三角函数

6.12总结

习题

编程项目

第7章 深入了解函数

7.1函数调用函数

7.2作用域

7.2.1实参、形参和命名空间

7.2.2传递可变对象

7.2.3返回复杂对象

7.2.4重构evens

7.3默认值以及形参为关键字

7.3.1示例：默认值和参数关键字

7.3.2默认值问题

7.4函数和对象

7.5示例：确定最终成绩

7.5.1数据

7.5.2设计

7.5.3函数：weightedGrade

7.5.4函数：grade

7.5.5函数：main

7.5.6使用示例

7.6 “传值”或者“传引用”

7.7总结

习题

编程项目

第8章 字典和集合

8.1字典

8.1.1字典示例

8.1.2Python 字典

8.1.3字典索引和赋值

8.1.4运算符

8.2单词计数示例

8.2.1统计字符串中的单词数

8.2.2《葛底斯堡演说》中的单词出现频率

8.2.3输出和注释

8.3示例：周期表

8.3.1使用CSV文件

8.3.2算法概述

8.3.3实现分治的函数

8.4集合

8.4.1集合的历史

8.4.2集合的组成

8.4.3Python 集合

8.4.4Python集合的方法、运算符和函数

8.4.5集合方法

8.5集合应用

8.5.1不同文件中单词之间的关系

8.5.2输出和注释

8.6作用域：完整的故事

- 8.6.1命名空间和作用域
- 8.6.2作用域搜寻规则
- 8.6.3局部命名空间
- 8.6.4全局命名空间
- 8.6.5内置模块
- 8.6.6封闭式变量
- 8.7Python指针：使用zip创建字典
- 8.8视觉场景：词频条形图
 - 8.8.1正确获取数据
 - 8.8.2标签和xticks命令
 - 8.8.3绘图
- 8.9总结
- 习题
- 编程项目
- 第9章 文件
 - 9.1什么是文件
 - 9.2存取文件：读取文本文件
 - 9.2.1其他文件存取方法
 - 9.2.2数据流
 - 9.3存取文件：写文本文件
 - 9.4在程序中存取文本文件
 - 9.5创建文件和重写文件
 - 9.5.1通用新行格式
 - 9.5.2文件内移动
 - 9.6关闭文件
 - 9.7CSV文件
 - 9.7.1CSV模块
 - 9.7.2CSV Reader
 - 9.7.3CSV Writer
 - 9.7.4示例：更新某些成绩
 - 9.8示例：反复提示，要求输入正确的文件名
 - 9.9模块：os
 - 9.9.1目录/文件夹的结构
 - 9.9.2os模块函数
 - 9.9.3os模块示例
 - 9.10总结
- 习题
- 编程项目
- 第10章 程序开发进阶
 - 10.1简介
 - 10.2分治
 - 10.3乳腺癌分类
 - 10.3.1问题
 - 10.3.2方法：分类
 - 10.3.3训练和测试分类器
 - 10.3.4构造分类器
 - 10.4设计分类器算法
 - 10.4.1先分解，再合并
 - 10.4.2数据结构
 - 10.4.3文件格式
 - 10.4.4makeTrainingSet函数
 - 10.4.5makeTestSet函数
 - 10.4.6trainClassifier函数
 - 10.4.7第2轮修改后的trainClassifier

