# 目 录

CONTENTS

# 项目1

# 学生成长档案——初来乍到，请多关照

## 情景导入


变量运算符——情景导入

青年兴则国家兴，中国发展要靠广大青年挺膺担当。年轻充满朝气，青春孕育希望。广大青年要厚植家国情怀、涵养进取品格，以奋斗姿态激扬青春，不负时代，不负华年。

大学是人生成才、成就事业的新起点，学习、生活、工作、社交等各方面都需要从这里开始去摸索、去思考、去实践。大学是青年人追逐梦想、展现才华的舞台，更是青年人成就梦想、追求真理的殿堂，在未来的大学生活中我们将收获知识、成长、快乐、健康，让青春在为祖国、为人民、为民族、为人类的奉献中焕发出更加绚丽的光彩。

师父：河小青，还记得你刚进入大学所做的第一件事情吗？

河小青：师父，刚进入校园，学校的志愿者同学热情地向我介绍校园生活、学习的方方面面，带着我到报道登记处进行新生入学登记。让新入学的我感受到了同学们“奉献、友爱、互助、进步”的志愿服务精神。

师父：在新生入学报到过程中，需要登记每位同学的基本信息，这是一项复杂且繁重的工作，会耗费很大的精力。

河小青：是的，师父。能否使用编程手段减少新生入学基本信息统计过程中的工作量呢？

师父：当然可以。使用 Python 中的变量、赋值语句、字符串、运算符等基础知识就可实现新生入学基本信息的登记，这样就可以将重复的信息统计工作交给计算机去处理啦。

本项目将学习搭建 Python 开发环境、Python 编程规范、常量与变量、数据类型、运算符、字符串等相关的知识点。通过项目学习，最终可以使用 Python 开发“新生入学登记信息”程序，方便记录和管理学生个人信息。

## 学习目标

### 知识目标

了解 Python 的发展历程和特点。

理解并掌握 Python 的基本语法。

理解并掌握 Python 的基本数值类型。

理解并掌握 Python 中的变量。

了解 Python 中的常用运算符及其优先级。

掌握字符串类型的概念和使用。

### 技能目标

能够独立搭建 Python 开发环境。

能够使用编程解决简单的实际问题。

### 素质目标

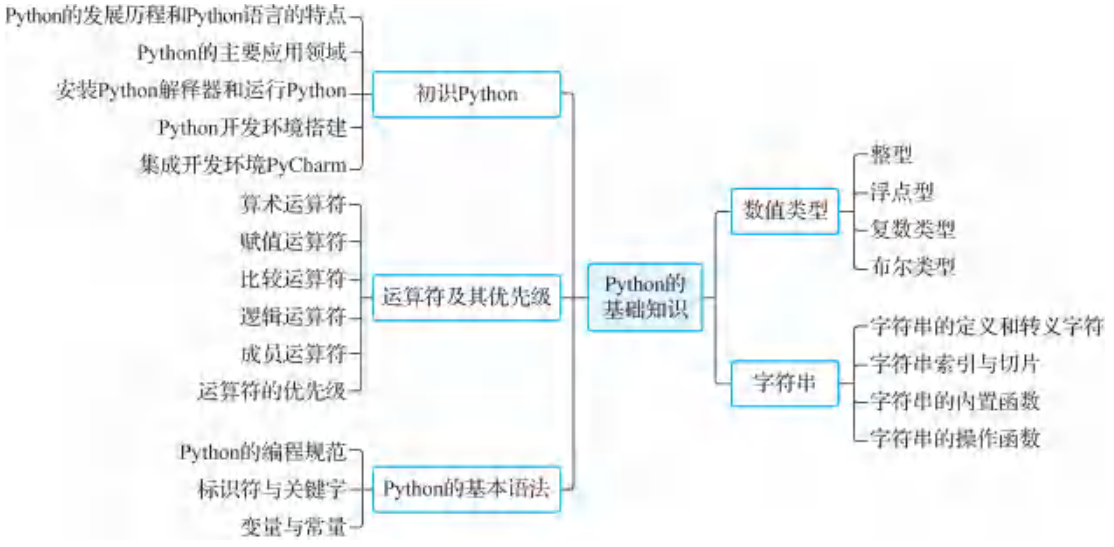
养成良好的编程习惯及编程风格。

确立务实创新、积极进取的学习态度。

培养精益求精的工匠精神。

提高政治素质，培养团队协作意识。

### 知识图谱



# 任务 1 搭建 Python 开发环境

## 课前预习单

### 一、任务场景

初入大学校园的你，将从此开启神秘而又充满魔力的 Python 探索之旅。在本任务中，可以了解到 Python 的发展历程、特点及应用，你将会发现它具有无限潜力。在深入学习 Python 编程之前，先搭建好稳定好用的 Python 开发环境将是一个不错的选择。

### 二、预习任务

分组完成下列任务，组长组织各组员查找相关资料，观看线上精品课视频、完成预习活动，并对收集的资料进行讨论和整理，记录学习过程中遇到的疑难问题。

1. 扫描二维码，观看视频，并完成下列题目。


初识 Python

（1）什么是程序设计语言？

（2）Python 语言的基本特点是什么？

（3）你认为 Python 语言受欢迎的原因有哪些？

2. 扫描二维码，观看视频，并记录搭建 Python 开发环境的步骤。

（1）下载并安装 Python 解释器，下载地址为_____

_____

下载版本为_____

（2）下载并安装 PyCharm 集成开发环境。


项目环境搭建

（3）如何检验 Python 是否安装成功？

### 三、疑难问题

# 课堂任务实施

## 知识要点

### 1.1.1 初识 Python

#### 1. Python 的发展历程

随着大数据及人工智能（artificial intelligence，AI）的飞速发展，Python 已成为当今最流行的开源编程语言之一，越来越受到世界各国的重视。

Python 的创始人为荷兰的 Guido van Rossum（吉多·范·罗苏姆）。Python 一词来自 Guido 所挚爱的电视剧 *Monty Python's Flying Circus*，因此，Python 便作为该编程语言的名称。

Guido 在 1989 年的圣诞节假期开始编写 Python 语言的解释器。Guido 曾经参与 ABC 语言的设计与实现。ABC 语言是由荷兰国家数学与计算机科学研究中心（Centrum Wiskunde&Informatica，CWI）开发的以教学为目的的语言，具备良好的可读性和易用性，但也存在一些问题，如非开放性、可拓展性差、不能直接进行 I/O、过度革新及传播困难等。因此，Guido 决心在 Python 语言的实现中继承 ABC 语言的优点，解决 ABC 语言的问题，并结合 C 和 Shell 语言的特点，使 Python 成为功能全面、语法简洁优美、易学易用、可拓展的编程语言。

1991 年，Python 的第一个编译器诞生，它是面向对象的解释型语言，用 C 语言实现，并能够调用 C 语言的库文件。Python 具有面向对象的类、函数、文件及异常处理等机制，包括以字符串、列表、字典及元组为核心的数据类型，是以模块为基础的拓展系统。

Python 最初完全由 Guido 本人开发，随着 Python 功能的完善，Guido 的同事也加入 Python 的改进中来，逐渐形成 Python 的核心团队。随后，Python 拓展到 CWI 之外。Python 将许多机器层面上的细节隐藏起来交给编译器处理，这样，Python 程序员就可以花更多的时间用于思考程序的逻辑，而不是具体的实现细节。正是这一特点吸引了更多的程序员使用 Python，使得 Python 逐渐流行起来，2011 年 1 月，Python 赢得了 TIOBE 编程语言排行榜 2010 年度语言的桂冠。

#### 2. Python 语言的特点

Python 语言主要具有以下几个特点。

（1）简单易学。Python 的语法极其简单，虽然 Python 是用 C 语言写的，但它抛弃了 C 语言中的指针，从而使 Python 的语法得到简化。Python 代码具有伪代码的本质，这就使程序编写者能够专注于解决问题，而不是去搞明白语言本身。

（2）解释性。Python 可以直接从源代码运行。在计算机内部，Python 解释器将源代

码转换为字节码的中间形式，然后将它翻译成计算机使用的机器语言。不需要担心程序的编译问题，这就使得使用 Python 变得更加简单，更加易于程序的移植。

（3）可移植性。由于 Python 开源的本质，Python 已经被移植到很多平台上，包括 Linux、Windows、macOS 及 Google 基于 Linux 开发的 Android 平台。

（4）可扩展性。如果需要一段关键代码运行得更快或希望某些算法不公开，可以将部分程序用其他语言编写出来，如 C 或 C++，然后在 Python 程序中使用它们。

（5）面向对象。Python 既支持面向过程的编程，也支持面向对象的编程。在面向过程的编程语言中，程序是由过程或可重用的代码构建起来的。在面向对象的编程语言中，程序可以通过组合和继承定义类构建起来，但与 C++ 或 Java 等面向对象的语言相比，其实现面向对象编程的方式更为简单。

### 3. Python 的主要应用领域

Python 的主要应用领域如下。

（1）Web 应用开发。Python 现已成为 Web 开发的主流语言之一，其有丰富的 Web 开发框架，如 Django、Flask、web.py、Zope2、Tornado、CubicWeb 及 Web2py 等。其中，Python + Django 框架最为流行，其应用范围广、开发速度快、学习门槛低，可以让程序员轻松地开发和管理复杂的 Web 程序。

（2）自动化运维。Python 在自动化运维方面已经成为运维工程师的首选编程语言。业内使用最为广泛的自动化运维平台 Ansible 和 SaltStack 都是基于 Python 的，因此其具有良好的二次开发扩展能力。

