



# 《Python程序设计基础教程（微课版）》

<http://dblabb.xmu.edu.cn/post/python>



## 第1章 Python语言概述



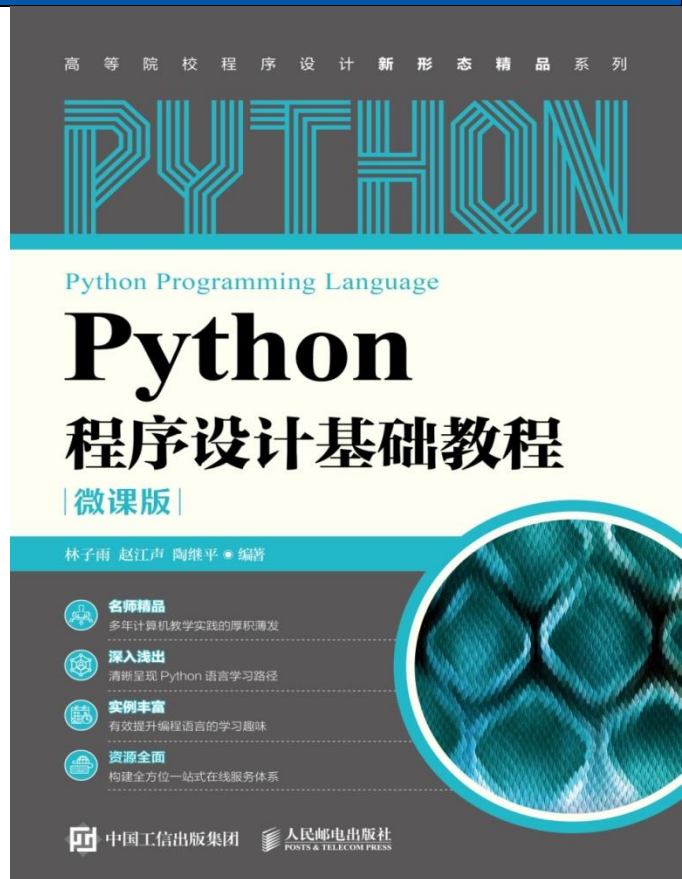


# 提纲

- 1.1 计算机语言
- 1.2 Python简介
- 1.3 搭建Python开发环境
- 1.4 Python规范

本PPT是如下教材的配套讲义：  
《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社  
《Python程序设计基础教程（微课版）》教材官方网站：  
<http://dblab.xmu.edu.cn/post/python>





# 1.1 计算机语言

1.1.1 计算机语言的种类

1.1.2 编程语言的选择



# 1.1.1 计算机语言的种类

1. 机器语言
2. 汇编语言
3. 高级语言



# 1.1.1 计算机语言的种类

## 1. 机器语言

- 机器语言是最低级的语言，是用二进制代码表示的计算机能直接识别和执行的一种机器指令的集合。
- 它是计算机的设计者通过计算机的硬件结构赋予计算机的操作功能。
- 机器语言具有灵活、直接执行和速度快等特点。
- 不同的计算机都有各自的机器语言，不同型号的计算机的机器语言是不相通的，按照一种计算机的机器指令编制的程序，不能在另一种计算机上执行。
- 在计算机发展的早期阶段，程序员使用机器语言来编写程序，编出的程序全是由0和1构成的指令代码，可读性差，还容易出错。
- 计算机语言发展到今天，除了计算机生产厂家的专业人员外，绝大多数的程序员已经不再去学习机器语言了。



# 1.1.1 计算机语言的种类

## 2. 汇编语言

- 汇编语言是用于电子计算机、微处理器、微控制器或其他可编程器件的低级语言，亦称为“符号语言”。
- 在汇编语言中，用助记符代替机器指令的操作码，用地址符号或标号代替指令或操作数的地址，从而增强了程序的可读性，并降低了编程难度。
- 使用汇编语言编写的程序，不能直接被机器识别，还要由汇编程序（或者叫“汇编语言编译器”）转换成机器指令。
- 汇编语言的目标代码简短，占用内存少，执行速度快，是高效的程序设计语言，到现在依然是常用的编程语言之一。
- 但是，汇编语言只是将机器语言做了简单编译，并没有根本上解决机器语言的特定性，所以，汇编语言和机器自身的编程环境是息息相关的，推广和移植比较困难。