10.4.8用新数据测试分类器

10.4.9reportResults函数

10.5在完整数据上运行分类器

10.6其他有趣的问题

10.6.1标签云

10.6.2标准普尔500预测

10.6.3用国旗预测宗教

10.7总结

习题

编程项目

第四部分 类：自定义数据结构和算法

第11章 类

11.1面向对象编程

11.1.1Python是面向对象的

11.1.2OOP特性

11.2使用OOP

11.3使用类和实例

11.3.1内置类和实例

11.3.2第一个类

11.3.3修改属性

11.3.4实例和类之间的特殊关系：instance of

11.4对象方法

11.4.1使用对象方法

11.4.2编写方法

11.4.3特殊参数self

11.4.4方法是类实例的接口

11.5融入Python类模型

11.5.1程序员定义的类

11.5.2Student类

11.5.3Python标准方法

11.5.4三种角色：类设计者、程序员和用户

11.6示例：Point类

11.6.1构造函数

11.6.2距离

11.6.3两点求和

11.6.4改进Point类

11.7Python和OOP

11.7.1封装性

11.7.2继承

11.7.3多态性

11.8Python和其他OOP语言

11.8.1公有与私有

11.8.2使用双下划线表示私有

11.8.3Python的宗旨

11.8.4修改实例

11.9总结

习题

编程项目

第12章 类进阶

12.1更多类属性

12.2Python实现机制

12.2.1类、类型与自检

12.2.2运算符重载

12.3自定义运算符重载

- 12.4创建有理数类
 - 12.4.1生成类
 - 12.4.2分数加法回顾
 - 12.4.3分数加法
 - 12.4.4相等和分数化简
 - 12.4.5应用分治
- 12.5错误消息
 - 12.5.1自检
 - 12.5.2修复int+Rational错误
- 12.6继承
 - 12.6.1“寻找属性”游戏
 - 12.6.2使用继承
 - 12.6.3实例：物理学标准模型
- 12.7总结
- 习题
- 第13章 使用类开发程序
 - 13.1捕食问题
 - 13.1.1规则
 - 13.1.2面向对象的模拟
 - 13.2类
 - 13.2.1Island类
 - 13.2.2捕食者和猎物、动物种类
 - 13.2.3捕食者类和猎物类
 - 13.2.4对象图
 - 13.2.5填充Island
 - 13.3添加行为
 - 13.3.1细化：添加移动
 - 13.3.2时间循环仿真
 - 13.4逐步求精
 - 13.4.1改进的时间循环
 - 13.4.2繁殖
 - 13.4.3进食
 - 13.4.4时钟节拍
 - 13.5细化问题
 - 13.5.1移动多少次
 - 13.5.2动物数量的图形化
 - 13.6总结
 - 习题
- 第五部分 成为更好的程序员
- 第14章 异常和异常处理
 - 14.1简介
 - 14.2基本的异常处理
 - 14.3有关异常的哲学
 - 14.4异常：else和finally
 - 14.5异常的用法
 - 14.5.1检查输入
 - 14.5.2检查文件打开
 - 14.6深入异常
 - 14.6.1raise
 - 14.6.2自定义异常
 - 14.7示例：密码管理
 - 14.8总结
 - 习题
- 第15章 测试

- 15.1为什么要进行测试
 - 15.1.1错误类型
 - 15.1.2“bug”和调试
- 15.2测试类型
 - 15.2.1测试很难
 - 15.2.2测试的重要性
- 15.3示例
 - 15.3.1NBA效率
 - 15.3.2基本算法
- 15.4混合测试
 - 15.4.1捕捉用户错误
 - 15.4.2捕获开发者犯的错误
- 15.5自动测试
 - 15.5.1doctest
 - 15.5.2其他类型的测试
- 15.6总结
- 习题
- 第16章 递归：另一种控制机制
 - 16.1什么是递归
 - 16.2数学和兔子
 - 16.3自定义递归：反转字符串
 - 16.4递归如何实现
 - 16.4.1栈的数据结构
 - 16.4.2栈和函数调用
 - 16.5用递归表示图形
 - 16.5.1递归树
 - 16.5.2Sierpinski三角形
 - 16.6从递归到非递归
 - 16.7总结
 - 16.8习题
- 附录
 - 附录A 开始使用Python
 - 附录B 用海龟绘图法进行简单绘图
 - 附录C 绘图和数值工具：快速浏览
 - 附录D Python 3.0
 - 附录E ASCII码表
 - 附录F 优先级
 - • • • • ([收起](#))

[Python入门经典_下载链接1](#)

标签

Python

编程

计算机

入门

python

Programming

脚本语言

练习

评论

表达式和赋值语句 空格

缩进和算术表达式以及运算符；控制结构选择和循环；所有的程序就是如此构造，其他的都是为了使程序易于阅读和书写；程序是算法的实现，已知算法和程序就能够解决复杂的问题；

不适合0基础的人看，有其他语言基础的人可以扫读。不建议买，建议图书馆借。新手入门建议看老齐py教程。全书废话过多，只建议扫读，当做字典参考。

入门。看了部分章节内容和习题，感觉还行。作者行文很流畅，也很轻松。这是最好的入门书。可惜下载的PDF内容不完整。还有机器学习, 看病. 训练集, 测试集.