（3）网络安全。自 2013 年“棱镜门”事件后，网络安全问题越发引起世界各国的重视。Python 作为高级编程语言的一种，编程简单、易学及易用，且有强大的第三方编程模块的支持，越来越受网络安全维护人员的青睐。一些常用的网络安全工具都是基于 Python 语言编写的，如 Scapy、Pcapy 及 Sulley 等。

（4）网络爬虫。在编写网络爬虫程序的众多语言中，Python 起到了举足轻重的作用，其中，Scrapy 爬虫框架应用得非常广泛，可以应用在数据挖掘、信息处理或存储历史数据中。此外，Python 还提供了多个网络爬虫框架以满足不同的应用需求，如 PySpider、Crawley、Newspaper、BeautifulSoup、Grab 及 Cola 等。

（5）游戏开发。Python 在游戏开发方面也有很多应用，如专门为游戏开发而设计的模块 pygame 能够使开发者快速开发出自己的游戏。相比于 Lua 或 C++，Python 比 Lua 具有更高阶的抽象能力，可以用更少的代码描述游戏的业务逻辑。例如，用 C++ 开发游戏，有时必须写一些扩展，从而增加游戏的代码量。

（6）数据分析。Python 拥有丰富的第三方库，如 Pandas、NumPy、Matplotlib 及 SciPy 等，可以让程序编写人员完成各种数据分析需求。

（7）人工智能。Python 在人工智能、神经网络、深度学习方面有强大而丰富的库。Python 是面向对象的动态语言，且适用于科学计算，这就使得 Python 在人工智能方面备

受青睐。目前，世界优秀的人工智能学习框架，如 Google 的 TensorFlow、Facebook 的 PyTorch 及开源社区的神经网络库 Keras 等均是用 Python 来实现的。

### 数字先锋

“人生苦短，我用 Python——Life is short,you need Python”，这句话是 Python 的创始人吉多·范·罗苏姆所说的。他在荷兰国家数学与计算机科学研究中心工作期间，参与研发了一种高级编程语言——ABC 语言，这款以教学为目的的语言相比当时的 BASIC、C 语言更加容易阅读、使用、记忆和学习，但是并没有流行起来。于是他抱着将编程语言变得“让用户感觉更好”的希望，从此前的 ABC 语言借用了一些特性，并于 1991 年 1 月公布了 Python 的第一个公开发行版。目前，Python 已成为全世界最欢迎的编程语言之一。同时因其在大数据和 AI 领域应用广泛，被称为人工智能时代的第一编程语言。

吉多·范·罗苏姆曾获得由自由软件基金会（Free Software Foundation，FSF）颁发的 2001 年自由软件进步奖，荷兰 UNIX 用户小组奖。同时，他也被美国计算机协会（Association for Computing Machinery，ACM）认定为著名工程师。尽管如此，他一直没有停下工作进取的脚步，曾先后在马里兰州国家标准及技术协会（NIST）、弗吉尼亚州 Reston 国家研究创新联合会（CNRI）、Google 和以 Python 建立主要架构的云服务提供商 Dropbox 工作，并于 2019 年 10 月宣布退休。2020 年，这位科技巨头宣布加入微软的开发部门，又变成了“奋斗一族”。

## 1.1.2 Python 开发环境搭建

工欲善其事，必先利其器。在正式进行 Python 开发之前，需要先搭建 Python 开发环境。由于 Python 的跨平台性，使其可以在如 Windows、macOS 和 Linux 等不同操作系统上进行编程和运行。读者可以根据需要在相应平台上安装 Python，本书中的案例是以 Windows 为开发平台实现的，因此，重点介绍如何在 Windows 平台上搭建 Python 开发环境。

### 1. 安装 Python 解释器

Python 是一种面向对象的解释型程序设计语言，Python 环境的核心组件就是 Python 解释器，它负责解释和执行 Python 代码。因此需要先下载和安装 Python 解释器。

（1）访问 Python 官网 https://www.python.org，根据自身的需求选择合适的版本下载安装包。通常建议下载最新的稳定版，同时考虑与本书项目的兼容性。

（2）双击下载好的 exe 安装文件进行安装，可以选择默认安装方式“Install Now”，也可以使用自定义安装方式“Customize installation”，灵活选择启用或禁用 Python 的某些功能等，如图 1-1-1 所示。


图 1-1-1　Python 安装方式选择

提示：选中“Add python.exe to PATH”复选框，安装程序就会帮助用户自动添加环境变量到系统中。

### 2. 运行 Python

按 Windows 徽标 +R 快捷键，在弹出的“运行”对话框中输入 cmd，单击“确定”按钮。在打开的命令提示符窗口中输入命令 python，并按 Enter 键，若在窗口中显示 Python 的版本信息，则说明 Python 安装成功，如图 1-1-2 所示。


图 1-1-2　安装成功命令提示符窗口输出信息

### 3. 集成开发环境 PyCharm

PyCharm 是由 JetBrains 打造的一款 Python 集成开发环境（integrated development environment，IDE），带有一整套可以帮助用户在使用 Python 语言开发时提高工作效率的工具，如调试、语法显示、Project 管理、代码跳转、智能提示、自动完成、单元测试及版本控制等。

（1）访问 PyCharm 官网，进入 PyCharm 下载页面，这里选择适合学习用的社区版本（community edition）下载。

（2）双击下载的安装文件，进入 PyCharm 安装界面，然后单击“Next”按钮，如图 1-1-3 所示。

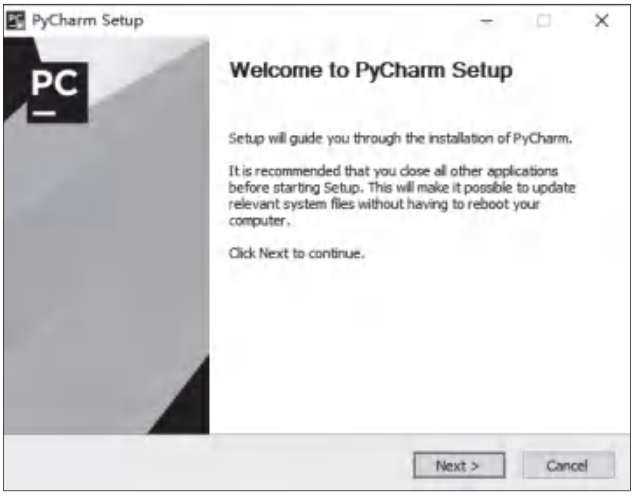



图 1-1-3　PyCharm 安装界面

（3）按照提示安装完成后单击“Finish”按钮，便可以使用 PyCharm 了。

## 任务实现

通过知识讲解，我们了解了 Python 语言的基本特点，学会了搭建 Python 开发环境，完成 Python 和 PyCharm 工具的下载和安装后，新建 Python 文件，输出“Welcome to Python world！”字符串，以验证 Python 文件能够被正常编译和解析。

（1）双击 PyCharm 快捷方式图标，进入 PyCharm 创建项目界面，如图 1-1-4 所示。

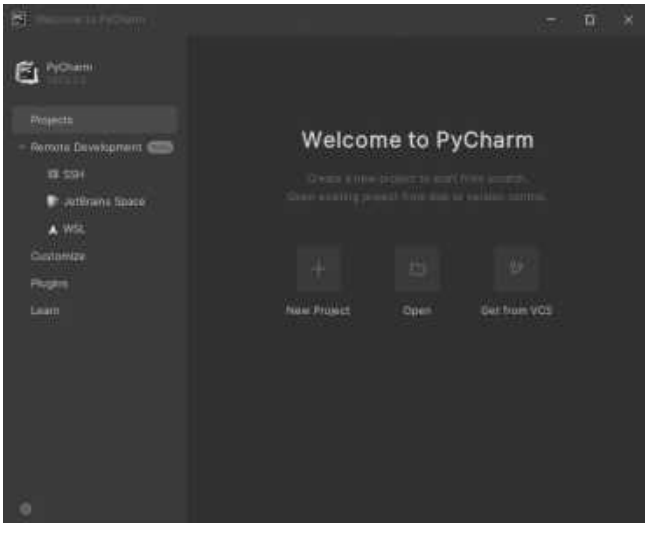



图 1-1-4　Python 创建项目界面

（2）单击“New Project”按钮，进入项目路径设置界面，设置好项目存放路径后单击“Create”按钮，项目创建完成。

（3）进入项目开发界面，右击项目名称，在弹出的快捷菜单中选择“New”→“Python File”选项，新建 Python 文件，如图 1-1-5 所示。

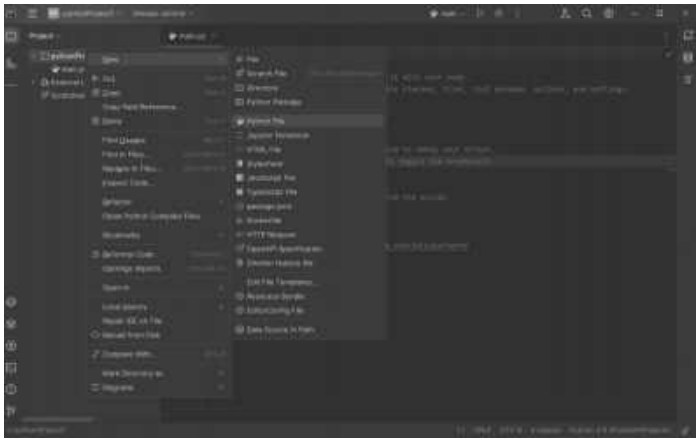


图 1-1-5　新建 Python 文件

在程序编辑区域中输入“print("Welcome to Python world!")”，右击文件名，在弹出的快捷菜单中执行“Run ‘first python file’”命令运行程序，运行窗口显示出字符串，说明环境配置成功，程序运行结果如图 1-1-6 所示。

