信息工程系

教

案

课程名称： 专业技能实训 （Python 编程部分）

教　　师： 黄梅佳

总 学 时 ： 36（总72学时）

理论学时：

实训学时： 36

上课班级： 移动应用开发 241

授课学期： 2025-2026 第二学期

课题名称	第一章 Python 入门	计划学时	4 学时
教学内容	课程介绍 Python 的前世今生及应用领域 搭建 Python 开发环境 利用 Python 编辑器编写简单 Python 程序		
教学目标及基本要求	1. 了解程序设计语言的分类 2. 了解 Python 的发展历程和特点 3. 掌握 print（）函数的使用 4. 了解关键字的概念 5. 熟练搭建 Python 开发环境 6. 熟悉利用 Python 编辑器编写简单 Python 程序		
教学重点	1. 搭建 Python 开发环境 2. print（）函数		
教学难点	●		
教学方式	教学做一体化		
教学过程	**第一课时** **（课程介绍）** **一、创设情境，导入课程** 小明的学校这个学期来了 8 位留学生。学校为留学生们单独开设汉语言课，强化汉语学习，以解决专业学习中的交流障碍。小明偶尔也会和他们用英语进行简单的交流。当我们想要主动与使用不同语言的人沟通的时候，需要使用双方都能听懂的语言。 假设，现在你想让计算机帮你做一些事情，那你该如何与它进行沟通？ 答案很简单，你需要用计算机能“听懂”的语言才能与它沟通。 问题来了，计算机能听懂什么语言？计算机能听懂的语言其实有许多，例如 C、C++、Java、Python 等。这些被我们称为程序设计语言。 课程思政：有效沟通的重要性。在介绍 Python 开源特点时，引导学		

生尊重开源协议，鼓励参与开源项目，培养诚信和奉献精神。在解决安装问题环节，鼓励学生自主探索、克服困难，培养坚韧不拔的意志品质。在实训操作中，要求学生规范编写代码，输出积极向上的内容，培养责任感与良好价值观。

介绍程序设计语言的发展。

二、介绍 Python 程序设计语言的发展及应用领域

由“人生苦短，我用 Python”，提出 Python 是目前最受欢迎的程序设计语言。Python 是由荷兰人吉多·范罗苏姆（Guido van Rossum）在 1989 年设计的一种计算机程序设计语言。它是一种动态的、面向对象的语言。

（1）Python 的前世今生

介绍 Python 的起源、龟叔 Python 之父的发明。介绍有趣的名字起源和图标的由来。增加课程趣味性。

课程思政：开放共享精神、开源奉献精神。

（2）Python 的应用领域

经过多年的发展，Python 已经成为最受欢迎的程序设计语言。由于 Python 具有简洁性、易读性和可扩展性，国内外用 Python 进行科学计算的研究机构日益增多。除此之外，Python 还能完成许多领域的工作：

- Web 开发；
- 大数据应用；
- 人工智能应用；
- 桌面界面开发；
- 软件开发；
- 后端开发；
- 网络爬虫应用。

三、介绍本门课程学习内容、学习方法、考核方式

（1）学习内容（学什么）

Python基础与办公自动化应用		
第一阶段：Python基础篇	入门	环境安装、第一行代码
	基础	运算符、表达式、循环与分支、数据类型、函数与模块
第二阶段：办公与自动化应用	数据提取	正则表达式、爬虫
	办公自动化	Python批量处理Excel
	文档的批量处理	Python批量处理Word、PDF
	图像批量处理	Python批量处理图像
第三阶段：拓展知识	大数据入门	数据统计分析、数据可视化
	人工智能入门	电影类型识别，机器学习

（2）学习目标（学了有什么用）

- 掌握 Python 语言的基本概念和基本语法
- 能够编写 Python 语言实现办公自动化领域中的一些具体应用
- 能够初步掌握 Python 语言在大数据和人工智能领域的简单应用

	让学生了解非计算专业学 Python 的用途。 （3）学习方法（怎么学） ● 认真听课、动手实践。 ● 登录人民邮电出版社教育社区（www.ryjiaoyu.com）免费下载配套学习视频、源代码等资源。 ● 通过配套的拓展学习平台——编程胶囊实现在线编程，扫描二维码即可实现相关操作。 让学生了解零基础学 Python 的途径。 （4）考核方式（怎么考） 在线测试，客观题（选择、判断、填空）+主观题（编程） **四、归纳总结** Python 应用广泛，掌握 Python 编程可解决学习、工作、生活中的一些实际问题。 **第二课时** （搭建 Python 开发环境，在 IDLE 中打开 Python，编写你的第一行代码） ● 一、下载并安装 Python （1）教师边讲边演示下载 Python 的安装包。 教师指导学生在 Python 官网上查找并选择适合的版本，介绍下载的方法及保存的位置。 课程思政：版权意识、安全意识。 （2）教师边讲边演示 Windows 中安装 Python 的步骤。注意安装路径与安装程序保存位置不要混在一起。 ● 知识点：什么是环境变量？ 环境变量（enviroment variables）是操作系统中用来指定操作系统运行环境的一些参数。在向 Windows 和 DOS 操作系统中搭建开发环境时常常需要配置环境变量 Path，以便系统在运行一个程序时可以获取到程序所在的完整路径。 课程思政：学习计算机专业要加强英语学习 （3）教师演示如何在命令提示符窗口验证 Python 是否安装成功。 帮助学生查找验证不成功的原因，教育学生学会分析问题、解决问题。 ● 二、启动 Python 的两种方式 （1）命令提示符窗口启动 Python 知识点：什么是命令提示符窗口 （2）在 IDLE 中启动 Python

	知识点：什么是 IDLE ● 三、编写你的第一行代码 教师演示计算机上显示“Hello World！” ● 知识点：关键字（Keyword） ● 教师分析如何通过报错信息找出问题所在 ● 课程思政：代码规范性、工作细致 ● 第三课时 ● （了解 Python 的两个特点，编写真正意义上的 Python 程序） ● 一、介绍 Python 的两个特点 ● Python 的优点有简洁、语法优美、简单易学、开源、可移植、扩展性良好、类库丰富、通用灵活、模式多样、良好的中文支持；Python 的缺点有执行效率不够高、Python 3.x 和 Python 2.x 不兼容。 （1）“计算” 教师边讲边演示简单算术运算代码编写 知识点：运算符 （2）“重复做某事” 教师边讲边演示重复输出的代码 知识点：Python 语言的简洁 ● 二、编写真正意义上的 Python 程序 （1）创建 Python 代码文件 教师边讲边演示操作步骤 （2）运行 Python 程序 ● 教师边讲边演示，先保存，后运行 ● 知识点：文件扩展名的概念 ● 三、布置课中实践操作，检查并评测 ● 教师布置随堂实践操作，重复输出 10 遍“我为人人，人人为我” ● 教师对学生完成情况进行分析，表扬优秀，分析未完成学生或出错学生存在的问题，制订帮扶机制。 ● 四、归纳总结，布置随堂练习 （1）回顾上课前的学习目标，对本节课知识点进行总结。 教师带领学生总结本节课需要了解或掌握的知识点，包括发展历程、特点。搭建 Python 开发环境的 3 个步骤。 （2）布置随堂练习，检查学生掌握情况。 结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。

	第四课时 **（实训操作）** 实训操作主要针对项目中需要重点掌握的知识点，以及在程序中容易出错的内容进行练习，通过实训操作可以考察学生知识点和技能点的掌握情况，通过反复训练以达到项目目标。 **上机一：** **形式：单独完成** **题目：（使用 print()输出表情符号）** 在计算机上安装 Python，具体要求如下： （1） 从 Python 官网下载相应的安装包； （2） 安装 Python 解释器，并将安装路径添加环境变量中； （3） 打开命令行工具，输出表情符号，退出 Python 环境。 **上机二：** **形式：单独完成** **题目：（使用 print( )输出个人信息）** （1） 创建代码文件； （2） 编写个人信息输出程序； （3） 运行代码程序，得出正确结果，退出 Python 环境。
作业	练习题/操作题 ● 安装 Python 并编写第一个程序： 安装 Python，并在 IDLE 或 PyCharm 中编写并运行一个程序，输出“Hello, World!”。尝试使用 print()函数输出你的名字和学号。 ● 变量和数据类型练习： 定义两个变量 a 和 b，分别赋值为 10 和 20，然后交换它们的值并打印结果。 定义一个字符串变量 name，赋值为你的名字，并使用 type()函数检查其数据类型。
教学后记	本次教学中，多数学生对 Python 表现出浓厚兴趣，在安装 Python 和编写简单代码方面积极性较高。但部分学生在理解环境变量概念和处理报错信息时存在困难。今后教学应加强实践指导，针对难点增加案例讲解，帮助学生更好地掌握知识。

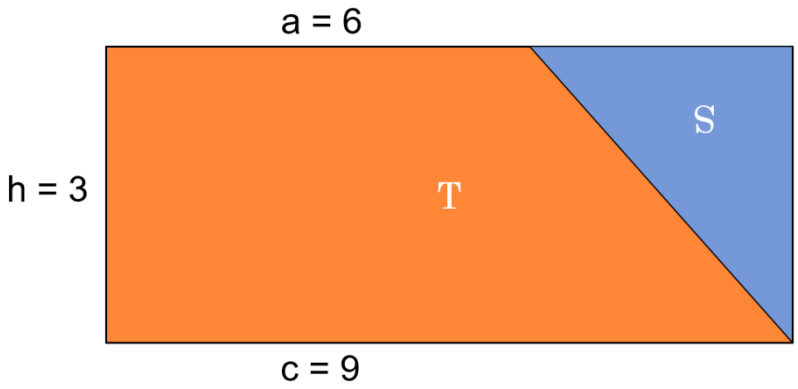
<table>
<tr><td>课题名称</td><td>项目二：解决简单的数学问题
Python 运算符与表达式</td><td>计划学时</td><td>4 学时</td></tr>
<tr><td>教学内容</td><td colspan="3">课程介绍
利用 Python 计算梯形面积
利用 Python 计算三角形面积
编写人民币与越南盾的兑换程序</td></tr>
<tr><td>教学目标及基本要求</td><td colspan="3">1. 掌握 Python 的基础语法（变量、注释、输入与输出）
2. 了解 Python 的运行时异常和错误
3. 掌握在 Python 中如何处理程序异常
4. 熟悉 Python 中的运算符和表达式（基本运算符、操作符、运算顺序、整除与余数）</td></tr>
<tr><td>教学重点</td><td colspan="3">1. Python 的基础语法
2. 在 Python 中如何使用数学运算符和表达式</td></tr>
<tr><td>教学难点</td><td colspan="3"></td></tr>
<tr><td>教学方式</td><td colspan="3">教学做一体化</td></tr>
<tr><td>教学过程</td><td colspan="3">第一课时
（课程介绍）
一、项目场景，导入课程
生活中，数学无处不在，特别是在编程中，经常会使用数学。在很多游戏和应用程序中我们经常会见到数据统计、排行榜等计算，可以说绝大多数的程序都会使用到数学。
在本项目中我们将通过日常生活中的一些案例来学习 Python 中数学计算的基础知识。
二、介绍 Python 与数学相结合解决简单的数学问题
在实践的过程中学习编程，通过编程与数学相结合的方式学习和练习编程技能，可以帮助我们锻炼编程思维，让我们在解决数学问题的过程中掌握编程技能。
（1）介绍项目目标-使用 Python 计算梯形面积</td></tr>
</table>

使用基础的数学知识——加减乘除，来解决三个数学问题。

（1） 求梯形“T”的面积；

（2） 使用公式计算等腰三角形 “S”的面积；

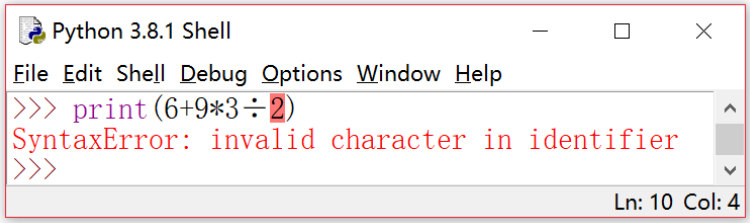
（3） 利用长方形面积计算三角形“S”的面积。



（2）使用公式计算梯形面积

$$T=（a+c）\times h\div 2$$

根据公式编写代码：print(6+9*3÷2)，运行代码的结果如图所示。



可以发现程序出现了错误，为什么会错呢？这是因为计算机键盘上没有除号（÷），在 Python 中使用斜杠（/）来表示除法。

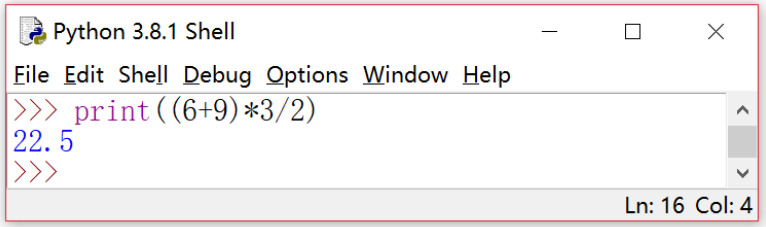
这个梯形的面积算对了吗？

这个答案是错误的。正确答案应该是 22.5 而不是 19.5 。既然错了，那如何修改这个程序呢？

也许你已经猜到了——加上一个括号就行！

是的，这是因为乘法和除法运算优先于加法运算，所以如果希望改变运算顺序，只需要在它两边加上括号即可，这和数学课上讲的是一样的。Python 会遵循正确的数学规则和运算顺序。

所以要使用如下写法。



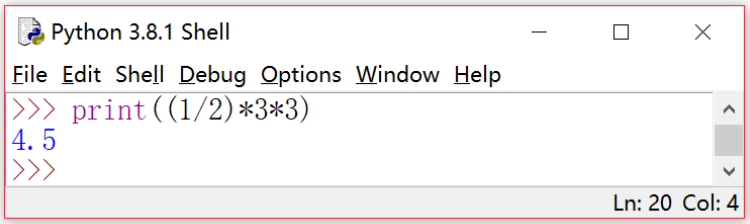
重点讲解**运算顺序**，运算符的优先级：和数学课上讲的一样。Python 会遵循数学规则和运算顺序。

三、使用公式计算三角形的面积

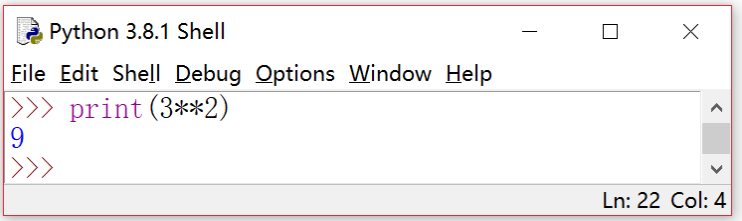
第二个问题是“使用公式计算等腰直角三角形的面积”，等腰直角三角形的面积公式如下：

$$S = \frac{1}{2}a^2$$

公式描述：公式中 a 为等腰直角三角形的直角边。通过观察图 2-1 可知，三角形的下边等于梯形的高，等于 3，三角形的上边等于 9-6，也等于 3。



在 Python 中，一个数的 N 次幂可以用双星号（**）表示，使用双星号（**）来表示一个数的 N 次幂就方便多了，例如 3 的 N 次幂，代码为 print（3**N）。下面编写 3 的平方，



四、归纳总结

可以使用提问的方式总结：

- 在 Python 中乘号是什么、除号是什么？
- 在 Python 中幂运算用什么符号？

第二课时

（Python 中的变量与代码注释）

一、变量

变量是 Python 存储数据的一个内存区域，可以理解为存放数据的地方，你可以用自己能看懂的方式来描述这个变量(给变量取名字，当然这个名字要让自己和别人都能看懂）。在程序中，直接用变量名来代替这些数据能让你的程序更加直观和简洁，也可以方便排查程序