用来回忆py的...

非常适合入门。我只是粗略过了一遍，没能跟着做题，不过讲解内容足以入门，跟着做题效果应该更好。

1、太啰嗦了，再也不看这种四五百页没什么干货的大部头了。
2、翻译很不好，爱迪生的名言竟然被翻译成“现在你又知道了这种方法行不通”，大哥，翻译成“现在你又知道了一条行不通的方法”是不是更好？两种翻译的意思完全不同。。。functional programming竟然被翻译成“功能性编程”，译者自己明显编程功底很弱
3、一堆勘误

翻译的不好，没有提到多线程。

看了一遍，还不错，一些例子很实用。

这本书的Python介绍写的非常实用和经典，有种上课的感觉。非常适合学习。很多地方侧重了许多科学计算中需要用到的东西，比如输入输出格式，字符串的修改。

以python2.7讲述python编程。以一个熟悉C/C++等语言的人来看的话，比较适合python语言的入门。

很好的入门书籍

非程序员入门级读物，仅仅是入门·····

都是基本的知识，看来应该算是入门不错的书籍

作为非程序员的入门书还可以

不错的入门书

基本看完了。名副其实，的确是python入门经典。都是每天早上看一章，没上机调试过代码。现在看《机器学习实战》，正好练下代码。

完全没学过编程语言的话，不妨试试。我只是很快翻了一下而已，里面的习题都没做，因为是图书馆借的缘故。不过书里面的实际应用倒是挺有意思的，一般都是数据处理之类的。

读者定位应该是初学者。校验比较差，出现了很多错误的地方。

这是我读过的最好的编程语言入门书

不仅仅是python入门，就是一门编程语言的入门

[Python入门经典 下载链接1](#)

书评

首先定个调，这是我读过的最好的编程语言入门书！！
超级适合没什么编程经验和编程实践的同学，不管你是不是有接触过其他语言。
----- 我先介绍下自己的情况，可能方便一些同学的比较：
我虽然断断续续有接触过C、C++、C#、Java，但编程实践和编程经验...

这本书的面向人群基本是没有任何编程基础的人，（这也很符合这本书作为教材的定位），书中不仅讲了python的语法，用法，而且很注重coding style/methodology这类知识的补充。
另外，这本书在章节编排上很用心：如果一块内容包含多个层次不一的知识点，则会拆分为两个章节，包...

这本书涵盖了python的基本部分如控制语句、数据结构、函数以及类。里面的实例对非计算机专业的人很有启发，比如对巴比伦算法的python语言实现，分析《葛底斯堡的演说》中每个单词出现的评论，利用类定义物理学标准模型等。
和其他的程序语言书籍不同，这本书没有过多的谈论Pyth...

开篇一直到结束都很基础，但有些点描述的比较清楚，收获较大的就是在命名空间与作用域的阐述上面...python变量与C的确有非常大差别，他的名称关联到对象更有一种c指针的感觉.没看这本书之前的确觉得python这块充满神秘感，但现在已经觉得不过如此...还有比较赞的是，这本书的...

1、太啰嗦了，再也不看这种四五百页没什么干货的大部头了。
2、翻译很不好，爱迪生的名言竟然被翻译成“现在你又知道了这种方法行不通”，大哥，翻译成“现在你又知道了一条行不通的方法”是不是更好？两种翻译的意思完全不同。。。functional programming竟然被翻译成“功能...

不知道别人的评论是怎么得出的，反正中文版的看着实在难受，难道他们看得都是原版的？编程在天朝也发展这么多年了，每种语言都有丰富的库文件，译者却非要用‘模块’，难道翻译者就没接触过编程？类似多多，实在看不下去了，明天扔回图书馆

[Python入门经典_下载链接1](#)