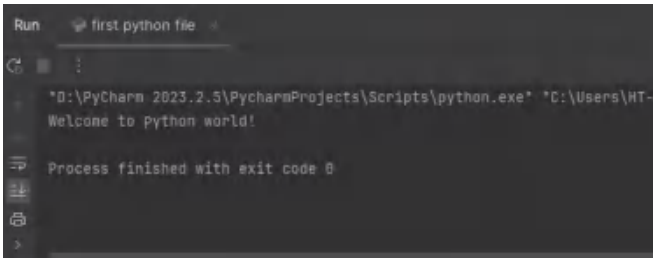



图 1-1-6　程序运行结果

# 课后拓展提升

## 赛题挑战

1. 以下选项中，（　　）不是 Python IDE。

A. PyCharm　　B. Jupyter Notebook

C. Spyder　　D. R Studio

2.IDLE 环境的退出命令是（　　）。

A. exit()　　B. cls()

C. close()　　D. esc()

3. 下列命令可用于列出本机已安装的所有 Python 软件包的是（　　）。

A. pip show　　B. pip info

C. pip search　　D. pip list

## 创新拓展

Python 开发工具多种多样，我们在本任务中学到了 PyCharm 这款功能强大的集成开发环境（IDE），实际上，Python 程序员可以选择的工具非常丰富。为了更加全面地了解这个领域，可以进行一些网络搜索，深入研究 Python 社区普遍使用的一些开发工具。在搜索的过程中，可以关注 Anaconda、PyDev、Eclipse、Visual Studio Code、Spyder 等工具，它们都有各自独有的特点和优势。

请将搜索到的工具的特点以表格形式列出，方便 Python 开发者根据个人或项目的需求进行选择。

# 任务 2 学生画像表示

## 课前预习单

### 一、任务场景

踏入新班级，面对新面孔，同学们或许怀有增进彼此了解、寻觅志同道合之友的期望。为实现这一目标，本任务将基于已搭建的 Python 开发环境，通过编程手段构建学生画像。在编写代码之前，须深入了解 Python 语言的编程规范，熟练掌握变量的运用，并灵活运用数字类型以精确描述学生的基本信息。

### 二、预习任务

分组完成下列任务，组长组织各组员查找相关资料，观看线上精品课视频、完成预习活动，并对收集的资料进行讨论和整理，记录学习过程中遇到的疑难问题。

1. 扫描二维码，观看视频，并完成下列题目。

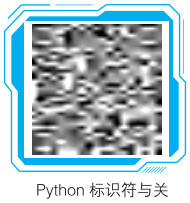

Python 编程规范

（1）为什么要遵循 Python 语言的编程规范？

（2）Python 中的注释有哪些类型？它们的作用分别是什么？

（3）描述一名学生的画像通常需要哪些信息？它们的数据类型是什么？

2. 扫描二维码，观看视频，并完成下列题目。


Python 标识符与关键字

（1）a="-5"，a 在 Python 中是（　　）数据类型。

A. 整型　　B. 浮点型　　C. 字符串　　D. 布尔型

（2）（　　）属于 Python 语言中标识符的命名规则。

A. 标识符由字母、数字和下划线组成　B. 必须以数字开头

C. Python 中的标识符不区分大小写　D. Python 中的标识符可以使用“-”符号

（3）Python 使用 _____ 格式划分语句块。

### 三、疑难问题

# 课堂任务实施

## 任务描述

金秋时节，校园内迎来了一批崭新的面孔。来自四面八方的学子们，专攻不同的专业，即将在共同的校园中展开全新的学术与人生旅程。

当我们怀揣着激动与憧憬的心情踏入这片学术圣地时，自然对周围同学的个人背景产生了浓厚的兴趣，包括他们的兴趣爱好、姓名、籍贯等。为了促进新生间的快速了解与交流，本任务将运用 Python 语言构建一个程序来描绘学生的基本画像。该程序将统计并整理学生的姓名、学号、性别、所属学院、班级名称、籍贯、政治面貌等关键信息。

让我们带着任务进入学习阶段，深入思考在进行学生画像描绘过程中所涉及的数据类型、变量设定及相应的编程方法。

## 知识要点

### 1.2.1 Python 的编程规范

#### 1. Python 的缩进规则

Python 使用缩进来定义代码块，而不是使用其他语言中常见的花括号 {}。这种做法是 Python 设计哲学的一个特点，称为“显式优于隐式”。缩进规则如下。

（1）缩进级别。每个代码块（如函数定义、循环、条件语句、类定义等）都应该有相同的缩进级别。同一代码块内的所有语句应该有相同的缩进。

（2）缩进方式。可以使用空格或制表符（Tab）进行缩进，但最好是选择其中一种方式，不要混用。Python 的 PEP 8 样式指南推荐使用 4 个空格进行缩进。

（3）一致性。在整个代码库中应该保持一致的缩进风格。不一致的缩进会导致 IndentationError，这是 Python 中的一种语法错误。

（4）悬挂缩进。在长语句中，如果需要将一行拆分为多行，可以使用悬挂缩进，即在第二行及以后的行上进行缩进，以便于阅读和理解。

在 Python 中，if 语句、while 循环语句及 for 循环语句需要缩进。缩进的方法为按一次 Tab 键或四次空格键。

【例 1-2-1】 Python 中缩进方法的使用。

```
# if 语句的缩进方法
a = 7
```

```
if a > 0:
    print("hello")      # 按一次 Tab 键或四次空格键缩进
# while 语句的缩进方法
a = 0
while a < 7:
    print(a)            # 按一次 Tab 键或四次空格键缩进
    a += 1              # 按一次 Tab 键或四次空格键缩进
```

例 1-2-1 代码第 4 行的 print 函数前面进行了代码缩进，按一次 Tab 键或四次空格键表示一个缩进。程序运行结果如图 1-2-1 所示。

```
hello
0
1
2
3
4
5
6

Process finished with exit code 0
```

图 1-2-1 代码缩进

### 2. Python 的注释

注释是对程序代码进行解释说明的，其本身不会被执行。它可以提升代码的可读性，方便日后检查和修改代码。使用注释也可以将暂时不用的程序代码屏蔽，将来需要时再将注释取消。

注释的内容对程序的运行结果没有影响。Python 中的注释分为单行注释和多行注释。

（1）单行注释。Python 中的单行注释以 # 开头，# 后为注释的内容。

（2）多行注释。Python 的多行注释用三引号包含要注释的内容，可以是三个单引号或双引号。

### 3. Python 中的引号

Python 支持三种类型的引号，分别是单引号 (' ')，双引号 (" ") 及三引号 ('' '' 或 """ """)。引号用于标识字符串字面量，它们告诉 Python 解释器哪些字符应该被视为文本而不是代码的一部分。

单引号和双引号是等效的，可以用来定义单行字符串。

三引号用于定义多行字符串或包含单引号、双引号的字符串，而不需要在字符串内部使用转义字符。

Python 中的引号有以下两个作用。

（1）表示注释。三对单引号或双引号表示单行或多行注释。

（2）用于定义字符串。

①单引号定义字符串，如定义 fruit='apple'，print(fruit) 的结果为 apple。

②双引号定义字符串，如定义 fruit="apple"，print(fruit) 的结果为 apple。

③三引号定义字符串，如定义 fruit='''apple''' 或 """apple"""，print(fruit) 的结果为 apple。

④单双引号嵌套，如定义 fruit="'apple'"，print(fruit) 的结果为 'apple'；定义 fruit='"apple"'，print(fruit) 的结果为 "apple"。

【例 1-2-2】 注释及引号的使用。

```
# 这是一个单行注释
""" 这是由三对双引号
构成的多行注释 """
name = ' 河小青 '   # 这是一个单行注释，该行使用单引号表示一个字符串
print(name)
name = " 河小青 " # 双引号的使用
print(name)
name = ''' 河小青 ''' # 三引号的使用
print(name)
name = """ 河小青 """ # 三引号的使用
print(name)
name = '" 河小青 "'
print(name)
```

在例 1-2-2 中，符号 # 后面的内容是单行注释，代码使用三对双引号进行多行注释，并通过单引号、双引号、三引号三种方式定义了字符串。关于字符串相关的知识点我们将在本项目的任务 4 中进行详细讲解，程序运行结果如图 1-2-2 所示。

```
河小青
河小青
河小青
河小青
"河小青"

Process finished with exit code 0
```

图 1-2-2　注释及引号的使用

## 1.2.2　Python 中的标识符与关键字

### 1. Python 中的标识符

在 Python 中，标识符是用户为变量、函数、类、模块和其他对象指定的名字。标识符的命名规则如下。

（1）合法性。标识符必须以字母（A~Z，不区分大小写）或下划线（_）开头，后面

可以跟任意数量的字母、数字（0~9）或下划线。

（2）区分大小写。Python 是大小写敏感的，因此，variable、Variable 和 VARIABLE 是三个不同的标识符。例如，a 和 A 是两个不同的标识符。

（3）长度限制。标识符的长度没有硬性限制，但应该保持在合理范围内，以便于阅读和理解。

（4）关键字避免。标识符不能是 Python 的保留关键字，如 if、else、while、True、False 等。这些关键字有特定的语言意义，不能用作普通标识符。