# 1.1.1 计算机语言的种类

## 3.高级语言

- 由于汇编语言依赖于硬件体系，且助记符量大难记，于是人们又发明了更加简单易用的高级语言。
  - 和汇编语言相比，高级语言不但将许多相关的机器指令合成为单条指令，并且去掉了与具体操作有关但与完成工作无关的细节，例如使用堆栈、寄存器等，这样就大大简化了程序中的指令。同时，由于省略了很多细节，编程者也就不需要有太多的专业知识，经过一定的学习之后都可以编程。
  - 但是，高级语言生成的程序代码一般比用汇编语言设计的程序代码要长，执行的速度也慢。
- 
- 高级语言主要是相对于低级语言而言，它并不是特指某一种具体的语言，而是包括了很多种编程语言，如流行的Java、C、C++、C#、Pascal、Python、Scala、PHP等，这些语言的语法、命令格式都各不相同。



# 1.1.1 计算机语言的种类

## 3. 高级语言

- 高级语言编写的程序不能直接被计算机识别，必须经过转换才能被执行，按转换方式可将它们分为两类：解释类和编译类。
- 对于解释类的高级语言而言（比如**Python**），应用程序源代码一边由相应语言的解释器“翻译”成目标代码（机器语言），一边执行，因此，效率相对较低，而且不能生成可独立执行的可执行文件，应用程序不能脱离其解释器，但这种方式比较灵活，可以动态地调整、修改应用程序。
- 对于编译类的高级语言而言（比如**Java**），在应用程序源代码执行之前，首先需要将源代码“翻译”成目标代码（机器语言），因此，其目标程序可以脱离其语言环境独立执行，使用比较方便、效率较高。但是，应用程序一旦需要修改，必须先修改源代码，再重新编译生成新的目标文件才能执行。





## 1.1.2 编程语言的选择

- 随着信息技术的发展，计算机程序设计课程在高校中成为一门必修的基础课程，对于这门课程而言，关键是要选择一种合适的编程语言（高级语言）。
- 以前，大部分高校采用的编程语言往往是C、C++或Java，但是，最近几年，Python凭借其独特的优势逐渐崭露头角。
- Python语言是一种解释型、面向对象的计算机程序设计语言，广泛用于计算机程序设计教学、系统管理编程、科学计算等，特别适用于快速的应用程序开发。
- 目前，各大高校越来越重视Python教学，Python已经成为最受欢迎的程序设计语言之一。与传统编程语言的复杂开发过程不同，Python语言在操作上非常方便、快捷，学习者容易掌握，可以提升编程效率，并增强其学习信心。



## 1.1.2 编程语言的选择

具体而言，Python的主要优势如下：

- （1）学习入门容易。与C、C++、Java语言相比，Python语言在编写代码时不需要建立main函数，无论是书写还是掌握计算机算法都比较简单。而且，Python语言中没有大量的语法知识，只要在理解的基础上掌握部分环节即可，能实现教学资源的合理配置。
- （2）功能强大。当使用Python语言编写程序的时候，我们不需要考虑如何管理程序使用的内存之类的细节。并且，Python有很丰富的库，其中既有官方开发的，也有第三方开发的，很多功能模块都已经写好了，我们只需要调用即可，不需要重新发明“轮子”。
- （3）应用领域非常广泛。Python语言可以应用到网站后端开发、自动化运维、数据分析、游戏开发、自动化测试、网络爬虫、智能硬件开发等各个领域。



## 1.1.2 编程语言的选择

- 此外，类似Python的编程语言也是高级语言发展的必然选择。
- 从程序设计语言发展角度来看，高级语言的设计一直在追求接近人类的自然语言。
- C、Java、VB等都在朝着这个方向努力，而Python语言则是更进了一步，它提供了十分接近人类理解的语法形式。
- 应该说，Python语言优化了高级语言的表达形式，简化了程序设计过程，提升了程序设计效率。
- 从计算思维培养角度来说，传统的C、Java和VB等语言过分强调语法，并不适合非计算机专业的学生。
- 从传统应用技能教育向计算思维培养转变过程中，教学内容变革是重中之重。
- 对于程序设计课程，选择适合时代和技术发展的编程语言，是显著提高培养效果的前提和基础。
- 从解决计算问题的角度，传统的C、Java和VB语言过分强调语法，而Python语言作为“轻语法”程序设计语言，相比其他语言而言具有更高的教学价值。