	的错误。 例如计算梯形面积和三角形的面积使用变量与不使用变量对比。 （1） 不使用变量 ``` print((9+6)*3/2) print(9*3 - (9+6)*3/2) ``` （2） 使用变量 ``` c = 9 h = 3 a = 6 M = c * h T = (a+c)*h/2 S = M - T print(T) print(S) ``` 可能你会觉得第一种不使用变量的方式更方便，但是如果现在长方形的边长即梯形的下底 c 和高 h 发生了变化,需要你再次进行计算，会不会觉得第一种方式有些麻烦呢？ 而如果使用变量就只需要修改 c 和 h 的值即可，其他的代码不需要修改。 如果数据更加复杂，代码更多，使用变量来代替数据就会让代码更加简洁，也更好理解。 既然变量有这么多好处，那我们怎么用好变量呢？ 课程思政：在讲解变量命名规则时，强调规范命名如同生活中遵守规则，培养学生的规则意识。 **二、变量命名** 要用好变量需要从两个方面入手。 （1） 给变量取个好名字； （2） 给变量赋值。 现在有四个数据需要定义变量并且为他们取名字：1.性别，2.年龄，3.张三的手机号，4.家庭地址。

你觉得下列哪几个选项的命名方式比较好？

A. n12

B. name

C. age

D. 123_p+

E. shoujihao

F. number_of_zhangsan

G. home_address

H. ,./@#

比较好的命名方式是： B、C、F、G ；错误的命名方式是 D、H。

这是因为给变量取个好名字一般来说要遵守如下两个规则。

（1） **必须以大小写字母或者下划线开头**，不得以数字开头，比如 name1 可以但是 1name 就不能作为变量名，所以 D 和 H 是错误的。

（2） **简洁易读**，变量名要有意义，比如想要定义“年龄”的变量名，可以使用年龄的英文“age”，如果取名“a”或者“abc”，虽然没错，却增加了阅读代码的时间成本（要花额外的时间来弄懂这个变量名究竟指代什么、在程序中用来干什么）。根据这个规则 A 和 E 都不合适。

三、变量赋值

名字取好了，就要给变量赋值了，一个变量的自我修养，需要有自己的值，使用等于（=）符号可以给变量赋值。例如。

```
age = 18
number = 13798765432
```

这两行代码定义了两个变量，一个是年龄，一个是手机号，并且把右边的数据绑定给了左边的变量，然后就可以利用变量名代表这两个数据了。例如。

```
age = age +2
number = 18711011012
```

这两行代码对变量的数据进行了修改，给年龄加上了 2，然后给 number

变量重新赋值。

四、代码注释

讲解三种注释类型

（1） 单行注释；

（2） 行尾注释；

（3） 多行注释。

上机一：

形式：单独完成

完成第一、第二课时上的案例

上机一：

形式：单独完成

1.在学习注释的时候我们只注释了前四行代码，后面四行没有注释，请你帮忙补充最后四行的注释。

```
c = 9 #定义变量 c 表示长方形的长。梯形的下底
h = 3 #定义变量 h 表示长方形的宽，梯形的高
a = 6 #定义变量 a 表示梯形的上底
M = c * h #计算长方形面积
T = (a+c)*h/2
S = M - T
print(T)
print(S)
```

2. Python 中计算 3*3*3*3 的有几种方法？

3.请使用注释让下列代码输出：codejiaonang.com

```
print("Hello World")
#print("codejiaonang.com")
print("Hello")
print("Code")
```

（1）“计算”

教师边讲边演示简单算术运算代码编写

4.编写一个程序，把温度从华氏度（100F，200F，F 为华氏温度单位）转换为摄氏度，转换公式如下。

$$C = \frac{5}{9}(F-32)$$

第三、四课时

（人民币与越南盾兑换程序）

现在越来越多的人喜欢出国旅游，其中有很多人选择了越南，假设现在你的一位朋友想要去越南旅游，但是他身上没有越南盾（越南的货币），只有 5342 元人民币，他想请你编写代码帮他计算这些钱能换到多少越南盾。现在已知，人民币对越南盾的汇率为 1 人民币=3347.84 越南盾。

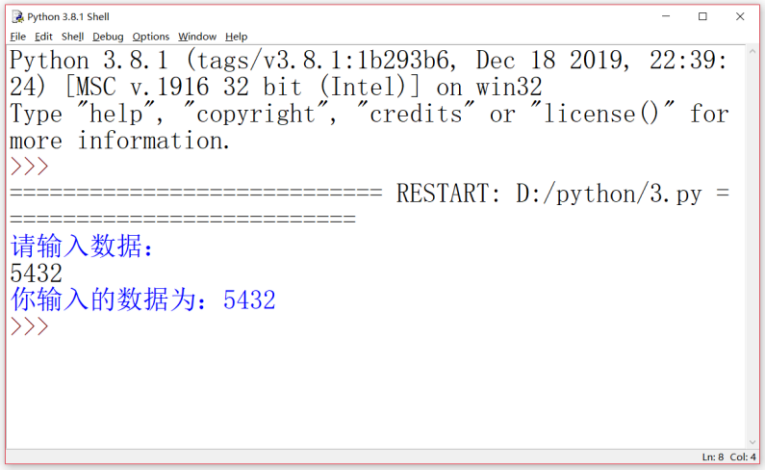
如果你还记得上节的内容，可能会编写出如下代码。

```
rmb = 5342 #定义人民币的值
exchange = 3347.84 #定义人民币对越南盾汇率
print(rmb*exchange) #打印兑换之后越南盾的值
```

但是这个时候，假设有十个朋友都想让你用编程计算他们的手上的人民币能换成多少越南盾，你该怎么办呢？

一、编写与用户交互的程序

在 Python 中有个内置函数 input()，可以使用这个函数从用户那里得到用户从键盘输入的数据。



二、计算用户输入的数据

有了 input()函数之后，就可以开发出一个无论有多少人，只要输入相应数值就能计算人民币兑换越南盾的程序了。

三、运行时错误

查看 Python 展示的运行时的错误信息，解决运行时错误

课程思政：在处理程序错误环节，鼓励学生面对错误不气馁，培养其坚韧不拔的科学精神，让学生明白挫折是学习的常态，坚持探索才能攻克难题。

数据类型

一、数据类型

我们人类能很容易分清楚整数，小数，文字的区别，但是计算机不能，计算机虽然很强大，不过有时候又比较傻，除非你告诉计算机，0 1 2 是数字，hello 这些是文字，否则计算机是弄不明白这些数据的区别的。

其实数据类型就是对常用的各种类型的数据进行了明确的划分，例如，想让计算机处理整数，那就传递整数给它，或者告诉它这是整数。想让计算机处理字符串，那就要传递字符串给它，而不能传递非字符串的数据。

二、整除

如何让数据的可读性更好一点呢？

可以让**整除**来帮忙，金额的总数整除 10000 就可以知道具体有多少万越南盾，在 Python 中整除使用//表示

三、取余数

Python 有一个特殊的操作符来计算整数相除的余数——**取余（modulus）操作符**，这个符号是**百分号（%）**例如 9 对 2 取余数

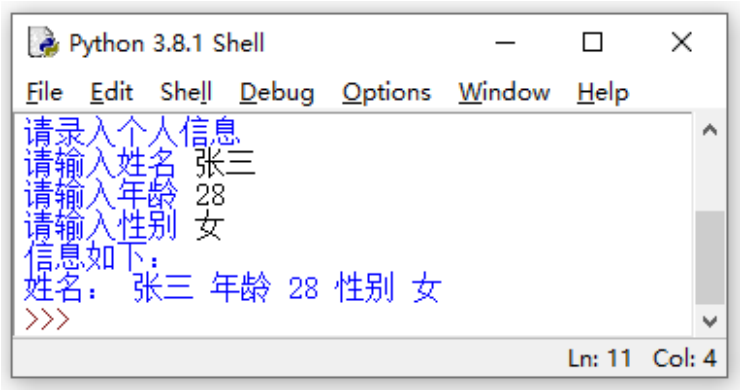
四、print()函数与逗号

要在打印数据的时候让两段数据拼接起来输出，需要在它们之间加上英文输入法的逗号

上机一：

形式：单独完成

题目：编写程序录入个人信息，效果如图所示。



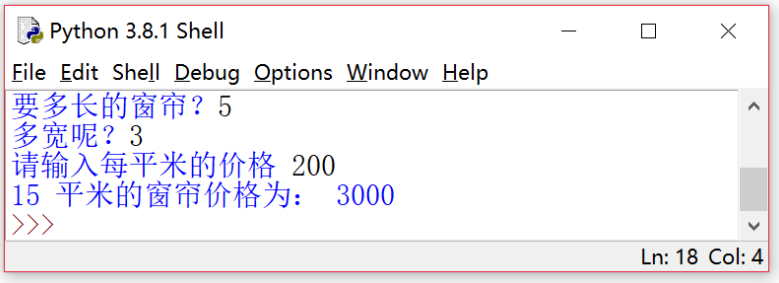
上机二：

形式：单独完成

题目：假设你现在开了一家窗帘店，顾客来买窗帘，你要编写一个程序可以完成如下功能：

（1） 询问顾客买多大的窗帘（单位是平方米）；

（2） 根据窗帘的定价告诉客户这样的尺寸需要多少钱，效果如图所示。



项目小结：

在本项目中我们使用 Python 解决了两个问题。

1. 使用 Python 计算梯形和三角形的面积；

2. 使用 Python 编写了一个可以和用户交互的人民币兑换

越南盾程序。

通过解决这两个 Python 问题，学习了许多 Python 基础知识，具体如下。

1. Python 中四大基本运算的符号：+ - * /，和数学中不同的是，Python 中乘号用*号代替，除号使用斜杠/代替；
2. Python 中实现幂运算使用两个星号（**）；
3. Python 中运算的优先级和数学中是一样的；
4. 变量的命名有两个规则。
   - 必须以大小写字母或者下划线开头；
   - 简洁易读，给变量取的名字需要有意义，让人容易理解。
5. Python 中的注释是为了解释代码，方便看代码的人理解，而没有其他具体的功能，执行代码的时候会忽略这些注释的内容；
6. 通过 input()函数可以获取用户输入的数据，可以利用它来编写一个可交互的程序；
7. Python 中运行时错误可以通过查看 Python 给出的错误提示，定位到相应的行来解决错误；
8. 数据类型就是对常用的各种类型数据进行明确的划分；
9. 通过 type()函数可以获取数据的数据类型；
10. 使用 int()函数可以将其他数据类型转换为整数类型；
11. Python 中的两个运算符，整除（//）与取余（%）；
12. 数据想要拼接输出可以在 print()函数中使用英文字符的逗号来拼接两段数据。

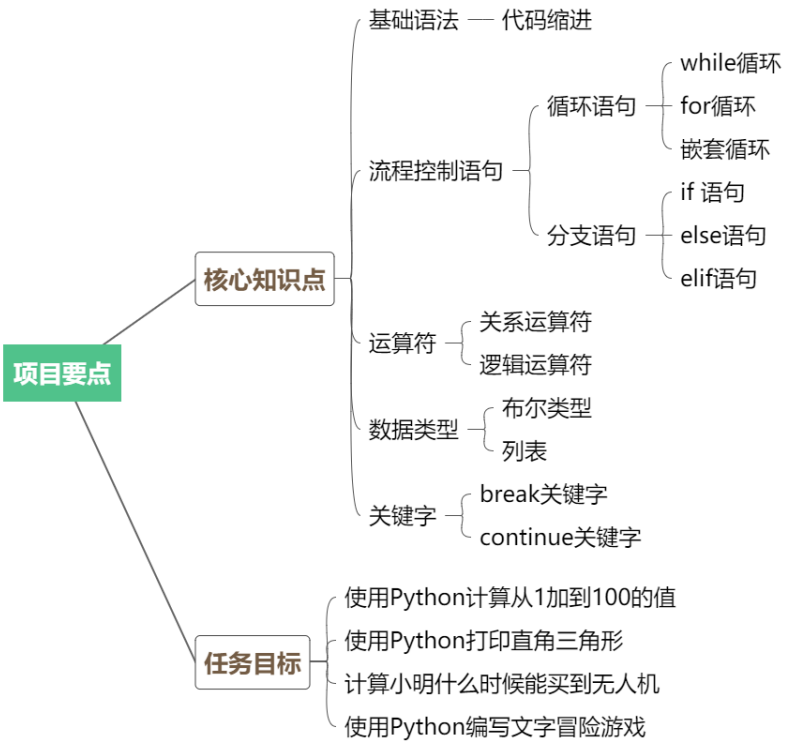
作业	**练习题/操作题** 1. **计算三角形面积：** o 编写一个程序，输入三角形的底和高，计算并输出其面积。 o 使用公式面积 = 0.5 * 底 * 高。 2. **汇率转换程序：** o 编写一个程序，输入人民币金额，将其转换为美元（假设汇率为 1 人民币=0.15 美元）。 o 使用 input()函数获取用户输入，并输出转换后的金额。
教学后记	本次教学，学生对用 Python 解决数学问题兴趣浓厚，在运算符和表达式学习上进步明显。但部分学生在变量命名规范运用及处理复杂数据类型转换时存在困难。后续教学应增加针对性练习，强化薄弱环节，提升学生编程能力。

<table>
<tr><td>课题名称</td><td>项目三：解决重复的事情
重复的事情交给计算机——循环与判断</td><td>计划学时</td><td>4 学时</td></tr>
<tr><td>教学内容</td><td colspan="3">课程介绍
使用 Python 计算从 1 加到 100 的值
使用 Python 打印直角三角形
计算小明什么时候能买到无人机
使用 Python 编写文字冒险小游戏</td></tr>
<tr><td>教学目标及基本要求</td><td colspan="3">1. 掌握 Python 的基础语法（代码缩进）
2. 掌握 Python 中的流程控制语句（循环语句、分支语句）
3. 掌握 Python 中的运算符
4. 掌握 Python 的数据类型（布尔类型、列表）
5. 掌握 Python 中的关键字（break 关键字、continue 关键字）</td></tr>
<tr><td>教学重点</td><td colspan="3">1. Python 的基础语法
2. Python 流程控制语句（循环语句、分支语句）
3. 运算符
4. 数据类型</td></tr>
<tr><td>教学难点</td><td colspan="3">循环语句，多重循环语句，分支语句</td></tr>
<tr><td>教学方式</td><td colspan="3">教学做一体化</td></tr>
<tr><td>教学过程</td><td colspan="3">第一课时
（课程介绍、编写从 1 加到 100 的程序）
一、项目场景，导入课程
很多人不愿意一直做重复的事情，会觉得枯燥无聊。人们会找到很多工具来代替重复劳动，所以现在大多数家庭都有洗衣机和其他自动化机器。在编程中也一样，我们肯定不愿意一直编写重复的代码。在编程中我们如何避免编写重复的</td></tr>
</table>

代码呢？答案是**循环**，我们可以利用**循环**语句来减少重复的代码从而提升编程效率。

做重复的事情一般要遵循一定的规则，例如洗衣机洗衣服要提前设置清洗时间和水位等规则，然后洗衣机就会按照这些规则去工作。那么如何让计算机程序也能遵循提前设置好的规则呢？答案是**判断语句**，判断语句可以对某个条件进行判断，然后根据判断的结果决定执行什么代码。

在项目开发中经常需要使用**循环**和**判断**，而且它们很多时候都是同时出现，本项目将带领你学习 Python 中的循环与判断语句，掌控程序执行的流程。



二、编写程序计算从 1 加到 100 的值

请你编写 Python 程序，计算从 1 加到 100 等于多少

三、while 循环

使用 While 循环计算重复输出特定语句。

```
i = 1
```

```
while(i <= 6):  # 注意这里是英文输入法的冒号
    print("做了",i,"个俯卧撑")
    i = i + 1   # 每次进行循环的时候 i 的值加 1
```

四：缩进

while 循环代码编写的风格与之前的代码有一些不同。while 循环中的代码需要进行**缩进。**

上机一：

形式：单独完成

使用 While 循环计算从 1 加到 100 的和

上机二：

形式：单独完成

下列程序有错误，请将程序修改成能正常执行的程序。

```
  print("www.codejiaonang.com")
print("codejiaonang")
      print("hello world!")
```

上机三：

形式：单独完成

3. 编写一个程序显示一个简单乘法表，输出如下。

```
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
```

一：嵌套循环

主要讲解嵌套循环的使用方法

嵌套循环就是在一个循环体中嵌入另一个循环。

嵌套循环的基本格式如下。

```
while(条件):
    <外层循环执行的语句>
    while(条件):
        <内层循环执行的语句>
```

使用嵌套循环输出直角三角形

二：FOR 循环

Python 中还有一种方式能实现循环的功能，并且有时候用起来比 while 循环更简单，这种循环就是 for 循环

讲解如何使用 for 循环。

例如使用 for 循环“做俯卧撑”。

```
for looper in [1,2,3,4,5]:
    print("做了", looper,"个俯卧撑")
```

三：列表

在 Python 中要表示所有的朋友可以这样写。

```
friends = ["张三"、"李四"、"王五"、"赵六"]
```

如果让你写上每天花了多少钱，可能你会这样写。

100、500、150...

