01 第一个Python程序

Python交互模式

在命令行模式下敲命令`python`，就看到类似如下的一堆文本输出，然后就进入到Python交互模式，它的提示符是`>>>`。

```
┌──────────────────────────────────────────────────────────┐
|Command Prompt - python                          - □ x |
├──────────────────────────────────────────────────────────┤
|Microsoft Windows [Version 10.0.0]                       |
|(c) 2015 Microsoft Corporation. All rights reserved.     |
|                                                         |
|C:\> python                                              |
|Python 3.7 ... on win32                                  |
|Type "help", ... for more information.                   |
|>>> _                                                    |
|                                                         |
|                                                         |
|                                                         |
|                                                         |
└──────────────────────────────────────────────────────────┘
```

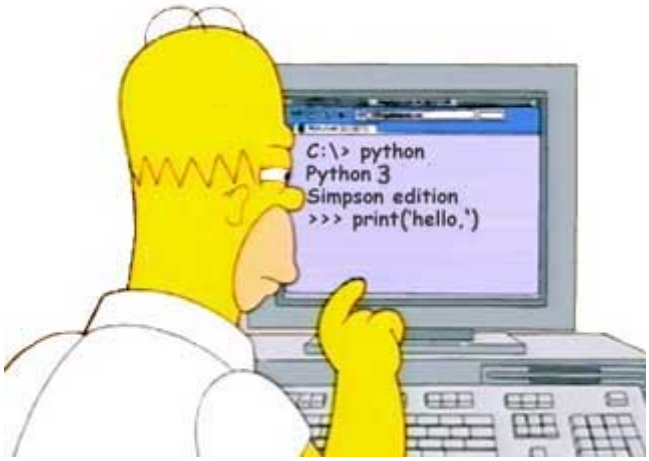
在Python交互模式下输入`exit()`并回车，就退出了Python交互模式，并回到命令行模式：

```
┌──────────────────────────────────────────────────────────┐
|Command Prompt                                   - □ x |
├──────────────────────────────────────────────────────────┤
|Microsoft Windows [Version 10.0.0]                       |
|(c) 2015 Microsoft Corporation. All rights reserved.     |
|                                                         |
|C:\> python                                              |
|Python 3.7 ... on win32                                  |
|Type "help", ... for more information.                   |
|>>> exit()                                               |
|                                                         |
|C:\> _                                                   |
|                                                         |
|                                                         |
└──────────────────────────────────────────────────────────┘
```

也可以直接通过开始菜单选择`Python (command line)`菜单项，*直接进入*Python交互模式，但是输入`exit()`后窗口会直接关闭，不会回到命令行模式。

了解了如何启动和退出Python的交互模式，我们就可以正式开始编写Python代码了。

在写代码之前，请*千万不要*用“复制”-“粘贴”把代码从页面粘贴到你自己的电脑上。写程序也讲究一个感觉，你需要一个字母一个字母地把代码自己敲进去，在敲代码的过程中，初学者经常会敲错代码：拼写不对，大小写不对，混用中英文标点，混用空格和Tab键，所以，你需要仔细地检查、对照，才能以最快的速度掌握如何写程序。



在交互模式的提示符`>>>`下，直接输入代码，按回车，就可以立刻得到代码执行结果。现在，试试输入`100+200`，看看计算结果是不是300：

```
>>> 100+200
300
```

很简单吧，任何有效的数学计算都可以算出来。

如果要让Python打印出指定的文字，可以用`print()`函数，然后把希望打印的文字用单引号或者双引号括起来，但不能混用单引号和双引号：

```
>>> print('hello, world')
hello, world
```

这种用单引号或者双引号括起来的文本在程序中叫字符串，今后我们还会经常遇到。

最后，用`exit()`退出Python，我们的第一个Python程序完成！唯一的缺憾是没有保存下来，下次运行时还要再输入一遍代码。

命令行模式和Python交互模式

请注意区分命令行模式和Python交互模式。

在命令行模式下，可以执行`python`进入Python交互式环境，也可以执行`python hello.py`运行一个`.py`文件。

执行一个`.py`文件*只能*在命令行模式执行。如果敲一个命令`python hello.py`，看到如下错误：

```
┌──────────────────────────────────────────────────────────┐
│Command Prompt                                  _ □ x     │
├──────────────────────────────────────────────────────────┤
│Microsoft Windows [Version 10.0.0]                        │
│(c) 2015 Microsoft Corporation. All rights reserved.      │
│                                                          │
│C:\> python hello.py                                      │
│python: can't open file 'hello.py': [Errno 2] No such     │
│file or directory                                         │
│                                                          │
│                                                          │
│                                                          │
│                                                          │
│                                                          │
└──────────────────────────────────────────────────────────┘
```