标识符在命名时要做到见名知意。尽量起一个有意义的名字，做到一眼就知道是什么意思，这样可以提高代码的可读性。例如，表示名字的标识符为 name，表示学生的标识符为 student。

合法标识符：x、b、id、score1、student_name2、Func。

非法标识符：2Name、tea@cher、*abc、26。

### 2. Python 中的关键字

关键字即预定义保留标识符，在 Python 语言中定义了一些专有词汇，统称为关键字，它们都有特定的含义，只能用于特定的位置。这些关键字存放在 keyword 模块中。

【例 1-2-3】 查看 Python 中的关键字。

```
import keyword
print(keyword.kwlist)
```

例 1-2-3 的代码中使用关键字 import 导入了 keyword 模块，然后使用 print 函数打印出该模块中的 kwlist 变量。程序运行结果如图 1-2-3 所示。

```
['False', 'None', 'True', '__peg_parser__', 'and', 'as', 'assert', 'async', 'await', 'break',
 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from',
 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise',
 'return', 'try', 'while', 'with', 'yield']

Process finished with exit code 0
```

图 1-2-3 Python 中的关键字

## 1.2.3 变量与常量

### 1. 变量的概念

人们在超市购物时，往往需要使用购物车存储物品；在上课时，用书包存放课本、文具、水杯等物品，如图 1-2-4 所示。这里存在一个现象，购物车存放的物品会随着使用人或使用场景的不同而发生变化。书包中的物品会随着课表的变动而发生更换。在 Python 语言中，将存储可变内容的量称为变量。变量的命名规则与标识符相似。

图 1-2-4　购物车与书包

【例 1-2-4】 河小青同学 18 岁，身高 175 厘米，体重 60.5 千克，使用变量表示河小青的基本信息。

```
studentName = "河小青"
print("姓名:", studentName)
studentAge = 18
print("年龄:", studentAge)
studentHeight = 175
print("身高:", studentHeight)
studentWeight = 60.5
print("体重:", studentWeight)
```

例 1-2-4 中第一行代码执行 studentName 等于 "河小青"，Python 解释器做了两个工作：在内存中开辟一个空间来存放字符串 "河小青"，在内存中创建了一个名为 studentName 的变量，并将它指向值"河小青"，此时 studentName 成了值"河小青"的引用。

程序运行结果如图 1-2-5 所示。

```
姓名: 河小青
年龄: 18
身高: 175
体重: 60.5

Process finished with exit code 0
```

图 1-2-5　变 量 命 名

## 2. 变量的赋值

Python 中使用变量不需要提前声明或定义，给变量赋值的过程就是对变量进行声明和定义的过程。给变量赋值的语法格式如下。

```
变量 = 值
```

其中，“=”称为赋值符号，读作将值赋给变量。赋值符号左侧必须是变量，赋值符号右侧可以是变量、值或表达式。

这里需要注意每个变量在使用前都必须赋值，赋值以后该变量才会被创建。如果变量没有被赋值就直接使用，会触发“NameError”异常。如图 1-2-6 所示，直接打印变量 yourName 会出现“NameError”异常。

```
NameError: name 'yourName' is not defined
```

图 1-2-6　未赋值变量异常

### 3. 多变量赋值

除了基本赋值方法外，还有链式赋值、解包赋值等赋值方式可以同时为多个变量赋值。

链式赋值是将同一个值赋予多个变量的一种赋值方式。其语法格式如下。

```
变量 1 = 变量 2 = 变量 3 = ... = 值
```

解包赋值是同时将多个值赋予多个变量的一种赋值方式。其语法格式如下。

```
变量 1, 变量 2, ..., 值 n = 值 1, 值 2, ..., 值 n
```

【例 1-2-5】 多变量赋值。

```
studentAge = yourAge = myAge = 18     # 链式赋值
print(studentAge, yourAge, myAge)
# 解包赋值
studentName, studentAge, studentHeight = 18, "河小青", 175
print(studentName, studentAge, studentHeight)
```

例 1-2-5 实现了为多个变量赋相同的值和不同的值。每一行赋值语句相当于 3 个赋值语句，这样的语句写起来更加简便。需要注意的是，在分别为多个变量赋值时，变量名和值之间按照先后顺序一一对应进行赋值。程序运行结果如图 1-2-7 所示。

```
18 18 18
18 河小青 175

Process finished with exit code 0
```

图 1-2-7　多变量赋值

**高手点拨**

在 Python 中，解包通常指的是将可迭代对象（如列表、元组）的元素分解成独立的变量。在变量赋值时可以将多个值解包赋给多个变量，如 a, b = 1, 2 。

```
# 解包一个列表，将列表中的三个元素分别赋值给 a,b,c
numbers = [1, 2, 3]
a, b, c = numbers
print(a)                    # 输出：1
print(b)                    # 输出：2
print(c)                    # 输出：3
```

### 4. 常量

与变量相反，在程序运行过程中，值不能被修改的量为常量。Python 中没有专门定义常量的方式，通常全部使用大写英文字母来定义常量名。例如，NAME = 'tony'，其本质还是变量。Python 常量包括数字、字符串、布尔值和空值。

## 1.2.4 数值类型

例如，统计学生的基本信息，如班级学生人数，同学们的年龄、学制、身高、体重及在学校每个月的消费，是否入团等。在 Python 中，需要使用不同的数字类型来表示这些数据。Python 的数值类型包括整型、浮点型、复数类型和布尔类型。

### 1. 整型

整型数据表示程序中用到的整数，如 −5、−4、−3、0、7、9 等。与大多数程序设计语言的整型精度有限不同，Python 语言整型的精度是不受语言本身限制的，开发者可以使用 Python 程序处理非常大的整数。

【例 1−2−6】 使用 Python 内置函数 pow 计算 3 的 100 次幂与 3 的 1000 次幂。

```
print("3 的 100 次幂为: ", pow(3, 100))
print("3 的 1000 次幂为: ", pow(3, 1000))
```

程序运行结果如图 1−2−8 所示。

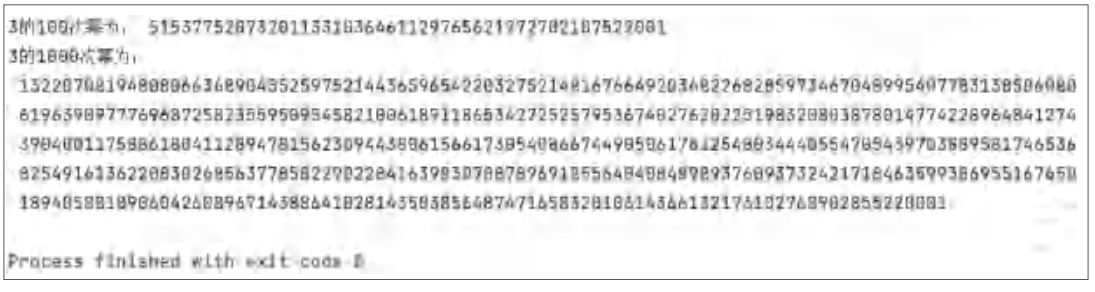



图 1−2−8 计 算 结 果

从计算结果可以看出，使用 pow 函数可以计算出很大的整数。当然，计算量越大，计算时间开销越高，消耗的内存空间越多。

Python 为整数类型提供了 4 种进制表示方式，分别是二进制、八进制、十进制和十六进制，表 1-2-1 中给出了不同进制整数的表示形式及各进制之间的转换方式。

表 1-2-1　不同进制整数的使用方法

| 进　制 | 说　明 | 例　子 | 进制转换 |
| --- | --- | --- | --- |
| 二进制整数 | 以正负号、0+ 小写 b（或大写 B）为前缀，后接数字 0 和 1 | 0b10111，-0B101 | bin(x) 将整数 x 转换为二进制 |
| 八进制整数 | 以正负号、0+ 小写 o（或大写 O）为前缀，后接 0 到 7 之间的数字 | 0o37、0O312 | oct(x) 将整数 x 转换为八进制 |
| 十进制整数 | 用正负号、数字 0~9 表示 | 18，-25 | int(x) 函数将整数 x 转换为十进制 |
| 十六进制整数 | 以正负号、0+ 小写 x（或大写 X）为前缀，后接 0 到 9 之间的数字、a 到 f 之间的字母（表示 10 到 15） | 0xff、0X1A2b3e | hex(x) 将整数 x 转换为十六进制 |

【例 1-2-7】 使用整数描述河小青同学的年龄、所在班级人数和学制。

```
studentAge, numberOfClass, lengthSchooling = 18, 45, 3
print("班级人数：", numberOfClass)
print("学制：", lengthSchooling)
print("学生年龄：", studentAge)
print("二进制学生年龄：", bin(studentAge))
print("八进制学生年龄：", oct(studentAge))
print("十六进制学生年龄：",  hex(studentAge))
```

例 1-2-7 中将三个整数 18、45、3 赋值给三个变量，表示河小青的年龄、所在班级人数和学制，然后使用 print 函数输出，再分别使用 bin、oct、hex 函数将十进制年龄转换成二进制、八进制和十六进制。程序运行结果如图 1-2-9 所示。

```
班级人数： 45
学制： 3
学生年龄： 18
二进制学生年龄： 0b10010
八进制学生年龄： 0o22
十六进制学生年龄： 0x12

Process finished with exit code 0
```

图 1-2-9　整数表示河小青同学的信息

### 2. 浮点型

班级同学的平均成绩、身高、每月的生活消费等信息通常是小数，在 Python 中使用浮点型表示。浮点型表示的是带有小数点部分的数字，如 1.234、-0.05、11.25 等。浮点数有小数形式和指数形式两种表示方式。