# 1.2 Python简介

1.2.1 什么是Python

1.2.2 Python语言的特点

1.2.3 Python语言的应用

1.2.4 Python的版本

本PPT是如下教材的配套讲义：

《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：

<http://dblab.xmu.edu.cn/post/python>





## 1.2.1 什么是Python

Python（发音['paɪθən]）是1989年由荷兰人Guido van Rossum发明的一种面向对象的解释型高级编程语言，它的标志如图1-1所示。Python的第一个公开发行人版发行于1991年，自从2004年以后，Python的使用率呈线性增长。TIOBE在2019年1月发布的排行榜显示，Python获得“TIOBE最佳年度语言”称号，这是Python第3次获得“TIOBE最佳年度语言”，也是获奖次数最多的编程语言。发展到今天，Python已经成为最受欢迎的程序设计语言之一。



图1-1 Python的标志



## 1.2.1 什么是Python

Python常被称为“胶水语言”，能够把用其它语言制作的各種模块（尤其是C/C++）很轻松地连接在一起。常见的一种应用情形是，使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，性能要求特别高，就可以用C/C++重写，而后封装为Python可以调用的扩展类库。Python的设计哲学是“优雅”、“明确”、“简单”。在设计Python语言时，如果面临多种选择，Python开发者一般会拒绝花哨的语法，而选择明确地没有或者很少有歧义的语法。总体来说，选择Python开发程序具有简单、开发速度快、节省时间和精力等特点，因此，在Python开发领域流传着这样一句话：“人生苦短，我用Python”。



## 1.2.2 Python语言的特点

1. Python语言的优点
2. Python语言的缺点



## 1.2.2 Python语言的特点

### 1. Python语言的优点

#### (1) 语言简单。

**Python**是一门语法简单且风格简约易读的语言。它注重的是如何解决问题，而不是编程语言本身的语法和结构。**Python**语言丢掉了分号以及花括号这些仪式化的东西，使得语法结构尽可能地简洁，代码的可读性显著提高。

相较于**C**、**C++**、**Java**等编程语言，**Python**语言提高了开发者的开发效率，削减了原来**C**语言、**C++**以及**Java**语言中一些较为复杂的语法，降低了编程工作的复杂程度，实现同样的功能时，**Python**语言所包含的代码量是最少的，代码行数是其他语言的1/5到1/3。





## 1.2.2 Python语言的特点

### 1. Python语言的优点

#### (2) 开源、免费。

开源，即开放源代码，也就是所有用户都可以看到源代码。Python的开源体现在两方面：首先，程序员使用 Python 编写的代码是开源的；其次，Python 解释器和模块是开源的。

开源并不等于免费，开源软件和免费软件是两个概念，只不过大多数的开源软件也是免费软件。Python 就是这样一种语言，它既开源又免费。用户使用 Python 进行开发或者发布自己的程序，不需要支付任何费用，也不用担心版权问题，即使作为商业用途，Python也是免费的。



## 1.2.2 Python语言的特点

### 1. Python语言的优点

#### (3) 面向对象。

面向对象的程序设计，更加接近人类的思维方式，是对现实世界中客观实体进行结构和行为模拟。**Python**语言完全支持面向对象编程，比如支持继承、重载运算符、派生以及多继承等。与**C++**与**Java**相比，**Python**以一种非常强大而简单的方式实现面向对象编程。

需要说明的是，**Python**在支持面向对象编程的同时，也支持面向过程的编程，也就是说，它不强制使用面向对象编程，这使得其编程更加灵活。在“面向过程”的编程中，程序是由过程或仅仅是可重用代码的函数构建起来的。在“面向对象”的编程中，程序是由数据和功能组合而成的对象构建起来的。



## 1.2.2 Python语言的特点

### 1. Python语言的优点

#### （4）跨平台。

由于Python是开源的，它已经被移植到许多平台上。如果能够避免使用那些需要依赖于系统的特性，那就意味着，所有Python程序都无需修改就可以在好多平台上运行，包括Linux、Windows、FreeBSD、Solaris等等，甚至还有PocketPC、Symbian以及Google基于Linux开发的Android平台。解释型语言几乎天生就是跨平台的。Python作为一门解释型的语言，它天生具有跨平台的特征，只要为平台提供了相应的Python解释器，Python就可以在该平台上运行。