在 Python 中要表示每天花了多少钱可以这样写。

```
spend = [100,500,150]
```

列表是 Python 中的数据类型 friends 变量和 spend 变量都是 Python 中的列表，**列表可以存储一组数据**。

四：Range 函数

在 Python 中使用 range() 函数，就可以只输入开始值和结束值，然后 range() 函数就会创建这个中间的所有值， range() 函数会将这些值构建成一个列表。

例如将“做俯卧撑”的示例改为 range() 函数实现。

```
for looper in range(1,6):
    print("做了", looper,"个俯卧撑")
```

五：使用 for 循环打印直角三角形

学习了 for 循环和 range()函数，要打印一个直角三角形就很方便了，编写代码如下。

```
for i in range(1,10):
    for j in range(i):
        print("*",end = " ")
    print("")
```

第三、四课时

（实训课）

自己独立编写第三课时的代码，完成课后练习

1. 请问下列程序的运行之后，count 的值为多少？

```
i = 3
j = 0
count = 0
while(i > 0):
    j = 1
    while(j < 3):
        count = count + 1
        j = j + 1
    i = i - 1
print(count)
```

2. 输出一个 99 乘法表（请分别使用 for 循环和 while 循环实现），效果如图 3-16 所示。

```
1 x 1 = 1
1 x 2 = 2   2 x 2 = 4
1 x 3 = 3   2 x 3 = 6   3 x 3 = 9
1 x 4 = 4   2 x 4 = 8   3 x 4 = 12   4 x 4 = 16
1 x 5 = 5   2 x 5 = 10   3 x 5 = 15   4 x 5 = 20   5 x 5 = 25
1 x 6 = 6   2 x 6 = 12   3 x 6 = 18   4 x 6 = 24   5 x 6 = 30   6 x 6 = 36
1 x 7 = 7   2 x 7 = 14   3 x 7 = 21   4 x 7 = 28   5 x 7 = 35   6 x 7 = 42   7 x 7 = 49
1 x 8 = 8   2 x 8 = 16   3 x 8 = 24   4 x 8 = 32   5 x 8 = 40   6 x 8 = 48   7 x 8 = 56   8 x 8
1 x 9 = 9   2 x 9 = 18   3 x 9 = 27   4 x 9 = 36   5 x 9 = 45   6 x 9 = 54   7 x 9 = 63   8 x 9
```

图 3-16　99 乘法表

3. 输出两个直角三角形（请分别使用 for 循环和 while 循环

实现），效果如图 3-17 所示。

```
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * * * *
-------------- 我是华丽的分界线 --------------
* * * * * * * * *
* * * * * * * *
* * * * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
```

图 3-17　正三角形与倒三角形

（小明什么时候能买到无人机）

今年 7 月，小明参加的社团想要组织一次户外活动，小明作为社团的宣传部长，他想要购买一台无人机在这次户外活动中进行航拍，但是小明没有足够的资金购买无人机，所以他打算参加勤工俭学，利用勤工俭学的收入来购买无人机。

现在假设无人机的价格为 **X 元**，并且每月涨价 **10%**。小明勤工俭学的收入为 **M 元**，每月上涨 **15%**。并且小明能够将每月勤工俭学的收入全部存下来。请你编写一个交互式程序，判断小明从 2 月份开始到 7 月份之前能不能买到无人机。

一、If 判断语句

在生活中，我们经常需要先做判断，然后才决定是否要做某件事情，比如说在英语课上，英语老师跟同学们说，如果这次英语口试作业打了 100 分，将奖励一本最新的英文杂志。

对于这种“需要先判断条件，条件满足后才执行的情况”。可以使用 if 条件语句来实现。

在 Python 中使用 if 语句的格式如下。

```
if(条件):
    <条件为真执行的语句>
```

二、关系运算符与布尔类型

你有没有注意到，在上面的例子中，判断 score 是否等于 100 时，使用的不是“=”号，而是 “==”号。

你可能会怀疑这样是不是写错了，其实这两个符号在 Python 中都是存在的，只是他们的含义不同，前者（=号）是赋值操作，后者（==号）是在做测试（测试两个数是否相等），所以 Python 使用了两种不同的符号。

要给变量赋值，Python 使用了一个等号（=）。

例如。

```
name = "张三"  # 一个等号是赋值操作
```

要测试是否相等，Python 使用了一个双等号（==），代码如下。

```
print(3 == 5)
print(5 == 5)
```

输出结果。

```
False
True
```

注意：混淆 = 号 和 == 号，是很多初学者容易犯的错误！

在这里==号被叫做比较操作符（也叫作关系运算符），比较操作符会得出一个结果，这个结果只有固定的两个值： **True 和 False**，即**真**和**假**的意思。

True 和 False 是 Python 中一种数据类型的值，这个数据类型是：**布尔类型**，布尔类型只有这两个值。

注意：这里 True 和 False 中的 T 和 F 都是大写！

elif 与逻辑运算符

一、elif 语句判断结果为假

现在我们已经知道了判断的结果为真（True），Python 会做些什么，但是如果第一次判断结果为假（False），还想继续进行判断应该怎么办呢？

例如：考试成绩的等级，在 60 分以下，打印不及格，60-80 分打印良，80-100 分打印优秀。

如果有这样的场景，可以使用 elif（这是 else if 的简写）即再进行一次判断。例如。

```
score = 75
if score < 60 :
    print("不及格")
elif 60 <= score < 80 :
    print("良")
elif 80 <= score <= 100 :
    print("优秀")
```

运行结果：良。

二、逻辑运算符

在编程中，经常需要判断多个条件。

在 Python 中使用逻辑运算符可以支持多个条件的判断。

现在假设变量 a 为 True，b 为 False，它们的逻辑关系可以用。

运算符	逻辑表达式	描述	实例
and（并且）	x and y	布尔"与"，如果 x 为 False，x and y 返回 False，否则它返回 y 的值。	(a and b) 返回 Flase。
or（或者）	x or y	布尔"或"，如果 x 和 y 其中一个是 True，则返回 True，如果 x 和 y 都是 False，则返回 False。	(a or b) 返回 True。
not（否定）	not x	布尔"非"，如果 x 为 True，返回 False 。如果 x 为 False，它返回 True。	not(a and b) 返回 True

使用 Python 编写文字冒险游戏

一、break 关键字

某些情况下，我们在编写循环语句的时候可能需要提前结束

循环，这个时候可以使用 break 键字。

例如。

```
i = 1
while i <= 10:
    print(i)
    i = i + 1
    if i == 5:
        break
```

运行结果如下。

```
1

2

3

4
```

运行结果只输出了 1 ~ 4 ，这是因为当 i 等于 5 的时候我们使用 break 语句提前结束了这个循环。

break 翻译过来是“打断”的意思，放在 Python 程序中的作用就是：**结束当前循环**。

但是如果不止一个循环，那 break 会对多个循环语句有什么影响呢？

二、continue 关键字

continue 和 break 类似，continue 关键字的用途是：**结束本次循环，开始下一个循环**，也就是忽略 continue 语句之后的语句，执行循环体的下一次循环。

例如。

```
i = 1
while i <= 8:
    if i == 5:
        i = i + 1
        continue
    print(i)
    i = i + 1
```

运行结果如下。

1

2

3

4

6

7

8

可以发现输出的结果没有 5，这是因为在 i 值为 5 的时候使用 continue 语句跳出了当前循环，直接执行下一次循环了。

continue 关键字的用途是：结束一次循环事件，开始下一个循环事件，也就是忽略该语句之后的语句，执行循环体的下一次循环。

可能你觉得已经掌握了 break 和 continue 的用法，但是不测一测怎么知道呢？

第九课时

（无限循环与文字冒险游戏编写）

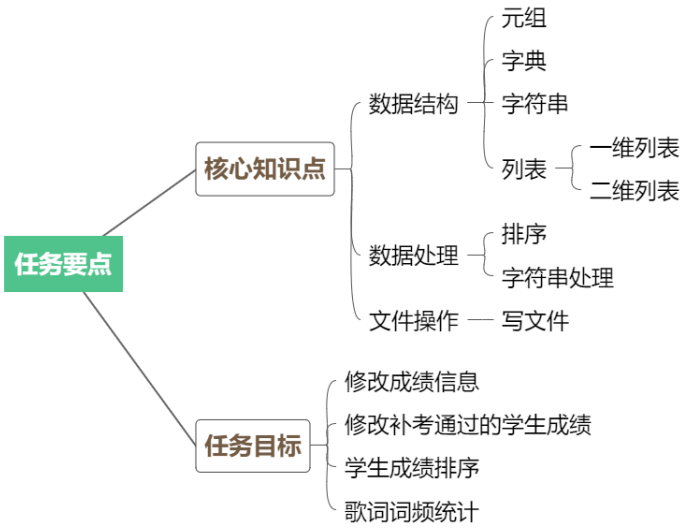
一、无限循环

无限循环也叫死循环，很多时候死循环都结合 continue 和 break 一起使用。

例如，设计一个循环，接收一个数字，这个程序只有两种情况才能退出。

```
i = 0
while True:
    num = int(input("请输入一个数字："))
    if num == 1:
        i = i + 1
        continue
    elif num == 2:
        break
    if i >=4:
```

	`` break ``` 运行这段代码会发现，只要一直输入 1 程序就会一直运行，输入了四次 1，然后输入其他数字，或者直接输入 2 程序就结束了。 这段代码中，while True 就是一个无限循环，退出由 break 控制，跳过此次循环执行下一次循环由 continue 控制。
作业	**猜数字游戏：** • 编写一个猜数字游戏程序，程序随机生成一个 1 到 100 之间的整数，用户通过输入猜测数字，程序提示用户“太大”或“太小”，直到猜中为止。
教学后记	本章重点介绍了循环和判断语句的使用。学生对 for 和 while 循环的理解逐渐加深，但在嵌套循环的逻辑上仍存在一些困难。部分学生在编写猜数字游戏时，对 break 和 continue 关键字的使用不够熟练。后续课程中需要通过更多实例帮助学生理解循环和判断语句的结合使用。

课题名称	项目四：处理身边的数据-数据类型	计划学时	4 学时
教学内容	课程介绍 使用 Python 修改成绩信息 使用 Python 修改补考通过的学生成绩 学生成绩排序 歌词词频统计		
教学目标及基本要求	1. 掌握 Python 数据结构（元组、字典、字符串、列表） 2. 掌握 Python 的数据处理 3. 掌握 Python 操作文件		
教学重点	1. 掌握 Python 数据结构（元组、字典、字符串、列表） 2. 掌握 Python 的数据处理 3. 掌握 Python 操作文件		
教学难点	Python 操作文件		
教学方式	教学做一体化		
教学过程	第一、二课时 （课程介绍、统计成绩信息） 一、项目场景，导入课程  		

问：请问如何编写代码保存班上 5 位同学的考试成绩？

答：简单，定义五个变量就行。

问：现在班上有 100 位学生，需要你保存他们的成绩，计算他们的成绩平均值、最高分，并且要将成绩排序，该怎么办？

答：emmm……

相信你已经发现了，如果仅仅使用 Python 的基础数据类型，不能存储大量的数据。

Python 提供了列表（list）与字典（dictionary）这两种数据类型来帮助开发者解决临时存储大量数据的问题。Python 中的列表和字典，可以将很多数据存储在一起，放在某种“组”或者“集合”中，这样就能对整个集合进行某些处理，也可以更容易地记录一组数据。有一类集合叫作**列表（list）**，另一类叫作**字典（dictionary）**。在这个项目中，我们将利用列表与字典处理更多的数据。

二、统计成绩信息

你的第一个任务是使用 Python 统计学生的成绩信息，需求如下。

（1） 程序可以依次录入 6 位学生的成绩；
（2） 计算学生成绩的平均分；
（3） 找出最高分；
（4） 对所有学生的成绩进行降序（从大到小）排序。

要完成统计成绩信息的任务，第一步需要先解决数据存储的问题。而在 Python 中使用**列表**来存储同一种类型的数据非常方便。

三、创建列表

在生活中存储东西需要使用容器（比如存储水需要水杯），在编程中也是一样，要存储大量数据可以使用列表作为容器，列表由**一系列按特定顺序排列的元素组成**。用方括号（[]）来表示，用逗号来分隔其中的元素。

例如定义一个名为 friends 的列表，表示所有的朋友，如图 4-1 所示。

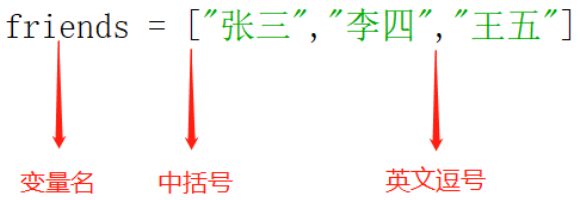


图 4-1　定义列表

如果我们要创建一个有规律的**数值列表**，可以使用更方便的方式来创建，代码如下。

```
arr = [i*2 for i in range(5)]
print(arr)
```

运行结果：［0，2，4，6，8］

这种方式也被称为**列表推导式**。

四：获取列表中的数据

列表中每个元素都是有序地排列，每个元素都有自己的位置编号（索引值）。我们可以使用列表名加索引值的中括号来提取相应位置的元素，如图 4-2 所示。

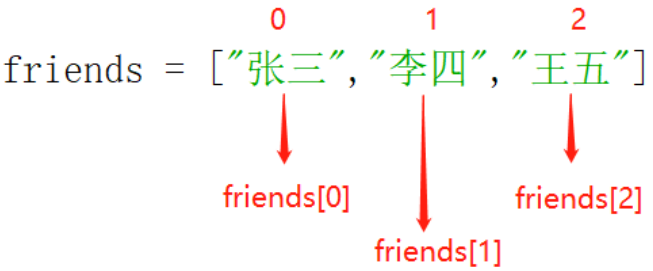


图 4-2　列表元素的位置

要获取列表中某个元素可以通过：列表名[索引值］来获取。

例如。

```
friends = ["张三","李四","王五"]
print(friends[0])
```

运行结果：张三 。

要获取列表的第一个元素，需要使用 friends[0]，可能你会有疑问，为什么索引值从 0 开始而不是从 1 开始？

五：获取列表中的多个元素

列表支持一次性提取多个元素。

提取多个元素可以使用**列表的切片**（类似数学的区间）操作，通过列表名[startIndex : stopIndex]即可提取多个元素。

六：向列表添加/删除元素

给列表添加元素可以使用 append()函数。

例如。

```
friends = ["张三","李四","王五"]
friends.append("赵六")  # 添加元素
print(friends)
```

运行结果：['张三', '李四', '王五', '赵六']。

要删除元素可以使用：del 列表名[索引值] 或者 列表名[starIndex: stopIndex]

七：录入学生成绩

掌握了列表的知识，就能完成第一步的工作了。

录入学生成绩，代码如下。

```
i = 0
student_scores = [] # 定义空列表存储学生成绩
while i < 6: # 使用循环控制录入成绩的次数
    print("请输入第",(i+1),"位学生成绩")
    score = int(input()) # 获取某位学生成绩
    student_scores.append(score) # 将每次录入的成绩存入列表
    i = i + 1
print(student_scores)
```

简单统计计算
一、计算平均分

平均分 = 总分 ÷ 学生数，要计算平均分，我们需要先求出总分。学生的成绩信息存储在列表中，可以使用循环将列表中的元素取出来，然后求和，最后计算平均分。

```
i = 0
student_scores = []
while i < 6:
```

```
        print("请输入第",(i+1),"位学生成绩")
        score = int(input())
        student_scores.append(score)
        i = i + 1
# 求总分
sum = 0
for score in student_scores:
        sum = sum + score
print("总分为：",sum)
print("平均分为：",sum/len(student_scores))
```