小数形式可以用来表示任意大小的实数，由正负号、数字 0~9 和小数点组成，如

1.75、−260. 58。

指数形式由尾数部分、指数部分和固定字符三部分组成，尾数部分用字母 a 表示，指数部分用字母 n 表示，固定字符用大写字母 E 或小写字母 e 表示。

【例 1-2-8】 用浮点数描述学生的身高、体重、月平均消费。

```
height, weight, avgConsume = 1.75, 60.5, 806.7
print("身高（米）: ", height)
print("体重（千克）: ", weight)
print("月平均消费: ", avgConsume)
```

例 1-2-8 中的代码使用多变量赋值分别为身高、体重及月平均消费三个变量赋值为 1.75、60.5 和 806.7，再使用 print 函数打印输出，程序运行结果如图 1-2-10 所示。

```
身高（米）： 1.75
体重（千克）： 60.5
月平均消费： 806.7

Process finished with exit code 0
```

图 1-2-10 浮 点 数

### 3. 复数类型

复数由实部和虚部两部分构成，在 Python 中复数的虚部以 j 或 J 作为后缀，具体格式如图 1-2-11 所示，其中，real 为复数的实部，imag 为复数的虚部。


图 1-2-11 Python 中复数的构成

复数的创建方式有两种，第一种方式是直接创建，即 real + imagj，第二种方式是使用 complex 函数创建，即 complex(real, imag)。

使用“复数 .real”和“复数 .imag”形式分别获得复数的实部和虚部。

【例 1-2-9】 使用复数表示方程的虚根。

```
s1 = -1.0 + 1.0j
s2 = complex(-1.0, -1.0)
print("方程的一个虚根为: ", s1)
print("方程的另一个虚根为: ", s2)
print("根 s1 的实部为 :", s1.real)
print("根 s1 的虚部为 :", s1.imag)
```

例 1-2-9 代码中将方程的两个根 -1.0 + 1.0j 和 -1.0 - 1.0j 分别赋值给变量 s1 和 s2，通过 s1.real 和 s1.imag 分别获得 s1 的实部和虚部。需要注意的是，复数的实部与虚部数值均为浮点类型。程序运行结果如图 1-2-12 所示。

```
方程的一个虚根为： (-1+1j)
方程的另一个虚根为： (-1-1j)
根s1的实部为： -1.0
根s1的虚部为： 1.0

Process finished with exit code 0
```

图 1-2-12　复数的创建与实部和虚部的获取

### 4. 布尔类型

人们在日常学习和生活中常常被问及：你成绩及格了吗？你成年了吗？人们往往以是或否来回答。在 Python 中，使用布尔类型的数据来表示这种只有是或否形式的数据。

布尔类型的变量只有两种取值，对应着布尔代数中的“真”与“假”这两种状态。在 Python 中，用 True 表示真，用 False 表示假，它们一般用来检查判断条件是否成立，布尔类型名为 bool。

**【例 1-2-10】** 使用布尔类型数据描述河小青同学是否成年、是否为团员、是否为党员。

```
print("是否成年：", True)
print("是否为团员：", True)
print("是否为党员：", False)
```

程序运行结果如图 1-2-13 所示。

```
是否成年： True
是否为团员： True
是否为党员： False

Process finished with exit code 0
```

图 1-2-13　布尔类型数据

## 任务实现

请使用 Python 语言编写代码描述河小青同学的基本信息。

```
name = "河小青"
age = 18
```

```
height = 175
weight = 60.5
id = "0618532416"
institute = "电子信息工程学院"
className = "大数据2201班"
adult = False
```

## 课后拓展提升

### 赛题挑战

1. Python 中表示空类型的是（　　）。

A. NULL　　B. " "　　C. None　　D. empty

2. 语句 x=3==3 运行结束后，变量 x 的值是（　　）。

A. 3　　B. 1　　C. False　　D. True

3. 以下表达式（　　）输出结果为 11。

A. print("1+1")　　B. print(1+1)

C. print(eval("1+1"))　　D. print(eval("1"+ "1"))

### 创新拓展

在本任务中，我们已经学习了 Python 中的输出语句。使用字符和 print( ) 函数可以打印出不同的图形。例如，打印三角形：

```
print("  /\")
print(" /  \")
print("/____\")
```

请试一试用不同的字符打印出正方形、圆形、五角星等图案。

# 任务 3 发展团员条件审核

## 课前预习单

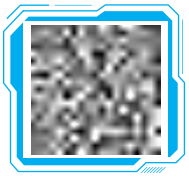
### 一、任务场景

又到了发展团员的时候，同学们纷纷报名，争相向我们的团组织靠拢，展现积极向上的精神风貌。本任务中，我们将要对各种类型的数据进行细致的运算操作，需要熟练地使用 Python 运算符，以便准确地进行团员条件的判断和选拔。

### 二、预习任务

分组完成下列任务，组长组织各组员查找相关资料，观看线上精品课视频、完成预习活动，并对收集的资料进行讨论和整理，记录学习过程中遇到的疑难问题。

**1. 扫描二维码，观看视频，并完成下列题目。**



Python 中的算术运算符和赋值运算符

（1）根据运算符的功能，运算符可分为哪几种类型？

（2）简单解释运算符 / 和 // 的区别？

（3）若 a = 3，b -= 2，则 a += b 的结果为 _____ 。

**2. 扫描二维码，观看视频，并完成下列题目。**

Python 中的比较运算符 逻辑运算符和成员运算符

（1）不是 Python 的逻辑运算符的是（　　）。

A. and　　B. or　　C. not　　D. in

（2）在 Python 中，（　　）运算符可以用于比较两个值是否相等。

A. 等于 (=)　　B. 大于 (>)

C. 小于 (<)　　D. 大于等于 (>=)

（3）Python 的成员运算符包含 _____ 和 _____ 。

### 三、疑难问题

# 课堂任务实施

## 任务描述

为进一步汇聚青春力量，吸纳更多优秀学子加入共青团，学院将进行团员选拔。选拔团员在班级基本条件的基础上，也要求学生在班级成绩优秀，对班级贡献度大，最好是班干部。

那么在选拔考核过程中，如何对学生成绩进行计算？竞选班委通过投票形式，如何判断票的多少？如何判断是否符合入团条件？这些需要对各种类型的数据进行运算操作，操作过程中使用的是运算符。本任务是使用运算符完成成绩计算和团员条件的审核判断，请带着任务学习 Python 运算符的使用。

## 知识要点

### 1.3.1 算术运算符和赋值运算符

#### 1. 算术运算符

Python 的算术运算符主要完成操作数的算术运算，包括加、减、乘、除、幂、取整等，具体描述及实例如表 1-3-1 所示（表中实例，假设变量 a 为 10，b 为 21）。

表 1-3-1 Python 的算术运算符描述及实例

| 运 算 符 | 描 述 | 实 例 |
| --- | --- | --- |
| + | 加，两个对象相加 | a+b，输出结果为 31 |
| – | 减，得到负数或是一个数减去另一个数 | a–b，输出结果为 –11 |
| * | 乘，两个数相乘或返回一个被重复若干次的字符串 | a*b，输出结果为 210 |
| / | 除 | b/a，输出结果为 2.1 |
| % | 取模，返回除法的余数 | b%a，输出结果为 1 |
| ** | 幂 | a**b，10 的 21 次幂，输出结果为 1 000 000 000 000 000 000 000 |
| // | 取整，返回商的整数部分 | 9//2，输出结果为 4；9.0//2.0，输出结果为 4.0 |

#### 2. 赋值运算符

赋值运算符“=”表示将右侧的值赋给左侧的变量或表达式。除简单的赋值运算符

外，Python 还提供了复合的赋值运算符，如表 1-3-2 所示。

表 1-3-2　Python 的赋值运算符描述及其实例

| 运　算　符 | 描　　述 | 实　　例 |
| --- | --- | --- |
| = | 简单的赋值运算符 | c = a + b，将 a + b 的运算结果赋值为 c |
| += | 加法赋值运算符 | c += a 等效于 c = c + a |
| -= | 减法赋值运算符 | c = a 等效于 c = c - a |
| *= | 乘法赋值运算符 | c *= a 等效于 c = c * a |
| /= | 除法赋值运算符 | c /= a 等效于 c = c / a |
| %= | 取模赋值运算符 | c %= a 等效于 c = c % a |
| **= | 幂赋值运算符 | c **= a 等效于 c = c ** a |
| //= | 取整赋值运算符 | c //= a 等效于 c = c // a |

【例 1-3-1】 河小青同学在新生入学考试中数学考了 85 分，语文考了 92 分，英语考了 87 分，编写代码帮助河小青同学计算他在这次考试中的平均成绩。

```
MathScore, ChineseScore, EnglishScore = 85, 92, 87
averageScore = (MathScore + ChineseScore + EnglishScore) / 3
print("河小青同学的平均成绩为: ", averageScore)
```

代码运行结果如图 1-3-1 所示，河小青同学的平均成绩为 88.0 分。

```
河小青同学的平均成绩为:  88.0

Process finished with exit code 0
```