## 1.2.2 Python语言的特点

### 1. Python语言的优点

#### （5）强大的生态系统。

在实际应用中，**Python**语言的用户群体，绝大多数并非专业的开发者，而是其他领域的爱好者。对于这一部分用户来说，他们学习**Python**语言的目的不是去做专业的程序开发，而仅仅是使用现成的类库去解决实际工作中的问题。**Python**极其庞大的生态，刚好能够满足这些用户的需求。这在整个计算机语言发展史上都是开天辟地的，也是**Python**语言在各个领域盛行的原因。丰富的生态系统也给专业开发者带来了极大的便利。大量成熟的第三方库可以直接使用，专业开发者只需要使用很少的语法结构就可以编写出功能强大的代码，缩短了开发周期，提高了开发效率。常用的**Python**第三方库包括 **Matplotlib**（数据可视化库）、**NumPy**（数值计算功能库）、**SciPy**（数学、科学、工程计算功能库）、**pandas**（数据分析高层次应用库）、**Scrapy**（网络爬虫功能库）、**BeautifulSoup**（HTML和XML的解析库）、**Django**（Web应用框架）、**Flask**（Web应用微框架）等。



## 1.2.2 Python语言的特点

### 2. Python语言的缺点

#### (1) 速度慢。

运行速度慢是解释型语言的通病，Python也不例外。由于Python是解释型语言，所以，它的速度会比C、C++、Java稍微慢一些。但是，由于现在的硬件配置都非常高，硬件性能的提升可以弥补软件性能的不足，所以，运行速度慢这一点对于使用Python开发应用程序基本上没有影响，除非是一些实时性比较强的程序可能会受到一些影响，但是也有解决办法，比如可以嵌入C程序。

#### (2) 存在多线程性能瓶颈。

Python中存在全局解释器锁（Global Interpreter Lock），它是一个互斥锁，只允许一个线程来控制Python解释器。当Python的默认解释器要执行字节码时，都需要先申请这个锁。

这意味着在任何时间点，只有一个线程可以处于执行状态。执行单线程程序的开发人员感受不到全局解释器锁的影响，但它却成为多线程代码中的性能瓶颈。



## 1.2.2 Python语言的特点

### 2. Python语言的缺点

#### (3) 代码不能加密。

我们在发布Python程序时，实际上就是发布源代码。这一点跟C语言不同，C语言不用发布源代码，只需要把编译后的机器码（也就是在Windows上常见的exe文件）发布出去。要从机器码反推出C代码是不可能的，所以，凡是编译型的语言，都没有这个问题。而对于Python这样的解释型语言，则必须把源码发布出去。

#### (4) Python2.x和Python3.x不兼容。

如果一个普通的软件或者库不能够做到向后兼容，通常会被用户抛弃。在Python中，一个饱受诟病的地方就是Python2.x和Python3.x不兼容，这给Python开发人员带来了无数烦恼。



## 1.2.3 Python语言的应用

### 1.数据科学

Python被广泛应用于数据科学领域。在数据采集环节，在Python的第三方库Scrapy的支持下，可以编写网络爬虫程序采集网页数据。在数据清洗环节，第三方库pandas提供了功能强大的类库，可以帮助我们清洗数据、排序数据，最后得到清晰明了的数据。在数据处理分析环节，第三方库NumPy和SciPy提供了丰富的科学计算和数据分析功能，包括统计、优化、整合、线性代数模块、傅里叶变换、信号和图像图例、常微分方程求解、矩阵解析和概率分布等。在数据可视化环节，第三方库Matplotlib提供了丰富的数据可视化图表。



## 1.2.3 Python语言的应用

### 2.人工智能

虽然可以使用各种不同的编程语言开发人工智能程序，但是，Python语言在人工智能领域具有独特的优势。在人工智能领域，有许多基于Python语言的第三方库，比如，**scikit-learn**、**Keras**和**NLTK**等。其中，**scikit-learn**是基于Python语言的机器学习工具，提供了简单高效的数据挖掘和数据分析功能；**Keras**是一个基于Python的深度学习库，提供了用Python编写的高级神经网络API；**NLTK**是Python自然语言工具包，用于诸如标记化、词形还原、词干化、解析、POS标注等任务。此外，深度学习框架**Tensorflow**、**Caffe**等，主体都是用Python实现的，提供的原生接口也是面向Python的。