二、计算最高分

太阳底下无新事，在编程中求最高分就像“打擂台”，我们可以定义一个擂主，有若干个挑战者，挑战者如果战胜了这个擂主，这位挑战者就成为擂主，直到最后没有挑战者，擂主就是最强的，代码如下。

```
max = student_scores[0] # 定义最大值为列表第一个元素
#使用 for 循环实现
for score in student_scores:
        if max < score:
                max = score # 挑战者成为擂主
print("最高分为：",max)

#使用 while 循环实现
i = 1
while i < len(student_scores):
        if max < student_scores[i]:
                max = student_scores[i] #挑战者成为擂主
        i = i + 1
print("最高分为：",max)
```

三、对成绩排序

对 Python 的列表进行排序很简单，只要使用 sort()函数即可。

```
student_scores.sort()
print(student_scores)
```

运行结果：[50, 70, 78, 89, 90, 100] 。

但是这个结果不对，任务的要求是降序（从大到小）排列，我们使用 sort()函数得到的结果是升序的。

要得到正确结果可以使用 reverse()函数，将列表进行反转即可。

```
student_scores.sort()
```

```
student_scores.reverse()
print(student_scores)
```

运行结果：[100, 90, 89, 78, 70, 50] 。

当然还有更方便的方式，直接在 sort()函数中加上一个参数即可。

```
student_scores.sort(reverse = True)
print(student_scores)
```

这个参数名为 reverse，它会按照你的意愿，将结果翻转过来。

编写交互程序，获取用户输入的 6 个名字数据并存入列表中，然后对列表进行如下操作。

- 第一步：将该列表末尾的元素删除，并将这个被删除的元素值保存到 deleted_list 变量；
- 第二步：将 deleted_list 插入到第一步删除后的列表索引位置为 2 的地方；
- 第三步：将第二步处理过的列表索引位置为 1 的元素删除；
- 打印输出 deleted_list 变量；
- 打印处理之后的列表。

修改补考通过的学生成绩

一、创建二维列表

二维列表：以一维列表作为元素的列表。

要使用二维列表则需要定义它，定义二维列表有两种方式。

（1） 静态方式创建，在已知所有二维列表数据的时候可以使用静态方式；

```
stu_scores = [
    [0,0,0],
    [1,1,1],
    [2,2,2]    # 结尾不要有逗号
]
```

（2） 动态方式创建，在不知道二维列表的数据时可以使用动态方式定义。

例如：使用循环方式创建一个值为 0 的二维列表。

```
stu_scores = []
for i in range(3):
    temp_list = []
    for j in range(3):
```

```
            temp_list.append(0)
        stu_scores.append(temp_list)
print(stu_scores)
```

运行结果：[[0, 0, 0], [0, 0, 0], [0, 0, 0]] 。

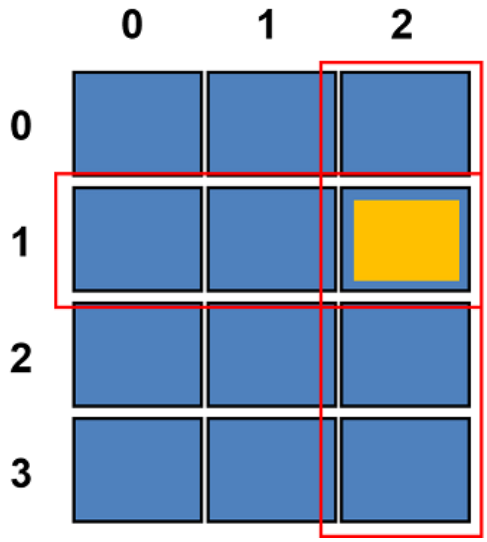
二、修改二维列表的值

要修改二维列表中某个元素的值，首先需要对列表中的元素进行访问。

访问二维列表中的元素分为两个步骤。

（1） 确定要访问的元素在哪一行；
（2） 确定要访问的元素在哪一列。

例如我们要访问二维列表中第二行第三列的元素



三、遍历二维列表

看到这个标题，可能你会有疑问，“遍历”是什么意思？

在编程中经常要用到遍历，你可以将遍历列表理解为：**将列表中所有的元素都看一遍**。

下面我们来遍历二维列表，循环输出列表中的数据，代码如下。

```
stu_scores = [
    [0,0,0],
    [1,1,1],
    [2,2,2]
]
```

```
for i in range(len(stu_scores)):
    # 定位每一行的元素，len(stu_scores[i]代表每一行的长度
    for j in range(len(stu_scores[i])):
        print(stu_scores[i][j],end = " ")
    print("")
```

运行结果如下。

```
0 0 0
1 1 1
2 2 2
```

上面这段代码有双重循环，第一重循环遍历的是行，遍历次数为len(stu_scores)次，len(stu_scores)代表的是 stu_scores 列表的长度。

第二重循环遍历的是当前行的列，len(stu_scores[i])代表的是当前行的长度，通过双重循环就可以遍历整个二维列表了。

四、修改补考通过的学生成绩

了解了二维列表，通过三个步骤即可修改补考通过的学生成绩。

（1） 定义学生成绩列表；
（2） 遍历所有学生成绩；
（3） 判断成绩是否低于 60 分。如果是则改为 60 分。

基于这三个步骤，编写代码如下。

```
student_scores = [
    [90,50,49,100],
    [80,70,7,80],
    [50,55,44,61]
]

for i in range(len(student_scores)):
    for j in range(len(student_scores[i])):
        if student_scores[i][j] < 60:
            student_scores[i][j] = 60
print(student_scores)
```

运行结果：[[90, 60, 60, 100], [80, 70, 60, 80], [60, 60, 60, 61]] 。

第三、四学时

学生成绩排序

一、创建字典

除了列表之外，在编程中你会经常想以另一种方式组织元素——**将某个值和另一个值关联起来**，例如汽车与车牌号，就是通过车牌与车关联起来。

Python 中的字典就是一种**将两个数据关联起来的方式**，被关联的两个数据，一个被称为**键（key）**，另一个被称为**值（value）**。

字典中的每一个条目（item），都由一个键（key）和一个值（value）组成，它们合起来被称为键值对（key-value）。

例如创建一个姓名与手机号码对应的字典，可以看到姓名是键（key）手机号码是值（value），如图 4-6 所示。

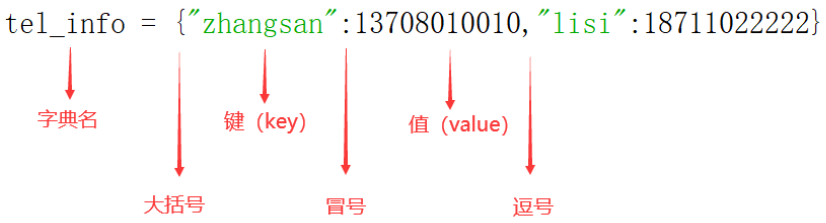


图 4-6　创建字典

二、添加数据

给字典添加数据很简单，使用字典名[key] = value 即可

三、删除数据

使用 del 字典名[key]可以删除字典中的数据。

四、字典排序

列表可以使用 sort()函数进行排序，字典要进行排序可以使用 sorted()函数。

五、元组

我们学过中括号[]和大括号{}，中括号表示的是列表，大括号表示的是字典。

当圆括号包裹数据的时候，表示的是 Python 中的一种数据类型——**元组（tuple）**。

Python 的元组与列表类似，不同之处有两点。

（1） 元组使用圆括号()，列表使用方括号[]；
（2） 元组中的元素不可更改，列表中的元素可以更改。

创建元组很简单，只需要在圆括号中添加元素，并使用逗号分隔即可。

例如。

```
tup1 = (1, 2, 3)
tup2 = ('zhangsan','lisi')
```

六、掌握三个函数
keys()和 values()、extend()函数
七、完成成绩排序程序

实训操作

1. 我们给学生成绩进行排序是通过总分排序，你能编写一个可以根据英语成绩降序排序的程序吗？

2. 模拟登陆程序，有用户数据如下。

用户名	密码
zhangsan	123456
xiaoma	12345678
laoma	87654321
lisi	admin123

请创建字典保存用户数据，然后编写程序实现如下功能。

（1） 用户输入用户名和密码可以进行登录；
（2） 用户名和密码与用户数据匹配则输出：登录成功；
（3） 若用户名与密码不匹配则输出：登录失败，请检查用户名和密码；
（4） 若用户名不存在则输出：用户名不存在，请检查输入是否有误。

歌词词频统计

首先让我们一起来看一段歌词。

Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.

Happy Birthday To You;
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;

相信你肯定知道这首生日快乐歌，这个小节的任务就是统计其中每个单词出现的次数，并保存至字典中。

要实现生日快乐歌中单词的统计可以分为两个步骤。

（1） 对相同的单词进行统计；
（2） 将统计的结果保存至字典中。

要完成这个任务，我们还需要学习一个技能——字符串处理。

一、字符串

在之前的项目中，我们反复提到过字符串，但是没有做很详细的探讨，现在就来和字符串做一次深入交流吧！

我们人类能很容易分清楚整数，小数，文字的区别，但是计算机它不能。计算机虽然很强大，不过有时候又比较傻，除非你告诉他，0 1 2 是数字，hello word 是单词，否则计算机是弄不明白的。

在计算机中要表示英文，中文，日文等等这些文字，我们就要用到**字符串**这种数据类型。

字符串是 Python 中最常用的数据类型。可以使用**单引号（’）**或**双引号（”）**来创建字符串。

例如。

```
str1 = 'hello '  # 使用单引号定义字符串
str2 = "world"    # 使用双引号定义字符串
str3 = str1 + str2 # 字符串拼接
print(str3)
```

二、字符串常用操作

字符串常用的操作有四种。

（1） 字符串查找；

（2） 字符串截取；

（3） 字符串替换；

（4） 字符串分割。

三、字典与字符串

很多时候字典都是和字符串一起使用的，而且字符串通常会作为字典的键（key）。

在字典中**键（key）是唯一的**，如果同时往字典中添加两个相同的 key 会发生什么呢？

四、编写歌词统计程序

```
str_info ="""Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday To You;
Happy Birthday dear my friend.
Happy Birthday To You;"""

#定义字典存储词频数据
word_dic = {}
str_info = str_info.replace(";"," ").replace("."," ")
word_list = str_info.split() # 将数据分割
for word in word_list:
    # 如果单词已经在字典中，则在原有的基础上加 1，否则初始值为 1
    if word in word_dic.keys():
        word_dic[word] = word_dic[word] + 1
    else:
        word_dic[word] = 1

#排序
```

sorted_word_list = sorted(word_dic.items(),key = lambda item:item[1],reverse = True)

print(sorted_word_list)

运行结果。

[('Happy', 16), ('Birthday', 16), ('To', 12), ('You', 12), ('dear', 4), ('my', 4), ('friend', 4)]

项目总结与课后练习

在本项目中，我们使用 Python 编写了四个程序。

1. 统计成绩信息；
2. 修改补考成绩；
3. 对学生成绩进行排序;
4. 歌词词频统计。

通过编写这些程序我们学到了许多 Python 基础知识，具体如下。

1. 一个新的数据类型——列表（list），知道了列表是由一系列按特定顺序排列的元素组成的集合；
2. 要访问列表中的元素需要通过 列表名[索引值]，并且索引值是从 0 开始计算的；
3. 给列表添加数据可以使用 append() 函数，删除列表中的元素可以使用 del 关键字；
4. 对列表进行排序可以使用 sort() 函数，sort() 函数默认是升序的，如果要降序可以使用 reverse ；
5. 在很多时候我们的数据是一个表格，这个时候可以用二维列表来表示这类数据，二维列表就可以看做以一维列表作为元素的列表；
6. 了解了编程中经常提到的“遍历”，遍历列表就可以理解为将列表中的元素都看一遍；
7. Python 中字典是一种将两个数据关联起来的方式，一个数据称为键（key），另一个称为值（value），字典中的每一个条目（item），都由一个键（key）和一个值（value）组成，他们合起来被称为键值对；
8. 字典添加数据直接使用 字典名[key] = value 的方式；
9. 字典不能使用 sort() 函数进行排序，但是可以使用 sorted() 函数进行排序，sorted() 会将排好序之后的数据作为新列表返回；
10. 字典是无序的；
11. 圆括号包裹的数据叫作元组。

12. 元组中的元素不可更改；
13. 使用单引号和双引号都可以创建字符串；
14. 三个字符串常用函数可以帮助我们操作字符串：1.find()查找子串位置，2.replace()替换子串，3.split() 分割字符串；
15. 很多时候字典都是和字符串一起使用的，字典中添加数据时可以先使用 key in dic.keys() 检查 key 是否存在。

课后练习

1. 列表 nums = [2, 7, 11, 15, 1, 8]，请你编写程序找到列表中任意相加等于 9 的元素集合，如：[(0, 1), (4, 5)]。
2. 有股市数据如表 4-4 所示。

表 4-4　股市数据表格

Date	Open
2014-01-01	79.382858
2014-02-01	71.801430
2014-03-01	74.774284
2014-04-01	76.822861
2014-05-01	84.571426
2014-06-01	90.565712
2014-07-01	93.519997
2014-08-01	94.900002
2014-09-01	103.059998
2014-10-01	100.589996
2014-11-01	108.220001
2014-12-01	118.809998
2015-01-01	111.389999

	请将这些数据存入 csv 文件中，然后编写代码读取文件信息，具体要求如下。 （1） 第一行代表之后每一行逗号分隔的各个数据的含义，这里我们只要知道 Date，Open 的含义，分别代表日期，开盘价； （2） 读取文件数据，将日期和开盘价保存至字典中，日期作为 key，开盘价作为 value； （3） 编写可交互式的方式，让用户可以通过日期查询到相关开盘价。
思作业	创建一个字典 student = {'name': '张三', 'age': 20, 'score': 90}，使用 update()方法更新 score 为 95，然后打印更新后的字典。
教学后记	本章介绍了 Python 中的数据类型和数据结构，学生对列表和字典的基本操作掌握较好，但在列表推导式和字典的高级操作上存在一些困难。部分学生对字典的 keys()、values()和 items()方法的理解不够深入，导致在数据处理时出现错误。后续课程中需要通过更多实例帮助学生巩固数据结构的操作。

<table>
<tr><td>课题名称</td><td>项目五：代码利用让代码更精简——函数与模块</td><td>计划学时</td><td>4 学时</td></tr>
<tr><td>教学内容</td><td colspan="3">函数的定义与调用
return 的作用和用法
代码复用的思想
常用模块的使用</td></tr>
<tr><td>教学目标及基本要求</td><td colspan="3">1. 掌握 Python 的函数的定义与调用
2. 掌握 return 的用法和作用
3. 了解函数的设计原则
4. 了解模块的概念
5. 掌握模块的使用流程
6. 熟悉常用模块的使用</td></tr>
<tr><td>教学重点</td><td colspan="3">1. 函数的定义与调用
2. return 的用法
3. 函数的设计原则
4. 模块的使用流程与常用模块的使用</td></tr>
<tr><td>教学难点</td><td colspan="3">1. 函数的设计原则
2. 函数与模块嵌套使用</td></tr>
<tr><td>教学方式</td><td colspan="3">教学做一体化</td></tr>
<tr><td>教学过程</td><td colspan="3">第一课时
（函数的定义与调用）
一、创设情境，导入课程
每当双 11 的时候，都可以使用无门槛红包，而且每满 400 元可以减 50 元。大家都会趁着活动清空自己的购物车，小明也不例外。请学生编写一个程序，该程序能够根据购物车的原始总价以及无门槛红包的金额，打印出双 11 当天清空购物车所需要花费的金额。此时学生应该很快能写出正确的程序。但程序虽然正确，但并不“好”，因为可扩展性不强，造成代码冗余的概率比较大。</td></tr>
</table>