图 1-3-1　算术运算符的使用

## 1.3.2　比较运算符、逻辑运算符和成员运算符

### 1. 比较运算符

比较运算符主要完成操作数的比较计算，结果通常是一个逻辑量（True 或 False）。Python 的比较运算符描述及实例如表 1-3-3 所示。

表 1-3-3　Python 的比较运算符描述及实例

| 运　算　符 | 描　　述 | 实　　例 |
| --- | --- | --- |
| == | 检查两个操作数的值是否相等，若是，则条件变为真 | 若 a = 3，b = 3，则 (a == b) 为 True |
| != | 检查两个操作数的值是否相等，若值不相等，则条件变为真 | 若 a = 1，b = 3，则 (a != b) 为 True |
| <> | 检查两个操作数的值是否相等，若值不相等，则条件变为真 | 若 a = 1，b = 3，则 (a <> b) 为 True。该运算符类似于 != |
| > | 检查左操作数的值是否大于右操作数的值，若是，则条件成立 | 若 a = 7，b = 3，则 (a > b) 为 True |
| < | 检查左操作数的值是否小于右操作数的值，若是，则条件成立 | 若 a = 7，b = 3，则 (a < b) 为 False |
| >= | 检查左操作数的值是否大于等于右操作数的值，若是，则条件成立 | 若 a = 3，b = 3，则 (a >= b) 为 True |
| <= | 检查左操作数的值是否小于等于右操作数的值，若是，则条件成立 | 若 a = 3，b = 3，则 (a <= b) 为 True |

### 2. 逻辑运算符

逻辑运算符用于将两个变量或表达式进行逻辑运算。Python 的逻辑运算符有与、或、非 3 种，如表 1-3-4 所示。

表 1-3-4　Python 的逻辑运算符描述及其实例

| 运　算　符 | 逻辑表达式 | 描　　述 | 实　　例 |
| --- | --- | --- | --- |
| and | x and y | 布尔“与”，如果 x 为 False，x and y 返回 False，否则返回 y 的计算值 | (a and b) 返回 20 |
| or | x or y | 布尔“或”，若 x 为 True，则返回 True，否则返回 y 的计算值 | (a or b) 返回 10 |
| not | not x | 布尔“非”，若 x 为 True，则返回 False；若 x 为 False，则返回 True | not(a and b) 返回 False |

**【例 1-3-2】** 班上举行班长选举投票，张强同学得了 36 票，河小青同学得了 41 票，判断张强同学是否能够当选为班长。

```
zhangQiang = 36
heXiaoQing = 41
print(zhangQiang > heXiaoQing)
```

程序运行结果为 False，如图 1-3-2 所示，说明张强不能当选班长。

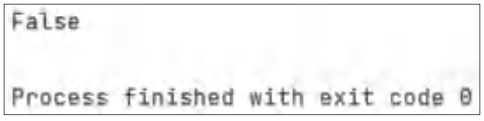



图 1-3-2　比较班长选票

### 3. 成员运算符

除了上述运算符之外，Python 还支持成员运算符，用于测试实例中是否包含了一系列成员，包括字符串、列表或元组。Python 的成员运算符描述如表 1-3-5 所示。

表 1-3-5　Python 的成员运算符描述

| 运　算　符 | 描　　述 |
| --- | --- |
| in | 如果在指定的序列中找到值，就返回 True，否则返回 False |
| not in | 如果在指定的序列中没有找到值，就返回 True，否则返回 False |

### 4. 身份运算符

身份运算符用于比较两个对象的存储单元，即判断两个标识符是否引自同一个对象，如表 1-3-6 所示。

表 1-3-6　Python 的身份运算符描述

| 运　算　符 | 描　　述 |
| --- | --- |
| is | 判断两个标识符是不是引自同一个对象，如果引用的是同一个对象，就返回 True，否则返回 False |
| is not | 判断两个标识符是不是引自不同对象，如果引用的不是同一个对象，就返回 True，否则返回 False |

## 1.3.3　运算符的优先级

在 Python 中，运算符是用于执行各种数学和逻辑运算的符号，不同运算符的优先级也各不相同。Python 运算符的优先级由高到低依次为括号运算符 (( ))，幂运算符 (**)，一元运算符 (+，−)，乘法 (*)，除法 (/)，取模 (%)，整除 (//)，加法 (+)，减法 (−)，比较运算符 (<、>、<=、>=、==、!=)，逻辑运算符 (and、or、not)，赋值运算符 (=、+=、−=、

*=、/=、%=、//=、**=)。

在编写代码时，应根据运算符的优先级来合理安排表达式的计算顺序，以确保得出正确的结果。还可以使用括号来明确指定运算的顺序，使代码更加清晰和易于理解。

## 任务实现

团员的必要条件：年龄在 14 周岁以上，28 周岁以下，没有受过处分。王强同学今年 18 岁，在校期间表现良好，编写代码判断王强同学是否符合入团条件。程序运行结果如图 1-3-3 所示。

```
age = 18
isPunished = False
print(age > 14 and age < 28 and isPunished == False)
```

```
True

Process finished with exit code 0
```

图 1-3-3　是否符合入团条件

### 知类通达

Python 中的 == 符号与 is 关键字是一样的吗?

双等号 (==) 是运算符，当赋的值与被比较对象的值相等时，双等号运算符返回 True。双等号运算符在比较对象时不匹配两个对象的内存位置，因此，即使两个对象的内存位置不同但值相同，双等号运算符也将返回 True。简而言之，双等号运算符比较对象是否相等。

is 是一个关键字，通过匹配两个或多个对象的内存位置来比较它们的身份。即使两个对象包含相同的项，如果对象不指向相同的内存位置，is 关键字也将返回 False。

通过将对象传递给 id() 函数，可以检查对象的内存位置。

# 课后拓展提升

## 赛题挑战

小蓝要和朋友合作开发一个显示时间的网站。在服务器上，朋友已经获取了当前时间，用一个整数表示，值为从 1970 年 1 月 1 日 00:00:00 到当前时刻经过的毫秒数。

现在，小蓝要在客户端显示这个时间。小蓝不用显示年月日，只需要显示时分秒即可，毫秒也不用显示，直接舍去即可。

给定一个用整数表示的时间，请输出这个时间对应的时分秒。

输入一行包含一个整数，表示时间。

输出时分秒表示的当前时间，格式形如 HH:MM:SS，其中，HH 表示时，值为 0~23，MM 表示分，值为 0~59，SS 表示秒，值为 0~59。时、分、秒不足两位时补前导 0。

样例输入如下。

```
46800999
```

样例输出如下。

```
13:00:00
1618708103123
```

## 创新拓展

小青是学校 100 年校庆的志愿者，需要统计优秀校友的信息，包括姓名、年龄、性别、毕业年份等。请利用学习的运算符知识写出表达式，划分出以下几类校友。

（1）近五年毕业的男性校友。

（2）年龄超过 60 岁或毕业超过 40 年的女性校友。

# 任务 4 新生入学思想教育

## 课前预习单

### 一、任务场景

在 Python 中，字符串是一种非常重要的数据类型，用于存储文本信息。本任务将运用字符串来制作校训等宣传栏目，让新同学了解校园文化。希望大家能够掌握字符串索引与切片及字符串的内置函数，学会使用字符串的操作方法。

### 二、预习任务

分组完成下列任务，组长组织各组员查找相关资料，观看线上精品课视频、完成预习活动，并对收集的资料进行讨论和整理，记录学习过程中遇到的疑难问题。

**1. 扫描二维码，观看视频，并完成下列题目。**


Python 中的字符串

（1）字符串的数据类型有哪些？

（2）字符串内置函数有哪些？

（3）对于新生入学思想教育，通常需要了解哪些信息？运用字符串如何存取这些信息？

**2. 扫描二维码，观看视频，并完成下列题目。**


字符串的操作方法

（1）下列（　　）不属于字符串。

A."I"　　B.'python'　　C."""^"""　　D.'1'.23

（2）下列关于字符串的说法，错误的是（　　）。

A. 字符串创建后可以修改

B. 字符串可以使用单引号、双引号和三引号定义

C. 转义字符 \n 表示换行

D. 格式符均由 % 和说明转换类型的字符组成

（3）拼接字符串可以使用 _____ 函数和运算符 _____ 。

### 三、疑难问题

# 课堂任务实施

## 任务描述

新生入学思想教育是大学教育的重要组成部分，对培养学生健康成长和全面发展具有重要意义。为了帮助新生更好地适应大学生活，让新生尽快了解校园生活，了解学校，了解专业，适应环境，学生会对新生进行一个简单的关于学校知识的小竞赛。例如，学校的校训是什么？核心价值观包括哪些内容？如何提取部分内容？如何输出一些个人专业信息？如何对这些内容进行排版？这些需要用到字符串的知识。本任务是运用字符串的知识实现学习心得的排版设计。

## 知识要点

### 1.4.1 字符串

#### 1. 字符串的定义

字符串是由零个或多个字符组成的有限序列，表示使用引号括起来的内容。

【例 1-4-1】 使用字符串表示：河南工业职业技术学院的校训是“厚德笃行，知行合一”。

```
xiaoxun = '河南工业职业技术学院的校训是“厚德笃学，知行合一”'
print(xiaoxun)
```