## 1.2.3 Python语言的应用

### 3. 网站开发

在网站开发方面，Python具有Django、Flask、Pyramid、Bottle、Tornado、web2py等框架，使用Python开发的网站具有小而精的特点。知乎、豆瓣、美团、饿了么等网站都是使用Python搭建的。这一方面说明了Python作为网站开发的受欢迎程度，另一方面也说明Python语言用作网站开发经受住了大规模用户并发访问的考验。



## 1.2.3 Python语言的应用

### 4. 系统管理

Python简单易用、语法优美，特别适合系统管理的应用场景。著名的开源云计算平台OpenStack就是使用Python语言开发的。除此之外，Python生态中还有Ansible、Salt等自动化部署工具，也是使用Python语言开发的。这么多使用广泛、功能强大的系统管理工具，都是使用Python语言开发，也反映了Python语言非常适合系统管理的事实。



## 1.2.3 Python语言的应用

### 5.网络爬虫

网络爬虫是一个自动提取网页的程序，它为搜索引擎从万维网上下载网页，是搜索引擎的重要组成部分。**Scrapy**就是用**Python**实现的爬虫框架，用户只需要定制开发几个模块就可以轻松地实现一个爬虫，用来抓取网页内容或者各种图片。



## 1.2.4 Python的版本

### 1. Python2.x和Python3.x的区别

Python官方网站目前同时发行Python2.x和Python3.x两个不同系列的版本，并且彼此之间不兼容，除了输入输出方式有所不同，很多内置函数的实现和使用方式也有较大差别。总体而言，在语法层面，二者的主要区别表现在以下几个方面：

- （1）在Python2.x中，`print`语句被Python3.x中的`print()`函数所代替；
- （2）在Python3.x中，整数之间的相除（采用除法运算符“/”实现），结果是浮点数，而在Python2.x中结果是整数；
- （3）Python3.x源码文件默认使用UTF-8编码，所以支持直接写入中文，而Python2.x默认编码是ASCII，直接写入中文会被转换成ANSI编码。所以在Python2.x中需要进行相应的转换；
- （4）在Python3.x中将`range()`与`xrange()`函数整合为一个`range()`函数，所以在Python3.x中不存在`xrange()`函数，而在Python2.x中这两个函数是并存的。



## 1.2.4 Python的版本

### 2.使用Python2.x还是Python3.x

Python2.x与Python3.x不兼容就会导致一个问题，对于一个编程的初学者而言，应该学习Python2.x还是学习Python3.x呢？这里建议直接学习Python3.x版本（比如，本教程使用Python3.8.7版本），理由如下：

（1）Python2.x和Python3.x的思想是共通的。实际上，编程重在对编程思想的理解和经验的积累，不同的编程语言，它们的很多思想都是共通的。Python2.x和Python3.x属于同一种编程语言，更是如此，即Python2.x和Python3.x在编程思想上基本是共通的。Python2.x和Python3.x的语法虽然存在不兼容的情况，但是，也只是一小部分语法不兼容。

（2）使用Python3.x是大势所趋。从总体趋势而言，会有越来越多的开发者选择Python3.x，放弃Python2.x。此外，围绕Python3.x的第三方库也会逐渐丰富起来，这也会让更多开发者投入Python3.x的怀抱。



# 1.3 搭建Python开发环境

- 1.3.1 安装Python
- 1.3.2 使用交互式执行环境
- 1.3.3 运行代码文件
- 1.3.4 使用IDLE编写代码
- 1.3.5 第三方开发工具

本PPT是如下教材的配套讲义：  
《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著, 人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：  
<http://dblab.xmu.edu.cn/post/python>





## 1.3.1 安装Python

Python可以用于多种平台，包括Windows、Linux和Mac OS等。本教程采用的操作系统是Windows7或以上版本，使用的Python版本是3.8.7。请到Python官网（<https://www.python.org/>）下载与自己计算机操作系统匹配的安装包，比如，64位Windows操作系统可以下载python-3.8.7-amd64.exe。在安装过程中，要注意选中“Add python 3.8 to PATH”（如图1-2所示），这样可以在安装过程中自动配置PATH环境变量，避免了手动配置的繁琐过程。