	课程思政：做事应精益求精。 **二、介绍 Python 中，函数的定义与调用** 由数学中的函数的特点，从而引出 Python 中函数的特性（计算逻辑不变）。引导学生了解定义与调用的先后关系。强调定义函数时需要记住 4 个要素：def、函数名、参数列表、函数体。通过简单的加法示例来强化以上 4 个要素。 万事俱备，只欠东风。定义好了函数就可以着手调用了。先让学生自己根据之前所学的知识来调用刚刚定义好的加法函数，再向学生讲解函数的调用流程和注意事项。 **三、归纳总结，布置随堂练习** （1）回顾上课前的学习目标，对本节课知识点进行总结。 教师带领学生总结本节课需要了解或掌握的知识点。 （2）布置随堂练习，检查学生掌握情况。 结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。 **第二课时** **一、单一职责原则** 教师通过上节课的代码来分析 find_best_student 函数中所做的事情是否单一，由此引出单一职责原则。 课程思政：做事应专一。 **二、return 的用法和作用** 教师通过工头和刷墙工的例子，解释什么是调用方、什么是被调用方、什么是返回。通过生动的例子，有助于学生的理解与掌握。然后再讲解 return 的用法和作用。 **三、继续改变功能需求** 向学生说明需求的改变情况，可以问学生，应该如何修改函数的定义才能既实现功能又能符合单一职责原则？培养学生独立思考的能力。然后教师逐步引导并演示。 **四、代码复用** 通过水仙花数的案例，向学生展示代码复用的威力以及讲解什么是代码复用。并让学生自己动手完成 5.2.4 的思考题。 **五、归纳总结，布置随堂练习** （1）回顾上课前的学习目标，对本节课知识点进行总结。

教师带领学生总结本节课需要了解或掌握的知识点。

（2）布置随堂练习，检查学生掌握情况。

结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。

第三、四课时

一、创设情境，导入课程

正所谓人生三问无非就是早上吃啥，中午吃啥，晚上吃啥。相信你应该和我一样，每天都为这吃啥三连所困扰。一般来说，像这种连自己都不知道想要什么的问题，不如留给老天爷来替我们做主。此时想必你可能会这样做，打开微信，找到骰子表情，然后心中默念一些美食，然后发出骰子表情，骰子扔到啥就吃啥。但是呢，有时候我们的选择会有很多，比如有 13 种美食可供选择，而微信的骰子表情却只有 6 面，怎么办呢？那我们不如尝试设计一个叫“今天吃啥”的程序来解决今天吃啥的问题。通过该案例引出模块的概念。

二、使用 random 模块并实现程序的功能

模块的使用步骤总共两步：1.请君入瓮；2.尽情使唤。教师通过代码演示让学生掌握模块的基本套路，并逐步引导学生实现程序的功能。

三、举一反三

教师向学生展示其他常用模块，以及其用法。让学生感受到模块给我们带来的便捷。并且能够让学生深刻体会到代码复用的威力。

四、归纳总结，布置随堂练习

（1）回顾上课前的学习目标，对本节课知识点进行总结。

教师带领学生总结本节课需要了解或掌握的知识点。

（2）布置随堂练习，检查学生掌握情况。

结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。

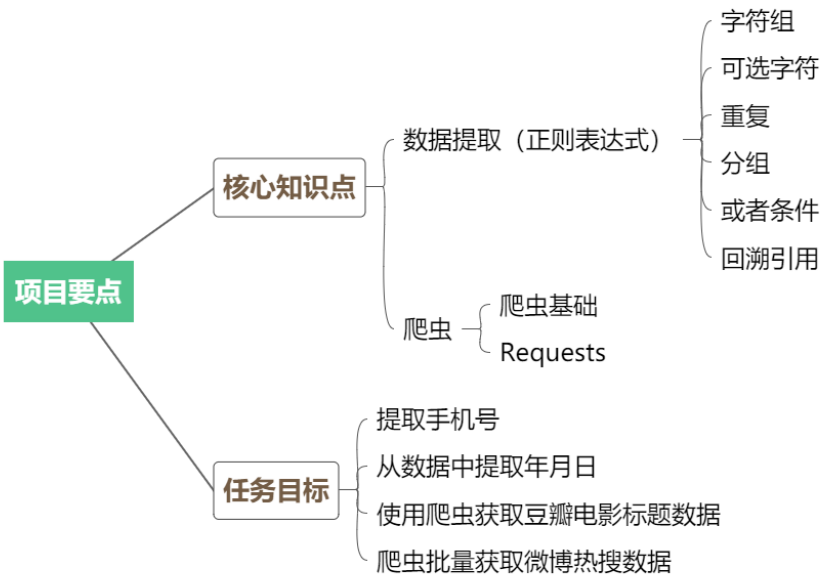
实训操作

实训操作主要针对项目中需要重点掌握的知识点，以及在程序中容易出错的内容进行练习，通过实训操作可以考察学生知识点和技能点的掌握情况，通过反复训练以达到项目目标。

	上机一： **形式：单独完成** 1. **题目：**（请编写一个程序，根据输入的学生信息，打印出身高最高的三位学生的姓名。测试样例如下。） 输入：{'张三':181, '王五':180, '赵六':179, '尼古拉斯赵四':147, '大力':165} 输出：张三 王五 赵六 输入：{'李四':161, '钱八':171, '赵六':159, '赵三':165, '大力':158} 输出：钱八 赵三 李四 **上机二：** **形式：单独完成** **题目：**（请编写一个交互式程序，向用户询问一个数字，然后该程序能够打印出这个数字的反转。测试样例如下） 输入：`1314` 输出：`4131` 输入：`2333` 输出：`3332`
作业	**练习题/操作题** 1. **定义一个函数：** ●编写一个函数 calculate_sum，接受两个参数 a 和 b，返回它们的和。 ●调用该函数并打印结果。 2. **使用模块完成计算：** ●使用 math 模块,编写一个程序计算一个数的平方根和绝对值。 ●示例输入：number = -16，输出平方根和绝对值。

教学后记	本章介绍了函数的定义和模块的使用。学生对函数的定义和调用掌握较好，但在理解函数的作用域和参数传递时仍存在一些问题。部分学生对模块的导入和使用不够熟练，尤其是在使用 math 模块时，对函数的调用语法不够熟悉。后续课程中需要通过更多实例帮助学生理解函数和模块的结合使用。

课题名称	项目六：重要信息的提取——正则表达式与爬虫	计划学时	6 学时
教学内容	课程介绍 提取手机号 从数据中提取年月日 使用爬虫获取豆瓣电影标题数据 爬虫批量获取微博热搜数据		
教学目标及基本要求	1. 数据提取（正则表达式） 2. 爬虫基础 3. Python 的 requests 库		
教学重点	1. 数据提取（正则表达式） 2. 爬虫基础 3. Python 的 requests 库		
教学难点			
教学方式	教学做一体化		
教学过程	**第一课时** **（项目介绍、使用正则表达式提取手机号码）** **一、项目场景，导入课程**		



随着移动互联网越来越普及，大数据时代来临，网络上的信息量越来越大，很多人获取数据的方式还是通过网站或者 App。但是有时候我们想要的数据并不能通过网站或者 App 下载得到，遇到这种场景我们可以使用**网络爬虫技术**，通过爬虫可以控制计算机不间断地从网络上获取我们想要的数据。

在爬虫获取到想要的数据之后，我们要从这些数据中提取关键词，比如从电影网站提取电影标题、评分，从招聘信息中提取薪资等等。

而要提取目标数据，可以使用正则表达式，**正则表达式**是提取数据的利器，它可以很方便地提取某些符合复杂规则的字符串。

本项目将带领你使用**爬虫**和**正则表达式**获取网页数据，提取关键内容，带你感受爬虫和正则表达式的威力。

二、使用正则表达式提取手机号码

第一个任务需要你了解什么是正则表达式，并且掌握正则表达式的用法。接下来我们以提取手机号为例来了解什么是正则表达式。

现在有一段字符串如下。

<table><tr>hello world 18111234589<tr><tr>name:张三,tel:18711001111</tr></table>

需要你从这段字符串中将手机号提取出来。

假设手机号码的规则如下。

（1） 必须是 11 位的数字；

（2） 第一位数字必须以 1 开头，第二位数字可以是[3,4,5,7,8]中的任意一个，后面 9 个数是[0-9]中的任意一个数字。

现在请你根据规则提取字符串中符合规则的手机号。

如果使用 if else 进行判断，需要多少个判断语句呢？

显然使用 if else 判断来提取特定规则的字符串太麻烦了，而使用**正则表达式**则可以非常方便地**提取某些符合特定规则的数据**。

三、search()函数

在 Python 中通过 re 模块即可使用正则表达式。re 模块具有很多函数，通过这些函数可以让开发者灵活地调用正则表达式实现不同的功能。

我们要学习的第一个函数是 search()函数，它的目的是**接收一个正则表达式和一个字符串，并返回第一个匹配的字符串**。

例如。

```
import re
# search()的第一个参数为正则表达式，第二个参数为要处理的字符串
result = re.search(r'fox','the quick brown fox jumpred')
print(result.span())  # span 方法获取的是正则表达式匹配到的位置
print(result)
```

运行结果如下。

(16, 19)

<re.Match object; span=(16, 19), match='fox'>

使用 re.search() 函数会返回一个 Match 对象，调用 Match 对象的 span()函数可以获取正则表达式匹配到字符的位置，直接输出 Match 对象则会得到 Match 对象的描述信息，如果没有匹配到任何数据则会返回 None。r'fox'为正则表达式，其中的 r 是告诉 python 解释器这是原始字符串，'fox'则表示正则表达式想要匹配的数据。

四：找到多个匹配的数据

re.search() 的一个限制是它仅仅返回**第一个匹配的数据**，但是，有时候我们期望获取**多个匹配的数据**。

五：字符组

字符组（[]）允许匹配一组可能出现的字符。

六：区间

有一些常见的字符组非常大，比如，**匹配所有的数字**，如果使用 6.1.3 中的方法，匹配所有的数字的正则为：[0123456789]。

七：取反

到目前为止，我们定义的字符组都是根据可能出现的字符来定义的，不过有时候我们可能希望**根据不会出现的字符来定义字符组**，这个叫作**取反**操作。

正则表达式

一、快捷方式

快捷方式	描述
\w	与任意单词匹配
\d	与任意数字匹配

\s	匹配空白字符
\b	匹配一个长度为0的子串

二、任意字符

字符代表匹配任何单个字符的快捷方式，它只能出现在方括号以外（出现在方括号内表示匹配 . 字符本身）。

注意：.字符只有一个不能匹配的字符，也就是换行符（\n）。

例如。

```
import re
result = re.findall(r'a..','all ak47 abc and a')
print(result)
```

运行结果如下。

['all', 'ak4', 'abc', 'and']

三、可选字符

有时候，我们可能想要匹配一个单词的不同写法，比如“color”和“colour”，或者“honor”与“honour”。

这个时候我们可以使用?号指定一个字符或字符组是可选的，这意味着该字符出现**零次或一次**。

例如。

```
import re
result = re.findall(r'欧?阳峰','欧阳峰 阳峰')
print(result)
```

运行结果如下。

['欧阳峰', '阳峰']

使用 “欧?阳锋”可以指定“欧”字可以出现零次或一次。通过这个示例你能否使用正则表达式匹配“color”和“colour”，

或者“honor”与“honour”呢？

四、重复

到目前为止，我们只是学习了关于仅出现一次的字符串匹配，在实际开发过程中，这样肯定不能满足需求，比如要匹配电话号码、匹配身份证号，这些都是很多个数字组成的。

如果遇到这样的情况，我们可能**期望一个字符组连续匹配好几次**。

在一个字符组后加上{N}，就可以表示{N}之前的字符组出现 N 次。

五、重复区间

有时候，我们不知道重复的次数具体是多少，比如身份证有 15 位也有 18 位。

这个时候可以使用**重复区间**，语法：{M,N}，M 是下界而 N 是上界。

例如。

```
import re
res = re.findall(r'\d{3,4}',' 020 0733')
print(res)
```

运行结果如下。

['020', '0733']

通过上述代码，可以发现\d{3,4}既可以匹配 3 个数字也可以匹配 4 个数字，不过当有 4 个数字的时候，优先匹配的是 4 个数字，这是因为正则表达式默认是**贪婪模式**，即尽可能地匹配更多字符，而要使用**非贪婪模式**，则需要在表达式后面加上 ?号。

六、开闭区间

有时候我们可能遇到字符组的重复次数没有边界，比如从 1 个到无穷多个，这个时候可以使用开闭区间，语法：{N,} 。

例如。

```
import re
result = re.findall(r'\d{1,}','1 20 020 0733')
print(result)
```

运行结果如下。

['1', '20', '020', '0733']

\d{1,}表示匹配 1 个到无穷多个数字。

七、速写

在正则表达式中有两个使用频率非常高的字符：* 和 +。

- + 号代表匹配 1 个到无穷个，等价于{1,}；
- * 号代表匹配 0 个到无穷个，等价于{0,}。

例如。

```
import re
result1 = re.findall(r'\d+','1 20 020 0733')
result2 = re.findall(r'编号\d*','编号 编号 89757')
print(result1)
print(result2)
```

运行结果如下。

['1', '20', '020', '0733']

['编号', '编号 89757']

第二课时

（实训操作）

完成前面课时的正则编写，独立完成使用正则提取手机号的程序

选择练习题完成练习

1. 请你使用字符组匹配 Ruby、Rube、ruby、rube。

2. 匹配所有的**数字**、**小写字母**和**大写字母**。需要匹配的数据如下：

abcdefg
012345678
987654321
ABCDEFG

3. 匹配 a-z 的小写字母和 A-F 的大写字母。

4. 匹配所有王姓同学的信息，数据如下。

王敏 0001
王磊 1234
王静 0102
王丽 0502
王秀英 0503
王芳芳 0503
张三 0731

第三课时

（从文本数据中提取年月日数据）

一、分组

在正则表达式中还提供了一种**将表达式分组**的机制，使用分组时，除了获得整个匹配，还能够在匹配中选择每一个分组。

要实现分组很简单，使用()即可。

例如。

```
import re
result = re.search(r'([\d]{4})-([\d]{7})','张三：
0731-8825951')
print(result.group())
print(result.group(1))
print(result.group(2))
```

运行结果如下。

二、或者条件

使用分组的同时还可以使用或者（or）条件。

要使用或者条件可以在各个条件之间加上一个 | 号：

例如。

```
import re
result= re.findall(r'(张三|李四)','张三、李四、王五、赵六
')
print(result)
```

运行结果如下。

['张三', '李四']

三、分组的回溯引用

正则表达式还提供了一种引用之前匹配分组的机制。

例如。

```
import re

result =
re.findall(r'<[\w_\-]+>.*?</[\w_\-]+>','0123<font>提示
</font>abcd')

print(result)
```

运行结果如下。

['提示']

上述代码确实可以匹配，不过可能还有另一种情况，如果解析数据为：0123<font>提示</bar>abcd，则结果可能并不符合我们的预期。

```
import re

res =
re.findall(r'<[\w_\-]+>.*?</[\w_\-]+>','0123<font>提示
</bar>abcd')

print(res)
```

运行结果如下。

<font>提示</bar>

font 和 bar 明显不是一对正确的标签，但是正则表达式还是将它们给匹配了，所以这个结果是错误的。

我们想让后面分组的正则也匹配 font，但是现在所有形式的都会匹配。

那如果想让后面分组的正则和第一个分组的正则匹配同样的数据该如何做呢？

可以使用**分组的回溯引用**，使用“\N”即可回溯引用编号为“N”的分组，因此修改代码如下。

```
import re
result1= re.findall(r'<([\w_-]+)>(.*?)</\1>','<font>提示</bar>')
result2 = re.findall(r'<([\w_-]+)>(.*?)</\1>','<font>提示</font>')
print(result1)
print(result2)
```

运行结果如下。

[]

[('font', '提示')]

使用了分组的回溯引用，我们可以使用\1 来代表第一个分组匹配到的结果。

四、提取年月日数据

接下来我们利用正则表达式中的分组对年月日的信息进行提取，编写代码如下。

```
import re
res =
re.findall(r'(\d{4})[-/](\d{1,2})[-/](\d{1,2})','2020-1-2、2020-2-2、2020-01-02、2020/01/20')
print(res)
```