错误提示`No such file or directory`说明这个`hello.py`在当前目录*找不到*，必须先把当前目录切换到`hello.py`所在的目录下，才能正常执行：

```
┌──────────────────────────────────────────────────────────┐
│Command Prompt                                  _ □ x     │
├──────────────────────────────────────────────────────────┤
│Microsoft Windows [Version 10.0.0]                        │
│(c) 2015 Microsoft Corporation. All rights reserved.      │
│                                                          │
│C:\> cd work                                              │
│                                                          │
│C:\work> python hello.py                                  │
│Hello, world!                                             │
│                                                          │
│                                                          │
│                                                          │
│                                                          │
└──────────────────────────────────────────────────────────┘
```

此外，在命令行模式运行`.py`文件和在Python交互式环境下直接运行Python代码有所不同。Python交互式环境会把每一行Python代码的结果自动打印出来，但是，直接运行Python代码却不会。

例如，在Python交互式环境下，输入：

```
>>> 100 + 200 + 300
600
```

直接可以看到结果`600`。

但是，写一个`calc.py`的文件，内容如下：

```
100 + 200 + 300
```

然后在命令行模式下执行：

```
C:\work>python calc.py
```

发现什么输出都没有。

这是正常的。想要输出结果，必须自己用`print()`打印出来。把`calc.py`改造一下：

```
print(100 + 200 + 300)
```

再执行，就可以看到结果：

```
C:\work>python calc.py
600
```

最后，Python交互模式的代码是输入一行，执行一行，而命令行模式下直接运行`.py`文件是一次性执行该文件内的所有代码。可见，Python交互模式主要是为了调试Python代码用的，也便于初学者学习，它*不是*正式运行Python代码的环境！

使用文本编辑器

在Python的交互式命令行写程序，好处是一下就能得到结果，坏处是没法保存，下次还想运行的时候，还得再敲一遍。

所以，实际开发的时候，我们总是使用一个文本编辑器来写代码，写完了，保存为一个文件，这样，程序就可以反复运行了。

现在，我们就把上次的`'hello, world'`程序用文本编辑器写出来，保存下来。

那么问题来了：文本编辑器到底哪家强？

Visual Studio Code!

我们推荐微软出品的Visual Studio Code，它不是那个大块头的Visual Studio，它是一个精简版的迷你Visual Studio，并且，Visual Studio Code可以跨！平！台！Windows、Mac和Linux通用。

请注意，*不要用Word和Windows自带的记事本*。Word保存的不是纯文本文件，而记事本会自作聪明地在文件开始的地方加上几个特殊字符（UTF-8 BOM），结果会导致程序运行出现莫名其妙的错误。

安装好文本编辑器后，输入以下代码：

```
print('hello, world')
```

注意`print`前面不要有任何空格。然后，选择一个目录，例如`C:\work`，把文件保存为`hello.py`，就可以打开命令行窗口，把当前目录切换到`hello.py`所在目录，就可以运行这个程序了：

```
C:\work>python hello.py
hello, world
```

也可以保存为别的名字，比如`first.py`，但是必须要以`.py`结尾，其他的都不行。此外，文件名只能是英文字母、数字和下划线的组合。

如果当前目录下没有`hello.py`这个文件，运行`python hello.py`就会报错：

```
C:\Users\IEUser>python hello.py
python: can't open file 'hello.py': [Errno 2] No such file
or directory
```

报错的意思就是，无法打开`hello.py`这个文件，因为文件不存在。这个时候，就要检查一下当前目录下是否有这个文件了。如果`hello.py`存放在另外一个目录下，要首先用`cd`命令切换当前目录。

视频演示：

直接运行py文件

有同学问，能不能像.exe文件那样直接运行`.py`文件呢？在Windows上是不行的，但是，在Mac和Linux上是可以的，方法是在`.py`文件的第一行加上一个特殊的注释：

```
#!/usr/bin/env python3

print('hello, world')
* ```