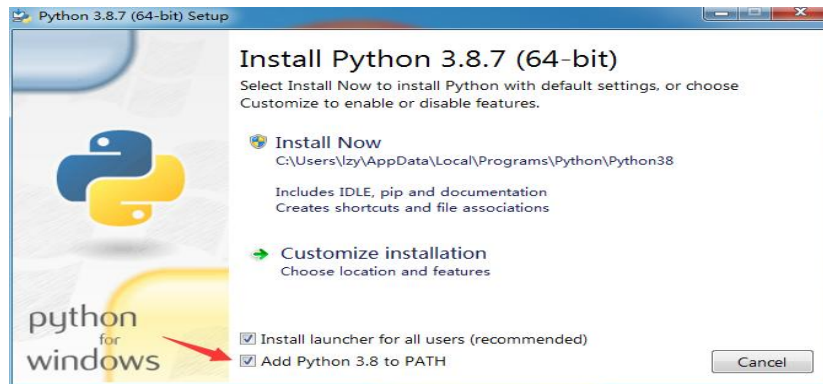
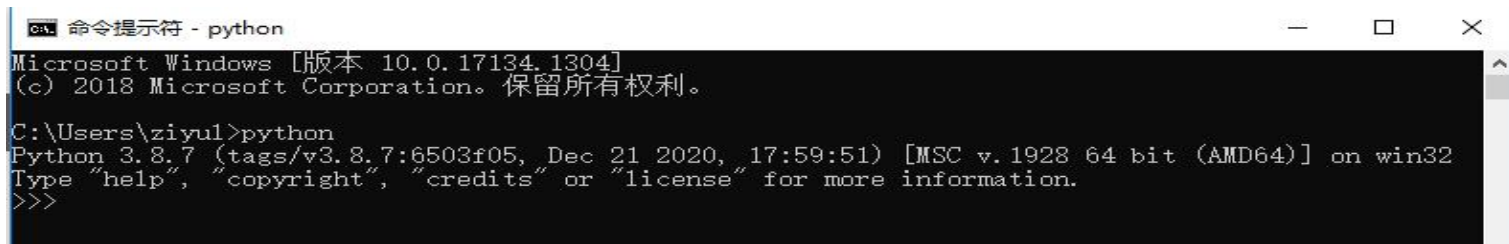


图1-2 Python安装界面



## 1.3.1 安装Python

安装成功以后，需要检测是否安装成功。可以打开Windows系统的cmd命令界面，并在命令提示符后面输入“python”后回车，如果出现如图1-3所示信息，则说明Python已经安装成功。



```
命令提示符 - python
Microsoft Windows [版本 10.0.17134.1304]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\ziyul>python
Python 3.8.7 (tags/v3.8.7:6503f05, Dec 21 2020, 17:59:51) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

图1-3 Python命令行界面





## 1.3.2 使用交互式执行环境

图1-3呈现的界面就是一个交互式执行环境（或称为“解释器”），可以在Python命令提示符“>>>”后面输入各种Python代码，按回车后就会立即看到执行结果，比如：

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

```
Hello World
```

```
>>> 1+2
```

```
3
```

```
>>> 2*(3+4)
```

```
14
```



## 1.3.3 运行代码文件

假设在Windows系统的C盘根目录下已经存在一个代码文件hello.py，该文件里面只有如下一行代码：

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

现在我们要运行这个代码文件。可以打开Windows系统的cmd命令界面，并在命令提示符后面输入如下语句：