运行结果如下。

[('2020', '1', '2'), ('2020', '2', '2'), ('2020', '01', '02'), ('2020', '01', '20')]

第四课时

（使用爬虫获取豆瓣电影标题）

一、使用爬虫获取豆瓣电影标题

了解了正则表达式，接下来可以利用正则表达式和 Python 爬虫从网络上抓取数据。

在这个小节中，我们将使用爬虫与正则表达式获取“豆瓣网”电影标题的数据。

二、理解网页结构

要使用爬虫获取网站的数据，第一步需要先理解网页的结构。

我们所看到的所有网站都是由结构化的代码构成的，经过浏览器的处理，呈现出各式各样的页面。

一个网页通常由三部分代码组成：HTML 代码、CSS 代码和 Javascript 代码。

例如。

```
<html>
    <div class="content">
    <h2><b>电影网站详情页</b></h2>
    <p>电影名称</p>
    <p>大圣归来</p>
    <a href = "https://www.codejiaonang.com/">编程胶囊网页链接</a>
    <img data-v-15b951ee=""
src="https://cdn.codejiaonang.com/img/logo@2x.198d744b.png">
    </div>
</html>
```

这是一段 HTML 代码，网络爬虫想要做的，就是从这些结构化的代码中提取出需要的信息，例如提取“大圣归来”这四个字。

三、查看网页源代码

要使用爬虫获取网页中的关键数据，我们首先需要先获取网页的源代码，这样才能从中获取到关键信息。

基本上浏览器都提供了查看网页源代码的功能。

查看网页的源代码有两种方式。

（1） 在网页中单击右键，选择“查看网页源代码”，如图 6-1 所示。

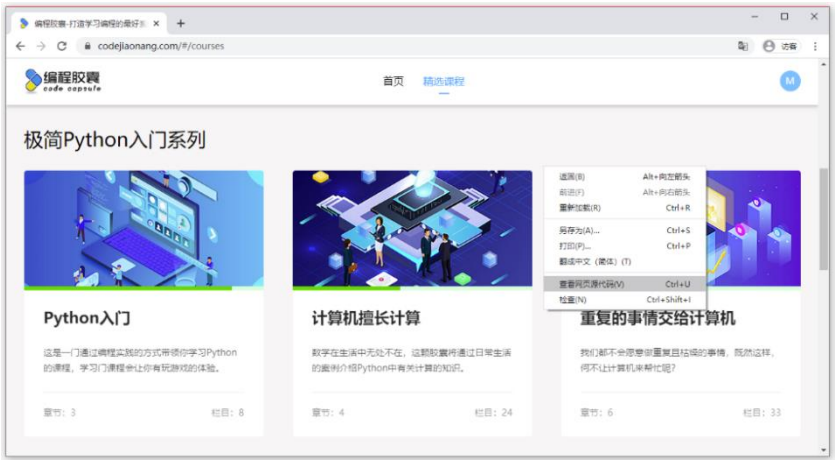


图 6-1 查看网页源代码

（2） 鼠标移动到在当前页面的任意内容上，单击右键选择 “检查”按钮，在新弹出的窗口中就能够看到这段内容对应的代码，如图 6-2 所示。

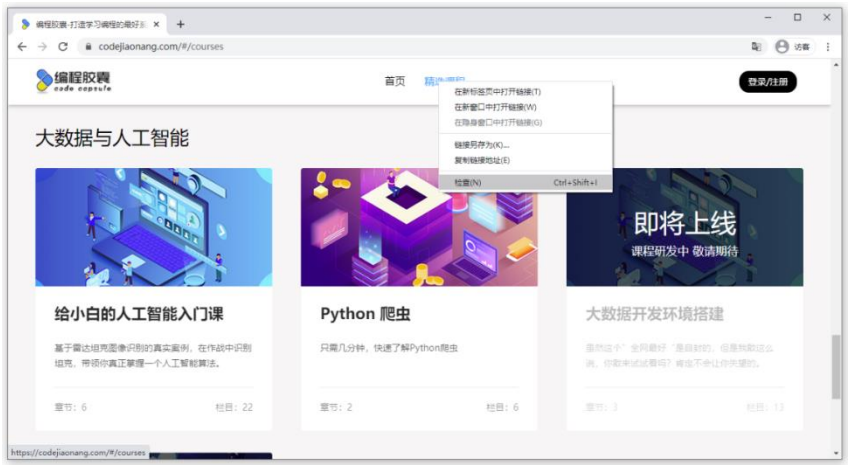


图 6-2 检查元素

四、获取网页数据

要提取网页中的关键信息，首先需要将这个网页数据下载下来。

使用 Requests 库可以很方便地下载网页代码。要使用

Requests 需要先安装它。安装 Requests，只要在你的命令行中运行下列命令即可。

```
pip install requests
```

注意：使用快捷键 windows 键+R，然后输入 cmd 之后按回车键即可进入命令行。

接下来使用 Requests 发送网络请求。在你的电脑任意位置，新建一个 crawler.py 文件。举个例子，如果要下载百度首页的网页代码，输入并执行以下 3 行代码。

```
import requests
res = requests.get('https://www.baidu.com/')
print(res.content.decode())
```

五、提取关键数据

获取到了网页的源代码之后我们想要从中提取到关键数据，例如标题数据。

要提取标题信息，在标题上单击鼠标右键，查看标题对应的 HTML 代码，如图 6-3 所示。

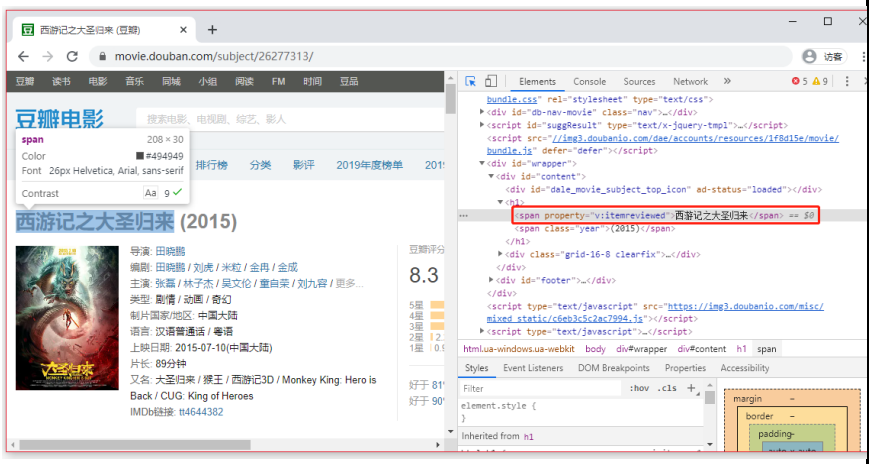


图 6-3 查看标题源代码

通过这个步骤我们可以发现标题在网页中的源代码为。

```
<span property="v:itemreviewed">西游记之大圣归来</span>
```

知道了标题的源代码，接下来可以使用正则表达式从网页源代码中提取标题数据。

```
headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/54.0.2840.99 Safari/537.36"}
```

```
import requests
url = 'https://movie.douban.com/subject/26277313/'
res = requests.get(url,headers = headers)
page = res.content.decode()
import re
title = re.search(r'<span
property="v:itemreviewed">(.*?)</span>',page,re.S)
print(title.group(1))
```

运行结果：西游记之大圣归来

可以发现使用(.*?)就能将我们想要的数据提取出来。

第五、六课时 使用爬虫批量获取微博热搜数据

一、查看网页的结构

首先尝试完成如下操作。

（1） 打开浏览器访问微博热搜；

（2） 单击鼠标右键[检查]，查看网页源代码；

（3） 查看某一条热搜的源代码，如图 6-4 所示。

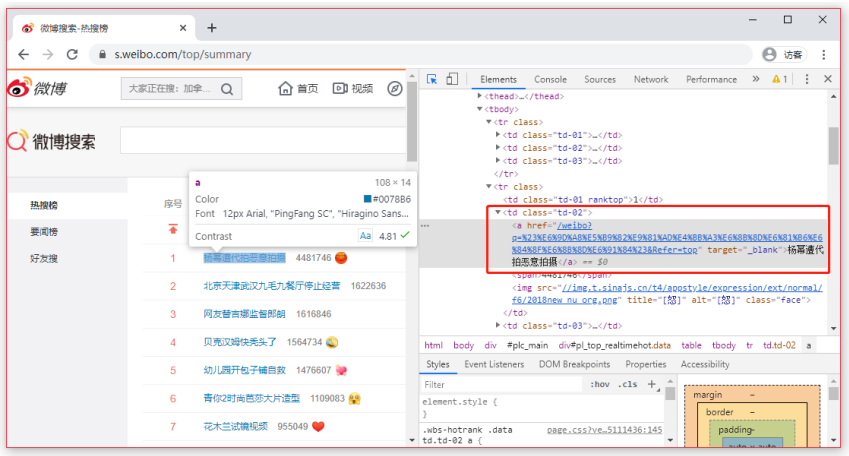


图 6-4 查看热搜标题的源代码

接下来我们需要使用爬虫下载网页数据。

```
import requests
cookies = {'SUB':'working'} # 设置 cookies
html =
requests.get('https://s.weibo.com/top/summary',cookies
= cookies)
```

```
content = html.content.decode()
print(content)
```

二、提取网页的关键信息

获取到网页数据之后，我们需要从网页数据中提取关键数据。

可以尝试复制其中一条热搜的源代码，在 td 标签上单击右键“Copy”-> “Copy element”即可，如图 6-5 所示。

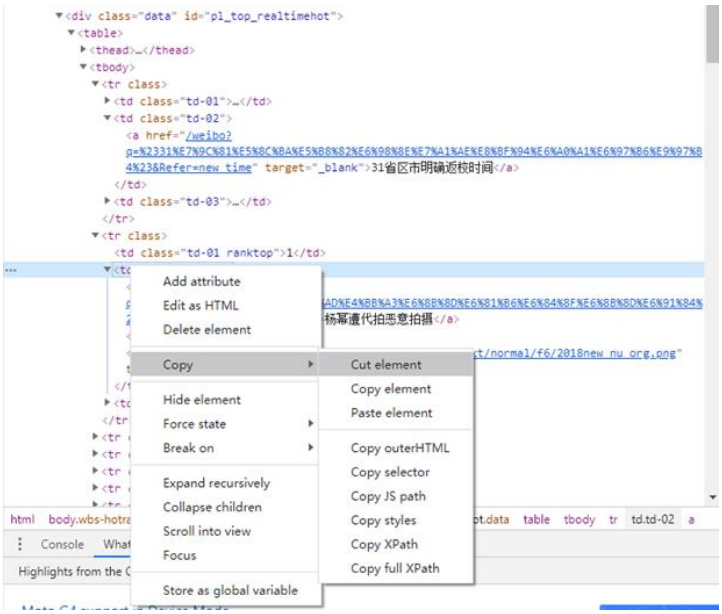


图 6-5 复制一条热搜数据

然后将这条热搜源代码粘贴到正则测试网站（正则测试网站可以实时测试正则表达式能匹配到的数据，通过搜索引擎搜索“正则测试”即可获取正则测试网站的网址）中，编写正则表达式“<td class="td-02">\s*(.*?)”，如图 6-6 所示。

图 6-6 编写正则表达式

可以发现关键信息已经被正则表达式匹配了，接下来可以编写代码进行批量获取。

三、批量获取网页数据

接下来我们只需要使用 requests 和正则表达式就可以实现微博热搜数据的获取，编写代码如下。

```
html = requests.get('https://s.weibo.com/top/summary')
content = html.content.decode()
import re
tbody = re.findall(r'<tbody>(.*?)</tbody>',content,re.S)
td_list = re.findall(r'<td class="td-02">\s*<a
href="(.*?)".*?>(.*?)</a>',content,re.S)
print(td_list)
```

运行结果如下。

```
[('/weibo?q=%232020%E5%B9%B4%E5%85%A8%E5%9B%BD%E8%AE%
A1%E5%88%92%E6%8B%9B%E8%81%98%E7%89%B9%E5%B2%97%E6%95
%99%E5%B8%8810.5%E4%B8%87%E5%90%8D%23&Refer=new_time'
,
'2020 年全国计划招聘特岗教师 10.5 万名'),
... 以下省略 ...
```

这段爬虫程序完成了批量获取微博热搜的链接与链接对应的标题。

四、数据加工

通过查看获取的数据可以发现，热搜的链接网址无法直接打开，还需要将微博的域名拼接上去。

```
# 处理数据
links = ["https://s.weibo.com/" + str(link[0]) for link
in td_list]
titles = [title[1] for title in td_list]
print(links)
print(titles)
```

运行结果如下。

```
['https://s.weibo.com//weibo?q=%23%E5%85%AC%E5%AE%89%E9
%83%A8%E6%89%93%E6%8B%90%E5%8A%9E%E9%A6%96%E6%AC%A1%E8%
AE%A4%E4%BA%B2%E7%9B%B4%E6%92%AD%23&Refer=new_time',
... 以下省略 ...
['公安部打拐办首次认亲直播', '中餐厅第四季国内录制', '天门山
```

翼装飞行失联女大学生已找到', '王一博 自己还是自己', '李慧琼当选香港立法会内委会主席', '郑爽坐路边自拍', '特朗普儿子称新冠病毒 11 月会消失',
... 以下省略 ...

再次测试可以发现热搜标题对应的链接已经能正常访问了。

五、数据持久化

如果想要重复利用提取到的数据，还需要对它进行数据持久化，也就是将它保存在**本地**或者**数据库**中。

接下来我们将获取到的数据保存至 csv 文件中。

```
import csv
# 写标题与内容
filePath = 'D://top_weibo.csv'  # 文件保存路径
with open(filePath, 'w') as csvfile:
    head = ['链接','标题']  # csv 文件标题
    writer = csv.writer(csvfile)
    writer.writerow(head) #写首行数据
    for line in range(len(links)):
        row = []   # 数据行
        row.append(links[line])
        row.append(titles[line])
        writer.writerow(row)  # 写一行数据
```

运行代码之后可以发现 D 盘根目录下生成了一个.csv 文件，打开它可以看到数据。

项目总结与课后练习

在本项目中，我们使用 Python 编写了四个程序。

1. 提取手机号；
2. 从文本数据中提取年月日；
3. 使用爬虫获取豆瓣电影标题;

4. 批量获取微博热搜数据。

通过编写这些程序我们学到了如下知识。

1. 爬虫程序本质上就是下载网络中的数据，然后提取关键信息；
2. 有一些网站不允许爬虫，可以通过 headers 参数伪装成普通用户的方式来获取网站数据；
3. 正则表达式可以帮助我们提取字符串中的关键数据；
4. 正则表达式的基本使用方法可以总结如表 6-2 所示。

表 6-2 正则表达式总结

实例	描述
Pp]ython	匹配“Python”或“python”。
ub[ye]	匹配“ruby”或“rube”。
abcdef]	匹配中括号内的任意一个字母。
0-9]	匹配任何数字。类似于 [0123456789]。
a-z]	匹配任何小写字母。
A-Z]	匹配任何大写字母。
a-zA-Z0-9	匹配任何字母及数字。
^au]	除了 au 字母以外的所有字符。
^0-9]	匹配除了数字外的字符。

.	匹配除换行符（\n）之外的任何单个字符。
?	匹配一个字符零次或一次，另一个作用是非贪婪模式
+	匹配 1 次或多次
*	匹配 0 次或多次
\b	匹配一个长度为 0 的子串
\d	匹配一个数字字符。等价于 [0-9]。
\D	匹配一个非数字字符。等价于 [^0-9]。
\s	匹配任何空白字符，包括空格、制表符、换页符等等。等价于 [\f\n\r\t\v]。
\S	匹配任何非空白字符。等价于 [^ \f\n\r\t\v]。
\w	匹配包括下划线的任何单词字符。等价于‘[A-Za-z0-9_]’。
\W	匹配任何非单词字符。等价于 ‘[^A-Za-z0-9_]‘。
\b	匹配一个长度为 0 的子串