程序运行结果如图 1-4-1 所示，在字符串中使用了引号。

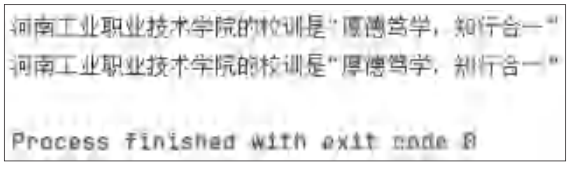



图 1-4-1 字符串输出

提示：当字符串包含双引号时，需要使用单引号或三引号作为界定符，如果使用双引号，那么系统无法正确判断界定符，会报错提醒。

#### 2. 转义字符

在 Python 程序中，当需要在字符中使用特殊字符时，需要用到以“\”表示的转义字符。常用转义字符的具体说明如表 1-4-1 所示。

表 1-4-1　常用转义字符的具体说明

| 转义字符 | 作用描述 | 转义字符 | 作用描述 |
|---|---|---|---|
| \( 在行尾 ) | 续行符 | \\ | 反斜杠符号 |
| \' | 单引号 | \" | 双引号 |
| \v | 垂直制表符 | \t | 水平制表符 |
| \n | 换行 | \f | 换页 |
| \a | 响铃 | \b | 退格 |
| \oaa | 八进制数（Unicode 码对应的字符，aa 代表的字符。例如，\o12 代表换行） | \xaa | 十六进制数（Unicode 码对应的字符，aa 代表的字符。例如，\x0a 代表换行） |
| \r | 回车 | | |

## 1.4.2　字符串索引与切片

### 1. 索引

所有有序的序列都是有索引概念的，区别在于序列可不可以被修改，索引可以理解为每个字符的下标。字符串中的每一个个体都可以看成一个元素，每一个元素都有其对应的索引。

提示：索引是从 0 开始的。字符串实际上就是字符的数组，所以也支持下标索引。

### 2. 切片

切片是指截取操作对象中一部分的操作。字符串、列表、元组都支持切片操作。

切片的语法格式如下。

```
[start : end : step]
```

提示：选取的区间属于左开右闭型，即从起始位开始到结束位的前一位结束（不包含结束位本身）。默认步长为 1。

【例 1-4-2】 利用索引和切片访问字符串。

```
s1 = '国家层面：富强、民主、文明、和谐'
s2 = '社会层面：自由、平等、公正、法治'
s3 = '个人层面：爱国、敬业、诚信、友善'
print(s1[5])
print(s2[8])
print(s3[-10])
print(s[5 : ])
```

程序运行结果如图 1-4-2 所示。

```
富
平
国
富强、民主、文明、和谐

Process finished with exit code 0
```

图 1-4-2　字符串索引

### 1.4.3　字符串的内置函数

#### 1. len( ) 函数

len( ) 函数用于求一个字符串所包含的字符的个数，即字符串的长度。

#### 2. str( ) 函数

str( ) 函数用于将任意类型的数据转换为字符串。

#### 3. ord( ) 函数和 chr( ) 函数

ord( ) 函数以一个字符作为参数，返回该字符对应的 ASCII 码值。chr( ) 函数与 ord( ) 函数的操作相反，chr( ) 函数根据给定的 ASCII 码值返回对应的字符串。

#### 4. max( ) 函数和 min( ) 函数

max( ) 函数和 min( ) 函数分别用于获取最大及最小 ASCII 码对应的字符。

【例 1-4-3】 计算给定字符串的 ASCII 码值及最大值和最小值。

```
s = "He Xiaoqing, Welcome! Your dream is beginning!"
print(" 字符串的长度: ", len(s))
print(" 字符串最小值: ", min(s))
print(" 字符串最大值: ", max(s))
print(" 第 10 个字符对应的 ASCII 码: ", ord(s[9]))
```

程序运行结果如图 1-4-3 所示。

```
字符串的长度：  46
字符串最小值：
字符串最大值：  u
第10个字符对应的ASCII码：  110

Process finished with exit code 0
```

图 1-4-3　字符串长度、最小值、最大值及 ASCII 码值的计算

### 1.4.4　字符串的操作函数

#### 1. find( ) 函数和 index( ) 函数

find( ) 函数用于检测指定字符串 sub 是否包含在当前字符串 s 中。若包含，则返回从左开始 sub 第一次出现的索引，否则返回 -1。

find( ) 函数的语法格式如下。

```
s.find(sub, [start, [end]])
```

在此语法中，sub 表示指定检测的字符串；start 表示开始索引，默认为 0；end 表示结束索引，默认为字符串的长度。

index( ) 函数和 find( ) 函数功能相同，不同在于当找不到子串时，index( ) 函数会触发“ValueError”异常。

【例 1-4-4】 在字符串中查找指定内容。

```
s = "Henan Polytechnic Institute"
print(s.find("e"))
print(s.rfind("e"))
print(s.index("e"))
print(s.rindex("e"))
print(s.find("q"))
print(s.index("q"))
```

程序运行结果如图 1-4-4 所示。从运行结果可以看出，如果找到了字符串，就返回对应的索引值，如果没有找到字符串，find( ) 函数返回 -1，而 index( ) 函数会触发“ValueError”异常。

```
ValueError: substring not found
1
26
1
26
-1

Process finished with exit code 1
```

图 1-4-4 使用 find( ) 函数查找指定内容

### 2. replace( ) 函数

replace( ) 函数将字符串中的 sub1 替换成 sub2，如果指定第 3 个参数，则替换次数不超过 count 次。

replace( ) 函数的语法格式如下。

```
s.replace(sub1, sub2[, count])
```

其中，s 表示字符串。sub1 为 s 中将被替换的子字符串，sub2 为 s 中的子串替换为的新字符串；count 则表示只替换前 count 个 sub1 子字符串，即替换次数不超过 count 次。返回结果为字符串中的 sub1( 原子字符串 ) 被替换成 sub2( 新子字符串 ) 后生成的新字符串，替换次数不超过 count 次。如果 s 中搜索不到子字符串 sub1，则无法替换，直接返回字符串 s( 不创建新字符串对象 )。

【例 1-4-5】 字符串替换。

```
s = "河小青是大数据 2201 班的一名学生"
print(s.replace("河小青","张强"))
print(s.replace("我","他"))
```

程序运行结果如图 1-4-5 所示。

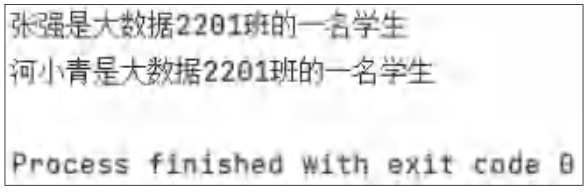



图 1-4-5 字符串替换

### 3. strip() 函数、lstrip() 函数和 rstrip() 函数

strip() 函数用于移除字符串头尾指定的字符 ( 默认为空格或换行符 )。

strip() 函数的语法格式如下。

```
s.strip([chars])
```

其中，s 代表指定检索的字符串，chars 代表移除字符串头尾指定的字符。返回结果为移除字符串头尾指定的字符后生成的新字符串，但不能删除中间部分的字符。

lstrip() 函数用于移除字符串头指定的字符 ( 默认为空格或换行符 )。

lstrip() 函数的语法格式如下。

```
s.lstrip([chars])
```

返回结果为移除字符串头指定的字符后生成的新字符串，但不能删除中间部分的字符。

rstrip() 函数用于移除字符串尾指定的字符 ( 默认为空格或换行符 )。

rstrip() 函数的语法格式如下。

```
s.rstrip([chars])
```

返回结果为移除字符串尾指定的字符后生成的新字符串，但不能删除中间部分的字符。

【例 1-4-6】 去除专业名称字符串中的空格。

```
major = "    大数据 技术          "
print(major.strip( ))
print(major.rstrip( ))
print(major.lstrip( ))
```

程序运行结果如图 1-4-6 所示。

```
大数据 技术
    大数据 技术
大数据 技术

Process finished with exit code 0
```

图 1-4-6　去除字符串中的空格

### 4. split( ) 函数

split( ) 函数通过指定分隔符对字符串进行切片，如果参数 num 有指定值，则分隔 mun+1 个子字符串。split( ) 函数的语法格式如下。

```
s.split(str = "", num = string.count(str))
```

其中，s 为字符串。str 代表分隔符，默认为所有的空字符，包括空格、换行符 (\n)、制表符 (\t) 等。num 代表分隔次数，默认为 -1，即分隔所有。这是一个非常重要的字符串操作函数，与 join( ) 函数的功能是相反的，用来将字符串分隔成序列。

提示：若不指定分隔符，则字符串中的空白符号（空格、换行符、制表符）的连续出现都将被认为是分隔符。

### 5. join( ) 函数

join( ) 函数将多个字符串按顺序合并成一个新的字符串。join( ) 函数的语法格式如下。

```
s.joincstr
```

其中，s 是连接符，是用于连接字符串 str 中每个字符的符号。

### 6. 字符串内容对齐函数

在 Python 中，可以使用 ljust( )、rjust( ) 和 center( ) 函数来对字符串进行对齐。

str.ljust(width[, fillchar])：返回一个原字符串左对齐，并使用 fillchar（默认空格）填充至 width 长度的新字符串。

str.rjust(width[, fillchar])：返回一个原字符串右对齐，并使用 fillchar（默认空格）填充至 width 长度的新字符串。

str.center(width[, fillchar])：返回一个原字符串居中对齐，并使用 fillchar（默认空格）填充至 width 长度的新字符串。

提示：如果指定的长度小于原字符串的长度，则返回原字符串。

**【例 1-4-7】** 字符串的分割与拼接。

```
s = "My major is big data"
print(s.split( ))   # 默认按空格进行分割
website = "www.hnpi.cn"
```

```
print(website.split("."))   # 指定按“.”对字符串进行分割
print("-".join(website.split(".")))   # 字符串拼接
website1 = website .rjust(20) # 调整字符串 website 的宽度为 20 且右对齐
print(website1 )
```