```
$ python C:\hello.py
```

运行结果如图1-4所示。



```
命令提示符
Microsoft Windows [版本 10.0.17134.1304]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\ziyul>python C:\hello.py
Hello World
```

图1-4 在cmd命令界面中执行Python代码文件



## 1.3.4使用IDLE编写代码

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

在Windows系统的“开始”菜单中找到“IDLE(Python 3.8 64-bit)”，打开进入IDLE主窗口（如图1-5所示），窗口左侧会显示Python命令提示符“>>>”，可以在提示符后面输入Python代码，回车后就会立即执行并返回结果。

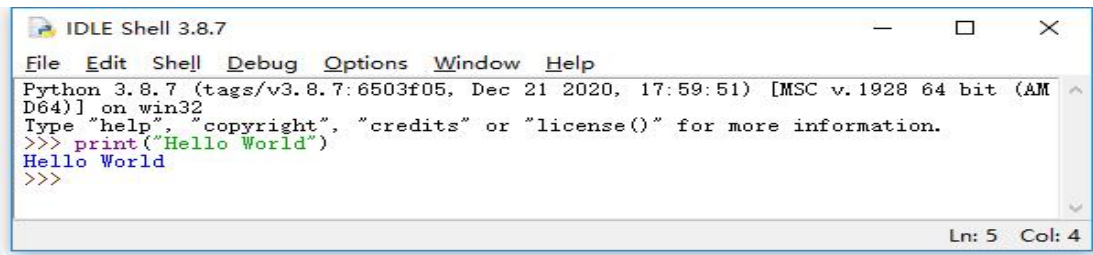


图1-5 IDLE主窗口



## 1.3.4使用IDLE编写代码

如果要创建一个代码文件，可以在IDLE主窗口的顶部菜单栏中选择“**File->New File**”，然后就会弹出如图1-6所示的文件窗口，可以在里面输入Python代码，然后，在顶部菜单栏中选择“**File->Save As...**”，把文件保存为hello.py。

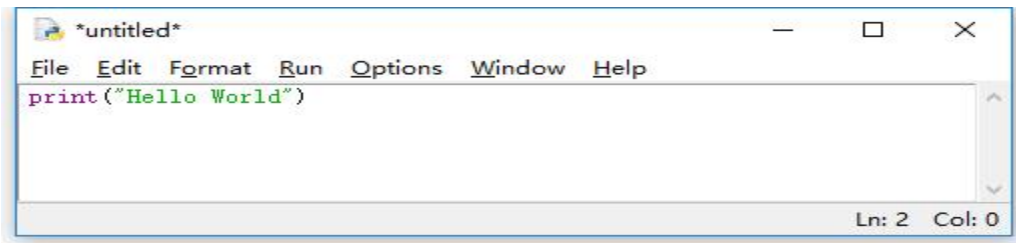
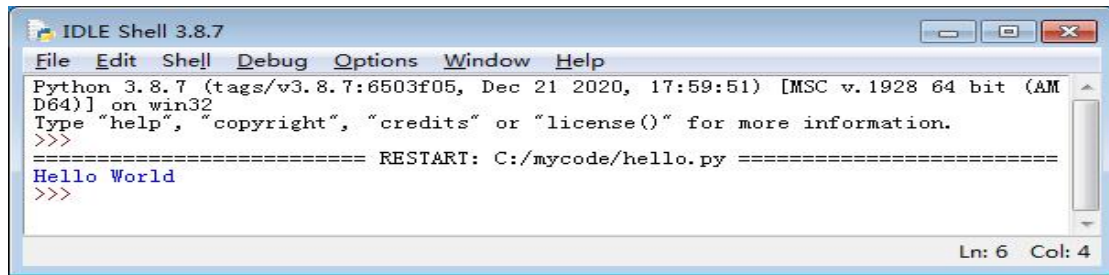


图1-6 IDLE的文件窗口



## 1.3.4使用IDLE编写代码

如果要运行代码文件hello.py，可以在IDLE的文件窗口的顶部菜单栏中，选择“Run->Run Module”（或者直接使用快捷键“F5”），这时就会开始运行程序。程序运行结束后，会在IDLE Shell窗口显示执行结果（如图1-7所示）。



```
Python 3.8.7 (tags/v3.8.7:6503f05, Dec 21 2020, 17:59:51) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/mycode/hello.py =====
Hello World
>>>
```

图1-7 程序执行结果



## 1.3.4使用IDLE编写代码

在实际开发中，可以通过使用IDLE提供的快捷键（如表1-1所示）来提高程序开发效率。

表1-1 IDLE常用快捷键

快捷键	功能说明
F1	打开Python帮助文档
F5	运行Python代码文件
Ctrl+]	缩进代码块
Ctrl+[	取消代码块缩进
Ctrl+F6	重新启动IDLE Shell
Ctrl+Z	撤销一步操作
Ctrl+Shift+Z	恢复上一次的撤销操作
Ctrl+S	保存文件
Alt+P	浏览历史命令（上一条）
Alt+N	浏览历史命令（下一条）
Alt+/	自动补全前面曾经出现过的单词，如果之前有多个单词具有相同前缀，可以连续按该快捷键，在多个单词中循环选择
Alt+3	注释代码块
Alt+4	取消代码块注释
Alt+g	转到某一行



## 1.3.5 第三方开发工具

除了Python自带的IDLE以外，我们还可以选择其他第三方开发工具进行Python编程，主要包括：

(1) **PyCharm**。PyCharm是一款功能强大的Python编辑器，具有跨平台性，可以应用在Windows、Linux和Mac OS系统中。PyCharm拥有一般的集成式开发环境应该具备的功能，比如调试、语法高亮、项目管理、代码跳转、智能提示、自动完成、单元测试、版本控制等。另外，PyCharm还提供了一些很好的功能用于Django开发，而且还支持Google App Engine。

(2) **Eclipse**。Eclipse是著名的、跨平台的自由集成式开发环境。最初主要用于Java语言开发，但是通过安装插件，可以将其作为其他计算机语言（比如C++和Python）的开发工具。如果要使用Eclipse进行Python开发，则需要安装插件PyDev。

(3) **Jupyter Notebook**。最初只支持Python开发，后来发展到可以支持其他40多种编程语言，目前已经成为用Python做教学、计算、科研的一个重要工具。



# 1.4 Python规范

## 1.4.1 注释规则

## 1.4.2 代码缩进

本PPT是如下教材的配套讲义：

《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：

<http://dbl原因.xmu.edu.cn/post/python>







## 1.4.1 注释规则

### 1.单行注释

Python中使用“#”表示单行注释。单行注释可以作为单独的一行放在被注释代码行之上，或者也可以放在语句或表达式之后。

【例1-1】单行注释作为单独的一行放在被注释代码行之上。