项目习题

1. 编写正则表达式提取所有的电话号码，数据如下。

   2118673676

   (211)8673676

   211.867.3676

   (211)867-3676

	211-867-3676 2. 获取 36 氪新闻网站首页的所有新闻。 目标数据如下（因为数据每天都会更新，以下数据仅作为格式参考）。 业绩快报 | 百度 Q1 净利润跌 80%，搜索公司总裁向海龙辞职 https://36kr.com/p/5205529 36 氪首发丨「100 课堂」获数千万 A 轮融资，“公立校+下沉”市场前景广阔 https://36kr.com/p/5205388
作业	**练习题/操作题** 1. **提取手机号码：** ● 编写一个程序，使用正则表达式从以下字符串中提取手机号码： 我的手机号是 13812345678，欢迎联系！ ● 输出提取的手机号码。 2. **爬取网页标题：** ● 使用 requests 和 BeautifulSoup，编写一个程序爬取百度首页的标题。 ● 输出爬取的标题内容。
教学后记	本章介绍了正则表达式和爬虫基础，学生对正则表达式的使用表现出较高的兴趣，但在复杂的模式匹配时仍存在一些困难。部分学生对爬虫的网页结构分析不够熟练，导致在提取数据时出现错误。后续课程中需要通过更多实例帮助学生理解正则表达式的高级用法和爬虫的实现逻辑。

<table>
<tr><td>课题名称</td><td>项目七：让烦琐的工作自动化——使用 Python 处理 Excel 文件</td><td>计划学时</td><td>4 学时</td></tr>
<tr><td>教学内容</td><td colspan="3">openpyxl 的安装
使用 openpyxl 创建 excel 文件
使用 openpyxl 创建、修改 sheet
使用 openpyxl 读写单元格数据
使用 openpyxl 保存 excel 文件</td></tr>
<tr><td>教学目标及基本要求</td><td colspan="3">1. 掌握 openpyxl 的安装命令式
2. 掌握 openpyxl 操作 Excel 文件的基本套路
3. 掌握 openpyxl 修改 Excel 文件数据的常用函数</td></tr>
<tr><td>教学重点</td><td colspan="3">1. openpyxl 操作 Excel 文件的基本套路 return 的用法
2. openpyxl 修改 Excel 文件数据的常用函数</td></tr>
<tr><td>教学难点</td><td colspan="3">openpyxl 修改 Excel 文件数据的常用函数</td></tr>
<tr><td>教学方式</td><td colspan="3">教学做一体化</td></tr>
<tr><td>教学过程</td><td colspan="3">第一课时
（openpyxl 的安装与基本使用方法）
一、创设情境，导入课程
在日常生活、学习、工作当中经常会接触到一款专门处理表格数据的软件——Excel。虽然该软件功能十分强大，但使用起来也十分繁琐。有的时候，为了整理一些表格，需要耗费我们大量的时间和精力。引出 Python 给我们提供了一个专门用来处理 excel 文件数据的模块：openpyxl。教师演示 openpyxl 的安装方法和验证是否安装成功的方法。
二、介绍 openpyxl 处理 excel 文件的基本套路
根据我们自己操作 excel 的流程引出 openpyxl 处理 excel 文件的基本套路，如：打开文件，选择具体的 sheet，修改单元格数据等。
三、归纳总结，布置随堂练习
（1）回顾上课前的学习目标，对本节课知识点进行总结。
教师带领学生总结本节课需要了解或掌握的知识点。</td></tr>
</table>

（2）布置随堂练习，检查学生掌握情况。

结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。

第二课时

（自动修改空调售价）

一、批量修改价格

教师讲解案例的功能要求，先让学生自己动手实现功能。学生动手期间巡视学生的完成情况，发现学生在实现过程中的共性问题，并予以讲解。

二、归纳总结，布置随堂练习

（1）回顾上课前的学习目标，对本节课知识点进行总结。

教师带领学生总结本节课需要了解或掌握的知识点。

（2）布置随堂练习，检查学生掌握情况。

结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。

第三、四课时

（多表合一）

一、创设情境，导入课程

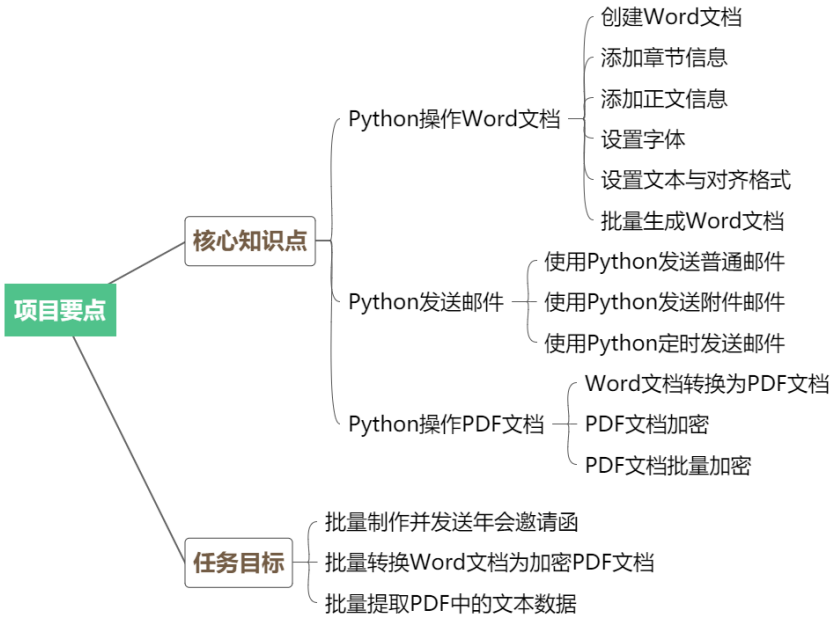
小明在某电商公司从事运营职位，有一天，老板给他安排了一个活：需要根据用户的评论来分析出哪些品牌，哪些型号的热水器的口碑好，为公司的采购部门提出有效建议。由于刚入职不久，就被安排这么有含金量的工作，小刚甚是开心，很爽快的答应一定会做好。

可当拿到数据之后，傻眼了。数据放在了 data 文件夹下，数据都是以 Excel 表格的形式存储的，而且 Excel 表格有很多，总共有 21 个。而每个 Excel 表格中将近有 10000 条左右的数据。

小明认为要将所有 Excel 表格的数据汇聚到一个 Excel 表格中，才好进行后续的分析。请问，你能帮助小刚将这些 Excel 表格的数据合并到一张表格中吗？

	课程思政：学以致用 **二、多表合一** 引导学生思考如何实现功能，并在引导过程中自主查阅教材，并让学生自己动手编程实现功能。 **三、归纳总结，布置随堂练习** （1）回顾上课前的学习目标，对本节课知识点进行总结。 教师带领学生总结本节课需要了解或掌握的知识点。 （2）布置随堂练习，检查学生掌握情况。 结合随堂练习资源，给学生布置随堂练习，检测学生的掌握程度，并及时解决学生出现的问题。
作业	**练习题/操作题** 1. **读取 Excel 文件：** o 使用 openpyxl 模块，读取一个 Excel 文件（如 data.xlsx），并打印第一列的所有数据。 2. **修改 Excel 文件：** o 使用 openpyxl 模块，打开一个 Excel 文件，将第一行的单元格内容全部设置为大写，然后保存文件。
教学后记	本章介绍了 Python 处理 Excel 文件的方法，学生对 openpyxl 模块的使用表现出较高的兴趣，但在处理复杂数据时仍存在一些困难。部分学生对 Excel 文件的结构理解不够深入，导致在读写操作时出现错误。后续课程中需要通过更多实例帮助学生巩固 Excel 文件的操作。

课题名称	项目八：文档的批量处理 Python 处理 Word 与 PDF	计划学时	4 学时
教学内容	课程介绍 批量制作并发送年会邀请函 将 Word 文件批量转换为加密 PDF 文件 批量提取 PDF 文件中的文本数据		
教学目标及基本要求	1. Python 操作 Word 文档 2. Python 发送邮件 3. Python 操作 PDF 文档		
教学重点	1. Python 操作 Word 文档 2. Python 发送邮件 3. Python 操作 PDF 文档 4. 文档的批量操作		
教学难点	批量操作文档		
教学方式	教学做一体化		
教学过程	**第一课时** **（项目介绍、批量制作并发送年会邀请函）** **一、项目场景，导入课程**		



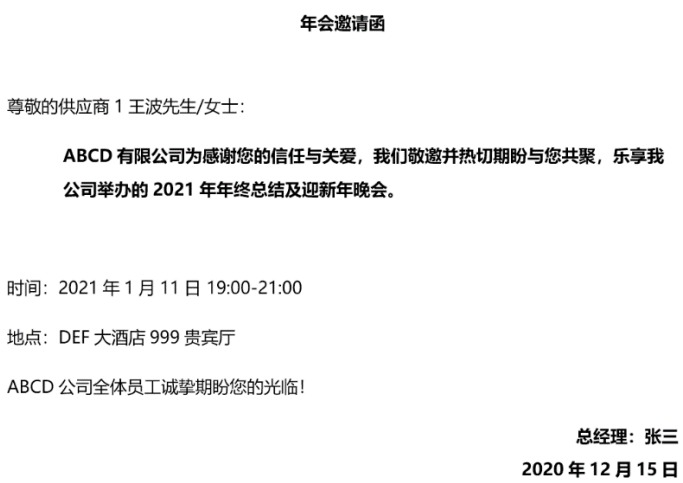
假设你现在收到一个非常枯燥的任务，使用 Word 文档编写 100 封年会邀请函，然后将 Word 格式的邀请函转换为 PDF 格式，最后将这些邀请函通过 Email 分别发送给对应的客户，邀请他们来参加公司年会。

可以发现完成这个任务需要做大量的枯燥的重复劳动，Python 有没有办法帮助我们批量地处理这些重复的工作呢？

答案是肯定的，使用 Python 可以批量地处理 Word 和 PDF 文档，在本项目中，我们将使用 Python 针对 Word 和 PDF 文档进行批量处理，使用 Python 让繁琐的工作自动化。

二、批量制作并发送年会邀请函

你的第一个任务是：编写很多份 Word 格式的年会邀请函，然后将这些文档通过电子邮件分别发送给公司的客户（客户信息保存在 Excel 文件中），文档的内容如图 8-1 所示。



年会邀请函

尊敬的供应商 1 王波先生/女士:

ABCD 有限公司为感谢您的信任与关爱，我们敬邀并热切期盼与您共聚，乐享我公司举办的 2021 年年终总结及迎新年晚会。

时间：2021 年 1 月 11 日 19:00-21:00

地点：DEF 大酒店 999 贵宾厅

ABCD 公司全体员工诚挚期盼您的光临!

总经理：张三

2020 年 12 月 15 日

图 8-1 年会邀请函

三、创建 word 文档

利用 python-docx 模块， Python 可以创建和修改 Word 文档。

在命令行输入 pip install python-docx 命令即可安装该模块。

四：添加标题

通过 add_heading('标题',level) 就可以给 Word 文档添加标题。

例如：

```
import docx
doc = docx.Document()
doc.add_heading('标题 0', 0)
doc.add_heading('标题 1', 1)
doc.add_heading('标题 2', 2)
doc.add_heading('标题 3', 3)
doc.add_heading('标题 4', 4)
doc.save('D://邀请函_addHead.docx')   # 保存更改
```

运行这段程序，然后打开 D 盘根目录下的“邀请函_addHead.doxc”文件，

可以看到该文件中已经有标题数据了

五：居中

除了添加普通的文本，我们还希望添加一些带样式的文本，例如居中的文本，代码如下。

```
import docx
from docx.enum.text import WD_ALIGN_PARAGRAPH   # 导入 docx 枚举

doc = docx.Document()
paragraph = doc.add_paragraph('年会邀请函')
# 添加居中属性
paragraph_format = paragraph.paragraph_format
paragraph_format.alignment = WD_ALIGN_PARAGRAPH.CENTER

doc.add_paragraph('尊敬的 **** 先生/女士：')     # 普通段落
doc.save('D://邀请函_addCenter.docx')            # 保存更改
```

六：缩进

缩进是段落和其容器边缘之间的水平空间，通常是页边距，要设置缩进需要导入 Inches 对象，导入语句为： from docx.shared import Inches，代码如下：

七：加粗

使用 run 对象可以对字体进行加粗，例如。

```
import docx
from docx.shared import Inches
doc = docx.Document()
doc.add_paragraph('尊敬的 **** 先生/女士：')
# 缩进
paragraph = doc.add_paragraph() # 在这个段落不添加内容
paragraph_format = paragraph.paragraph_format
paragraph_format.left_indent = Inches(0.5)
# 加粗
run = paragraph.add_run('邀请您参加我们公司的年会')
run.bold = True
doc.save('D://邀请函_addBold.docx')   # 保存更改
```

。

八、设置中文字体

可以发现使用 Python 创建的 Word 文档默认使用的是西文字体，而不是中文字体，生成的 Word 文档字体格式不符合我们邀请函的需求，接下来将它改为“宋体”

第二课时

（读取客户数据，写入 Word 文档）

一、读取 Excel 中的客户数据

二、批量生成邀请函

客户信息读取完成之后，接下来只需要三个步骤就可以批量生成邀请函。

（1） 在 D 盘下新建一个名为“邀请函”的文件夹。

（2） 从 Excel 文件中获取受邀请人单位的名字；

（3） 通过客户数据批量生成邀请函。

```
import docx
from docx.enum.text import WD_ALIGN_PARAGRAPH  # 导入 docx 枚举
from docx.shared import  Cm
from docx.oxml.ns import qn

# 为方便测试，在这里手动编写供应商的数据
names=['供应商 1 王波 6319@xxx.com', '供应商 2 刘海洋 123@xxx.com', '供应商 3 少和光 1234@xxx.com', '供应商 4 真凡巧 12321@xxx.com']

# 批量生成邀请函
for name in names:
    info_list = name.split()  # 分割客户的数据 分割成 单位、姓名、邮箱
    doc = docx.Document() # 创建 Word 文档
    doc.styles['Normal'].font.name = u'微软雅黑'

doc.styles['Normal']._element.rPr.rFonts.set(qn('w:eastAsia'), u'微软雅黑')
    paragraph = doc.add_paragraph()
    run_title = paragraph.add_run("年会邀请函\n")
    run_title.bold = True
    paragraph_format = paragraph.paragraph_format
    paragraph_format.alignment = WD_ALIGN_PARAGRAPH.CENTER
    doc.add_paragraph('尊敬的' + info_list[0] + info_list[1] + '先生/女士：')
    paragraph2 = doc.add_paragraph()
```

```
    paragraph_format2 = paragraph2.paragraph_format
    paragraph_format2.first_line_indent = Cm(1) #  设置首行缩进
    run = paragraph2.add_run('ABCD 有限公司为感谢您的信任与关爱，我们敬邀并热切期盼与您共聚，乐享我公司举办的 2020 年年终总结及迎新年晚会。')
    run.bold = True
    doc.add_paragraph() # 添加空行
    doc.add_paragraph("时间：2021 年 1 月 11 日 19:00-21:00")
    doc.add_paragraph("地点：DEF 大酒店 999 贵宾厅")
    doc.add_paragraph("ABCD 公司全体员工诚挚期盼您的光临！")
    paragraph_right = doc.add_paragraph()
    paragraph_format_left = paragraph_right.paragraph_format
    paragraph_format_left.alignment = WD_ALIGN_PARAGRAPH.RIGHT
    run_right = paragraph_right.add_run("总经理：张三\n2020 年 12 月 15 日")
    run_right.bold = True
    # 保存文件
    doc.save('D://邀请函/邀请函_' + info_list[0] + '_' + info_list[1] + '.docx')
```

三、使用 Python 发送邮件

在最后我们使用 Python 和 QQ 邮箱将所有的邀请函发送给客户，在发送邮件之前我们需要先设置好 QQ 邮箱，让它可以支持使用编程语言发送邮件。

首先登陆 QQ 邮箱，然后选择“设置”->“账户”。

三、获取邮箱授权码

通过编程的方式发送邮件，需要获取 QQ 邮箱的授权码（其他邮箱也类似），滑动到页面的中间部分可以看到“POP3/IMAP/SMTP...”等服务的设置，在这里需要开启“POP3”和“SMTP”的服务，然后单击“生成授权码”，如图 8-12 所示**四、重复**