例 1-4-7 分别使用 split( ) 函数和 join( ) 函数完成了字符串的分隔与拼接，拼接符号为“-”，最后用 rjust( ) 函数实现了字符串调整，程序运行结果如图 1-4-7 所示。由结果可以看到，如果不提供分隔符，系统会默认将空格作为分隔符。

```
['My', 'major', 'is', 'big', 'data']
['www', 'hnpi', 'cn']
www-hnpi-cn
          www.hnpi.cn

Process finished with exit code 0
```

图 1-4-7　字符串分隔、拼接与对齐

### 7. 字符串的判断函数

在 Python 编程中，经常要将字符串进行转换，那么如何判断字符串是否符合转换的要求？ Python 内置了字符串函数进行格式的判断，如表 1-4-2 所示。

表 1-4-2　常用的字符串判断函数

| 函　数　名 | 函数功能及说明 |
| --- | --- |
| islower( ) | 如果字符串中的字母都是小写，则返回 True，否则返回 False |
| isupper( ) | 如果字符串中的字母都是大写，则返回 True，否则返回 False |
| isalpha( ) | 如果字符串中的所有字符都是字母，则返回 True，否则返回 False |
| isalnum( ) | 如果字符串只包含数字或字母，则返回 True，否则返回 False |
| isdigit( ) | 如果字符串只包含数字，则返回 True，否则返回 False |
| istitle( ) | 如果字符串中所有单词的首字母均为大写且其他字母均为小写，则返回 True，否则返回 False |
| startswith( ) | 用于检查字符串是否以指定字符串开头 |
| endswith( ) | 检查字符串是否以指定字符串结束，若是，则返回 True，否则返回 False |

【例 1-4-8】 字符串内容判断。

```
s = "abc123456"
print(s.islower( ))
print(s.isupper( ))
print(s.isalpha( ))
print(s.isalnum( ))
print(s.isdigit( ))
print(s.isnumeric( ))
```

```
print(s.startswith("ab"))
print(s.endswith("567"))
```

程序运行结果如图 1-4-8 所示。

```
True
False
False
True
False
False
True
False

Process finished with exit code 0
```

图 1-4-8　字符串内容判断函数运行结果

## 任务实现

宣传栏内容包括校训、欢迎语及单位和日期。

```
xiaoxun = '校训：“厚德笃学，知行合一”'
xuexiao = '河南工业职业技术学院欢迎你！ '
danwei = '电子信息工程学院'
date = '2023 年 9 月 1 日'
print(xiaoxun)
print( )
print(xuexiao.center(30))
print( )
print(danwei.rjust(30))
print(date.rjust(32))
```

程序运行结果如图 1-4-9 所示。

```
校训:“厚德笃学，知行合一”

        河南工业职业技术学院欢迎你!

                          电子信息工程学院
                            2023年9月1日

Process finished with exit code 0
```

图 1-4-9　输出宣传栏内容

# 课后拓展提升

## 赛题挑战

2020 年春节期间，有一个特殊的日期引起了大家的注意：2020 年 2 月 2 日。因为如果将这个日期按“yyyymmdd”的格式写成一个 8 位数是 20200202，恰好是一个回文数。人们称这样的日期为回文日期。

有人表示 20200202 是“千年一遇”的特殊日子。对此小明很不认同，因为不到 2 年之后就是下一个回文日期：20211202 即 2021 年 12 月 2 日。

也有人表示 20200202 并不仅仅是一个回文日期，还是一个 ABABBABA 型的回文日期。对此小明也不认同，因为大约 100 年后就能遇到下一个 ABABBABA 型的回文日期：21211212 即 2121 年 12 月 12 日。算不上“千年一遇”，顶多算“千年两遇”。

给定一个 8 位数的日期，计算该日期之后下一个回文日期和下一个 ABABBABA 型的回文日期各是哪一天。

输入包含一个 8 位整数 N，表示日期。

输出两行，每行 1 个 8 位数。第一行表示下一个回文日期，第二行表示下一个 ABABBABA 型的回文日期。

样例输入如下。

```
20200202
```

样例输出如下。

```
20211202
21211212
```

## 创新拓展

中国目前实施的是 18 位身份证号码制度，其中，身份证号码的第 7 至第 10 位数字代表出生年份，第 11 至第 12 位数字表示出生月份，第 13 至第 14 位数字表示出生日期。第 17 位数字用于区分性别，奇数代表男性，偶数代表女性。基于以上规则，请利用字符串处理的知识编写一个小程序，能够接收身份证号码作为输入，并输出用户的性别、年龄和生日信息。

## 项目实现

师父：小青，你已经完成了 Python 变量定义、字符串、运算符知识点的学习，还记得师父布置的任务吗？

河小青：当然记得了师父，您给我的第一个任务是实现入校信息登记，我已经能够独立实现了，我设计了学号、性别、所在学院、出生日期、所在班级等八项信息。编程的思路如下。

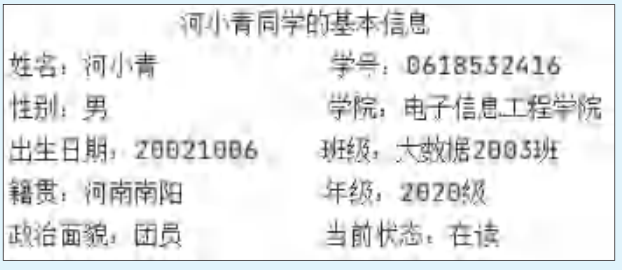
（1）先使用 print() 函数输出一句提示语“开始输入河小青同学的基本信息”。

（2）再使用 input() 函数接收姓名信息的输入，并存放在变量 name 中。

（3）依次实现学号、性别、所在学院、出生日期、所在班级等基本信息的输入。

（4）信息输入完毕后，再使用 print() 函数打印输出学生的基本信息。

（5）为了美观起见，我还使用字符串内容对齐函数 ljust() 控制字符串中的内容左对齐，使其占用 15 个字符的宽度。项目实现的结果如图 1-4-10 所示。请师父扫码查看并指导。




登记入学信息

图 1-4-10　项目实现结果

师父：不错，思路很清晰，结果呈现也挺好，通过本项目 4 个任务的学习，你已经了解了 Python 的基本编程规范，认识了 Python 中的数据类型，学会了变量、常量和字符串的操作方法。通过编写代码完成以上任务，可以看出你已经掌握了前面所学的知识点。完成入学信息登记后，你将开始全新的大学生活，这里也是你梦想启航的地方。希望在以后的生活中你能够坚持不懈，努力学习，使自己成为一个对社会、对国家有用的栋梁之材。

河小青：谢谢师父鼓励。

## 测试题

### 一、单选题

1. 下列（　　）选项不符合 Python 语言变量命名规则。

A. iSum　　B. teacher　　C. 5_abc　　D. _QE

2. 关于 Python 的复数类型，下列选项中描述错误的是（　　）。

A. 复数的虚数部分通过后缀 J 或 j 来表示

B. 复数的虚数部分只能通过后缀 J 来表示

C. 对于复数 z，可以用 z.real 得到它的实数部分

D. 对于复数 z，可以用 z.imag 得到它的虚数部分

3. 假设 a = 2，b = 1，下列（　　）的运算结果为 False。

A. a − b != 1 | a − 1 & b − 1　　B. b − 1

C. str(b − 1)　　D. 0b010 − 2

4. 下列说法错误的是（　　）。

A. 一个汉字作为一个字符计算　　B. +、−、* 也可作为字符运算符

C. 乘法运算符号是 *　　D. 余数的运算符号是 /

### 二、填空题

1. 单行注释的符号是 _____ 。

2. Python 表达式 3 ** 2 ** 2 的值为 _____ 。

3. 执行以下程序。

```
x = True
y = False
z = False
    print(x or y and z)
```

输出结果为 _____ 。

4. 假设 x = y = z = 6，x %= y + z，则 x = _____ 。

5. 假设 str1 = "Python 排版编辑 "，则 str1[:6] = _____ ，str1[8:10] = _____ 。

### 三、编程题

1. 输入两个数字存入变量 a、b 中，交换 a、b 的值。

2. 编写程序，格式化输出 298 的二进制、八进制、十六进制表示形式，以及对应的 Unicode 字符。

3. 编写程序，利用 input 函数输入任意一个 3 位整数，要求分别输出此整数的百位、十位和个位上的数字。

## 项目评价反馈

| 项目名称 | 学生成长档案——初来乍到，请多关照 | | | |
|---|---|---|---|---|
| 班级 | | 姓名 | | |
| 评价类型 | 考核内容 | 分值 | 得分 | 评价主体 |
| 过程评价 | 课前预习单完成情况 | 10 | | 师父 |
| | 项目资源浏览情况 | 10 | | 教师 |
| | 教学活动参与情况 | 10 | | 教师 |
| | 随堂测验情况 | 10 | | 教师 |
| | 作业提交情况 | 10 | | 教师 |
| | 小组任务完成情况 | 10 | | 组长 |
| 项目评价 | 项目功能实现 | 10 | | 教师 |
| | 代码编写规范 | 10 | | 师父 |
| | 项目创新拓展 | 10 | | 教师、学生 |
| 增值评价 | 学习效率提升 | 3 | | 学生 |
| | 学习兴趣提升 | 3 | | 学生 |
| | 小组工作贡献 | 4 | | 学生 |
| 合计 | | 100 | | |
| 综合得分 | | | | |

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　黄燕美

# Python
## 程序设计项目化教程

Python CHENGXU SHEJI XIANGMUHUA JIAOCHENG

主编　任越美　李　垒　李江岱



扫描二维码
关注上海交通大学出版社
官方微信

ISBN 978-7-313-30507-7
9 787313 305077 >
定价：59.80元