```
pi = 3.14  
r = 2  
# 使用面积公式求出圆的面积  
area = pi*r*r  
print(area)
```

当单行注释作为单独的一行放在被注释代码行之上时，为了保证代码的可读性，建议在“#”后面添加一个空格，再添加注释内容。



## 1.4.1 注释规则

【例1-2】单行注释放在语句或表达式之后。

```
length = 3 #矩形的长  
width = 5 #矩形的宽  
area = length*width #求出矩形的面积  
print(area)
```

当单行注释放在语句或表达式之后时，同样为了保证代码的可读性，建议注释和语句（或注释和表达式）之间至少要有两个空格。



## 1.4.1 注释规则

### 2. 多行注释

当注释内容过多，导致一行无法显示时，就可以使用多行注释。Python 中使用三个单引号或三个双引号表示多行注释。

【例1-3】使用三个单引号的多行注释。

```
'''
```

```
文件名: area.py
```

```
用途: 用于求解矩形的面积
```

```
创建日期: 2021年7月1日
```

```
创建人: XMU
```

```
'''
```



## 1.4.1 注释规则

【例1-4】使用三个双引号的多行注释。

```
"""
```

```
文件名: area.py
```

```
用途: 用于求解矩形的面积
```

```
创建日期: 2021年7月1日
```

```
创建人: XMU
```

```
"""
```



## 1.4.1 注释规则

### 3. 中文注释

中文注释主要是为了解决Python2.x中不支持直接写中文的问题，虽然该问题在Python3.x中已经不存在，但是，为了方便他人了解代码文件所使用的编码，建议在文件开始位置加上中文注释。

**【例1-5】** 在文件开始位置添加中文注释。

```
01      # -*- coding:utf-8 -*-  
02      length = 3 #矩形的长  
03      width = 5 #矩形的宽  
04      area = length*width #求出矩形的面积  
05      print(area)
```



## 1.4.2 代码缩进

Python和其他编程语言（比如C和Java）很不一样的地方在于，Python采用代码缩进和冒号来区分代码之间的层次，而Java和C语言则采用大括号来分隔代码块。如果我们有其他语言（如C或者Java）的编程经验，那么Python的强制缩进一开始会让我们很不习惯。但是，如果习惯了Python的缩进语法，就会觉得它非常优雅。

缩进可以使用空格或者Tab键来实现（建议使用空格作为缩进）。当使用空格作为缩进时，建议使用4个空格作为一个缩进量。



## 1.4.2 代码缩进

【例1-6】Python的缩进语法。

```
length = 3 #矩形的长
width = 5 #矩形的宽
area = length*width #矩形的面积
if area > 10:
    print("大矩形")
```

常用的集成式开发环境（比如IDLE、PyCharm、Eclipse等）都具有自动缩进的机制，比如输入“:”号之后，按“回车”会自动进行缩进。

The background is a solid blue gradient. It features several faint, light-blue silhouettes of people. In the top left, a group of four people is holding hands in a circle. In the top center, a group of seven people is holding hands in a line. On the right side, a person is shown in profile, looking towards the center. In the bottom left, there are two silhouettes of people, one appearing to be looking at the other. The text "Thank You!" is centered in the middle of the image in a white, bold, sans-serif font.

**Thank You!**