到目前为止，我们只是学习了关于仅出现一次的字符串匹配，在实际开发过程中，这样肯定不能满足需求，比如要匹配电话号码、匹配身份证号，这些都是很多个数字组成的。

如果遇到这样的情况，我们可能**期望一个字符组连续匹配好几次**。

在一个字符组后加上{N}，就可以表示{N}之前的字符组出现 N 次。

四、发送普通邮件

有了授权码之后就可以发送邮件了，首先来发送一条普通邮件，代码如下。

```
import smtplib
from email.mime.text import MIMEText
mailserver = 'smtp.qq.com'                  #邮箱服务器地址
sender = '你的邮箱'
password = '填写你获取到的邮箱授权码'     #使用授权码
receiver = '填写收件人的邮箱'               #多个邮箱用逗号分隔
mail = MIMEText('这是发送的邮件内容！')
mail['Subject'] = '邮件的主题'
mail['From'] = sender   #发件人
mail['To'] = receiver   #收件人
smtp = smtplib.SMTP(mailserver,port=25) # 连接邮箱服务器
smtp.login(sender,password)                 # 登录邮箱
#使用 sendmail 发送邮件，三个参数分别是发送者，接收者，邮件内容
smtp.sendmail(sender,receiver,mail.as_string())
smtp.quit() # 发送完毕后退出 smtp
print('发送成功')
```

五、定时发送邮件

如果现在公司的领导让你在明天早上 6 点 30 分将年会邀请函通过邮件发送给公司所有的客户，我想你肯定不愿意早上给自己定一个闹钟，然后起床发邮件。

现在我们已经知道如何使用 Python 发送有附件的邮件，公司领导的这个要求就可以通过定时执行 Python 任务来解决了。

第三课时

（将 Word 批量转换为加密 PDF）

一、将 Word 文档转换为 PDF 文件

要将 Word 文档转换为 PDF 文件可以使用 python win32 库。

在命令行使用 pip install pywin32 即可安装 pywin32。

安装好之后在 D 盘的根目录下创建一个名为 python_practice 的文件夹，在该文件夹下创建一个名为 test.docx 的 Word 文档，并在 Word 文档中添加内容，也可以将任务 8.1 中的生成 Word 文档改名为 test.docx，如图 8-26 所示。

年会邀请函

尊敬的供应商 1 王波先生/女士:

ABCD 有限公司为感谢您的信任与关爱，我们敬邀并热切期盼与您共聚，乐享我公司举办的 2021 年年终总结及迎新年晚会。

时间：2021 年 1 月 11 日 19:00-21:00

地点：DEF 大酒店 999 贵宾厅

ABCD 公司全体员工诚挚期盼您的光临！

总经理：张三

2020 年 12 月 15 日

图 8-26 年会邀请函

接下来编写如下代码即可将 Word 文档转换为 PDF 文件。

```
from win32com.client import gencache
from win32com.client import constants, gencache

def createPdf(wordPath, pdfPath):
    """
    Word 文件转 pdf 文件
    :param wordPath: Word 文件路径
    :param pdfPath:  生成 Pdf 文件路径
    """
    word = gencache.EnsureDispatch('Word.Application')
    doc = word.Documents.Open(wordPath, ReadOnly=1)
```

```
        doc.ExportAsFixedFormat(pdfPath,
                                constants.wdExportFormatPDF,
                                Item=constants.wdExportDocumentWithMarku
p,
                                CreateBookmarks=constants.\
                                wdExportCreateHeadingBookmarks)
    word.Quit(constants.wdDoNotSaveChanges)

#在这里使用绝对路径，上级目录与下级目录之间使用两个反斜线（\\）
doc_name = 'D:\\python_practice\\test.docx'
pdf_name = 'D:\\python_practice\\test.pdf'
createPdf(doc_name,pdf_name)
```

运行代码可以看到 D 盘根目录下的“python_practice”文件夹中出现了一个“test.pdf”文件，如图 8-27 所示。

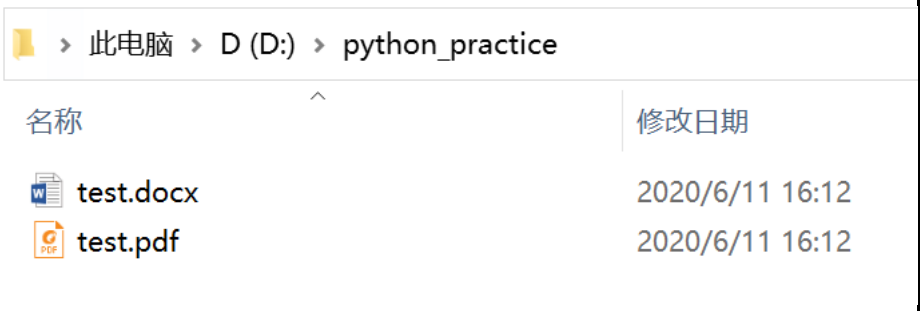


图 8-27 将 Word 转换为 PDF

打开“test.pdf”文件可以看到其中的内容和 Word 文档中相同。

二、批量转换 Word 文档

使用循环的方式批量转换

三、PDF 加密

首先我们来尝试对一个文件进行加密操作，在 D 盘根目录下创建一个名为“python_pdf_secret”的文件夹，在其中放入一个名为 “test.pdf”的文件。

然后编写如下代码。

```
from PyPDF2 import PdfFileWriter,PdfFileReader

with open('D:\\python_pdf_secret\\test.pdf','rb') as pdf_obj:
    pdf_reader=PdfFileReader(pdf_obj)    # 读取要加密的文件
    pdf_writer=PdfFileWriter()
    # 将每一页的数据写入 pdf_writer 对象
```

```
    for page_num in range(pdf_reader.numPages):  # pdf_reader.numPages 为 PDF 文件的总页数
        page_obj=pdf_reader.getPage(page_num)
        pdf_writer.addPage(page_obj)
    pdf_writer.encrypt('123') # 加密,将密码设为'123'

#以二进制的方式写入加密文件，将保留源 PDF 文件中的所有信息
pdf_output_file=open("D:\\python_pdf_secret\\test_sec.pdf",'wb')
    pdf_writer.write(pdf_output_file)
    pdf_output_file.close()
```

运行这段代码可以看到 “python_pdf_secret”目录下生成了一个名为 “test_sec.pdf”的文件，打开该文件则会弹出一个提示窗口让你输入密码，如图 8-30 所示。

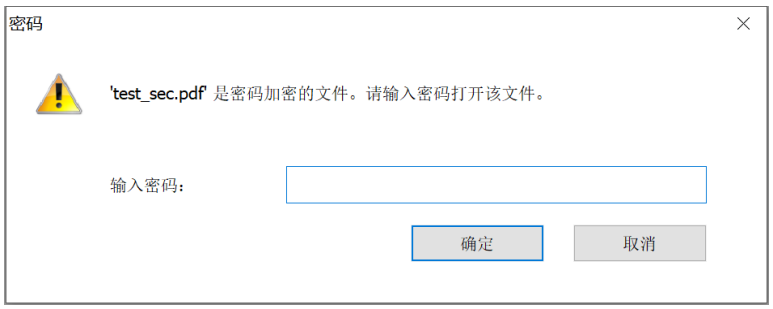


图 8-30 密码输入框

三、PDF 批量加密

使用循环方式批量加密

第四课时

（批量提取 PDF 中的文本数据）

一、从 PDF 中提取文本数据

首先我们在 D 盘根目录下创建一个名为“test.pdf”的文件，在该 pdf 文件中写入邀请函的数据，也可以直接将任务 8.2.1 中生成的 test.pdf 文件直接复制到 D 盘根目录下。

接下来使用 pdfplumber 获取 PDF 文档中的文本数据，代码如下。

```
import pdfplumber
pdf = pdfplumber.open('D:\\test.pdf')
```

```
page = pdf.pages[0]      # 获取第一页的数据
text = page.extract_text()
print(text)
pdf.close()
```

运行结果如下。

年会邀请函

尊敬的供应商 1 王波先生/女士：

ABCD 有限公司为感谢您的责任与关爱，我们敬邀并热切盼望与您共聚，乐享我公司

举办的 2021 年年终总结与迎新年晚会。

时间：2021 年 1 月 11 日 19:00-21:00

地点：DEF 大酒店 999 贵宾厅

ABCD 公司全体员工成之期盼您的光临！

总经理：张三

2020 年 12 月 15 日

使用 pdfplumber 库的 page.extract_text()函数可以非常方便地读取 PDF 文件中的内容，不过这个例子中我们只能获取一页的数据，如果一个 PDF 文档有多页数据，则可以使用如下代码。

```
import pdfplumber
pdf = pdfplumber.open('D:\\test.pdf')

text = ''
for page_num in range(0,len(pdf.pages)):
    page = pdf.pages[page_num]
    text = text + page.extract_text()

print(text)
pdf.close()
```

使用 len(pdf.pages) 函数可以获取 PDF 文档的总页数，这

样就可以通过循环将 PDF 文档中的内容全部提取出来。

二、批量读取 PDF 中的文本

要提高效率还得批量地提取数据，接下来我们以 8.2.2 小节中生成的 PDF 文件为例，提取 D 盘下“python_practice_pdf”目录下所有的 PDF 文件中的内容，并将文本保存到 D 盘下的“pdf_text”目录（需要提前创建该文件夹）下

编写代码如下。

```
import pdfplumber
import os
# 读取 PDF 中的文本
def pdf2text(filePath):
    pdf = pdfplumber.open(filePath)
    text = ''
    for page_num in range(0,len(pdf.pages)):
        page = pdf.pages[page_num]
        text = text + page.extract_text()
    pdf.close()
    return text
# 批量解析 PDF 文件，生成 TXT 文件
def batchConvertPDF(pdfDirPath,textDirPath):
    fileList = os.listdir(pdfDirPath) # 获取文件夹中所有文件列表
    # 批量转换
    for filename in fileList:
        # 生成 pdf 文件名
        text = pdf2text(pdfDirPath + "\\"+ filename)
        filename = filename[:filename.rfind('.')] #截取文件前缀
        # 将 text 保存为 TXT 文件
        with open(textDirPath + "\\"+  filename + "_text.txt",'w') as textFile:
            textFile.write(text)

pdfDirPath = 'D:\\python_practice_pdf'      # PDF 文件目录
textDirPath = 'D:\\pdf_text'                # TXT 文件目录

batchConvertPDF(pdfDirPath,textDirPath)     #调用转换方法
```

运行代码，可以发现 D 盘下已经生成了对应的“txt 文件”，如图 8-34 所示。

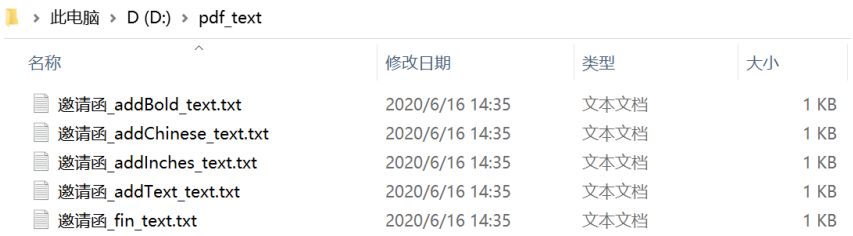


图 8-34 pdf 文件批量转换为文本文件

三、项目总结

在本项目中我们总共完成了三个程序。

1. 批量制作并发送年会邀请函；
2. Word 文档批量转换为加密 PDF 文档；
3. 批量提取 PDF 文档中的文本。

通过完成这三个程序，掌握了 Python 处理 Word 与 PDF 文档的知识，具体如下。

1. 使用 Python 生成 Word 文档，并且向 Word 文档中添加数据；
2. 使用 Python 发送邮件；
3. 将 Word 转换为 PDF 文档；
4. 批量地对 PDF 文档进行加密；
5. 从 PDF 文档中提取文本数据，并保存为 txt 文件。

作业

项目习题

1. 现在公司的领导需要你将 10 份邀请函全都整合成一个 Word 文档，然后将它们全都打印出来。
2. 有 10 份封面相同的 PDF 文档，需要你将他们合并为一个 PDF 文档，并且只保留一份封面，其他的封面都删除。

习题中有一些知识点是本文没有涉及到的，但是解决这些问

	题的思路是一致的，顺着批量处理的思路相信你肯定能找到解决方案！
教学后记	本章介绍了 Python 处理 Word 和 PDF 文件的方法，学生对文档的创建和转换表现出较高的兴趣，但在处理文件路径和批量操作时仍存在一些困难。部分学生对文件操作的逻辑理解不够深入，导致在实现功能时出现错误。后续课程中需要通过更多实例帮助学生巩固文档处理的操作。

课题名称	项目九：自动化办公小助手（Excel 数据处理+邮件通知）	计划学时	2 学时
教学内容	用 Python 自动化处理 Excel 数据 自动发送带图表的邮件通知		
教学目标及基本要求	1. 掌握使用 pandas 库读取、清洗和输出 Excel 数据的方法。 2. 学会通过 Python 脚本自动统计表格数据（如求和、去重）。 3. 能够生成可视化图表（柱状图/折线图）并保存为图片。		
教学重点	1. pandas 的基本操作（数据筛选、聚合）。 2. 数据可视化（Matplotlib/Seaborn）。		
教学难点	数据清洗（处理缺失值、重复值）。 图表的自定义样式（标题、坐标轴标签）。		
教学方式	教学做一体化		
教学过程	**第一、二课时** **（项目介绍、批量制作并发送年会邀请函）** **一、项目场景，导入课程** 情境导入 展示一份杂乱的学生成绩表（含重复数据、空值），提问：“如何快速整理并分析成绩？” **二、知识讲解** 识讲解		

1. 安装依赖库：

pip install pandas matplotlib seaborn openpyxl

代码演示：

读取 Excel：

import pandas as pddf = pd.read_excel("scores.xlsx")

数据清洗：

df.drop_duplicates() # 去重 df.fillna(0, inplace=True) # 空值填充

数据统计：

average_score = df["数学"].mean()top_students = df[df["总分"] > 250] # 筛选总分前几名

可视化：

import matplotlib.pyplot as pltdf["语文"].plot(kind="bar") # 柱状图 plt.title("语文成绩分布")plt.savefig("chinese_scores.png")

邮件发送代码框架：

```python
import smtplib
from email.mime.text import MIMEText
from email.mime.image import MIMEImage
from email.mime.multipart import MIMEMultipart

# 发送方配置
sender = "your@qq.com"
password = "xxxxxx"  # QQ邮箱授权码
receiver = ["parent1@qq.com", "parent2@qq.com"]

# 构造邮件对象
msg = MIMEMultipart()
msg["From"] = sender
msg["To"] = ", ".join(receiver)
msg["Subject"] = "学生成绩通知"
```

	# 添加正文文本 text = """ 亲爱的家长： 以下是本次考试的成绩汇总： """ text_part = MIMEText(text, "html") msg.attach(text_part) # 添加图表附件 with open("chinese_scores.png", "rb") as f: image = MIMEImage(f.read()) image.add_header("Content-Disposition", "attachment", filename="scores_chart.png") msg.attach(image) # 发送邮件 server = smtplib.SMTP_SSL("smtp.qq.com", 465) server.login(sender, password) server.sendmail(sender, receiver, msg.as_string()) server.quit() 关键细节： QQ 邮箱需开启“接收方显示为发件人名称”。 使用 HTML 标签嵌入图片（如 <img src="cid:image1">）。 | 思政融入：合理使用自动化工具提升沟通效率，但需注意隐私保护。 拓展思考：能否结合 Excel 数据生成个性化邮件模板（如不同分数段的不同评语）？ |
作业	1. **基础任务**：编写脚本整理班级考勤表，生成每位学生的出勤率报告（邮件正文+图表）。 2. **挑战任务**：设计一个“自动发送周末学习计划邮件”的程序，包含个性化课程表和倒计时功能。
教学后记	项目九通过“数据处理+自动化工具”串联了 Python 的实际应用场景，学生不仅掌握了技术技能，还初步具备了“发现问题→分析需求→编程解决”的工程思维。未来教学中，需进一步平衡技术深度与课堂时间分配，同时加强数据伦理教育，为学生的技术成长打下扎实基础。