然后，通过命令给`hello.py`以执行权限：
```

$ chmod a+x hello.py

就可以直接运行`hello.py`了，比如在Mac下运行：

![run-python-in-shell]
(https://www.liaoxuefeng.com/files/attachments/923626376177344/0)

小结

用文本编辑器写Python程序，然后保存为后缀为`.py`的文件，就可以用Python直接运行这个程序了。

Python的交互模式和直接运行`.py`文件有什么区别呢？

直接输入`python`进入交互模式，相当于启动了Python解释器，但是等待你一行一行地输入源代码，每输入一行就执行一行。

直接运行`.py`文件相当于启动了Python解释器，然后一次性把`.py`文件的源代码给执行了，你是没有机会以交互的方式输入源代码的。

用Python开发程序，完全可以一边在文本编辑器里写代码，一边开一个交互式命令窗口，在写代码的过程中，把部分代码粘到命令行去验证，事半功倍！前提是得有个27'的超大显示器！

Python代码运行助手

Python代码运行助手可以让你在线输入Python代码，然后通过本机运行的一个Python脚本来执行代码。原理如下：

- 在网页输入代码：

![write-py-code]
(https://www.liaoxuefeng.com/files/attachments/1183575581181440/l)

- 点击`Run`按钮，代码被发送到本机正在运行的Python代码运行助手；
- Python代码运行助手将代码保存为临时文件，然后调用Python解释器执行代码；
- 网页显示代码执行结果：

![py-code-result]
(https://www.liaoxuefeng.com/files/attachments/1183575446104096/l)

下载

点击右键，目标另存为：[learning.py]
(https://raw.githubusercontent.com/michaelliao/learn-python3/master/teach/learning.py)

[备用下载地址](https://gitee.com/liaoxuefeng/learn-java/raw/master/teach/learning.py)

```
### 运行

在存放`learning.py`的目录下运行命令：
```

C:\Users\michael\Downloads> python learning.py

```
如果看到`Ready for Python code on port 39093...`表示运行成功，不要关闭命令行窗口，最小化放到后台运行即可：

```ascii
┌──────────────────────────────────────────────────────────┐
|Command Prompt                                - □ x |
├──────────────────────────────────────────────────────────┤
|Microsoft Windows [Version 10.0.0]                        |
|(c) 2015 Microsoft Corporation. All rights reserved.      |
|                                                          |
|C:\Users\michael\Downloads> python learning.py            |
|Ready for Python code on port 39093...                    |
|Press Ctrl + C to exit...                                 |
|                                                          |
|                                                          |
|                                                          |
|                                                          |
|                                                          |
└──────────────────────────────────────────────────────────┘
```

启动时如果遇到类似`UnicodeDecodeError`的如下错误：

```
Traceback (most recent call last):
  File "learning.py",
    ...
    hostname, aliases, ipaddrs = gethostbyaddr(name)
UnicodeDecodeError: 'utf-8' codec can't decode byte ...
```

这是因为Python自带的`socket`库试图解析计算机名称的时候遇到中文报错。可以把计算机名称改成英文，然后重启。

试试效果

需要支持HTML5的浏览器：

- IE >= 9
- Firefox
- Chrome
- Sarafi
```

输入和输出

输出

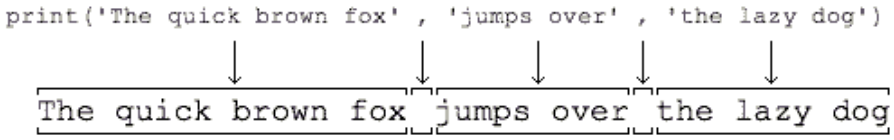
用`print()`在括号中加上字符串，就可以向屏幕上输出指定的文字。比如输出`'hello, world'`，用代码实现如下：

```
>>> print('hello, world')
```

`print()`函数也可以接受多个字符串，用逗号“,”隔开，就可以连成一串输出：

```
>>> print('The quick brown fox', 'jumps over', 'the lazy dog')
The quick brown fox jumps over the lazy dog
```

`print()`会依次打印每个字符串，遇到逗号“,”会输出一个空格，因此，输出的字符串是这样拼起来的：



`print()`也可以打印整数，或者计算结果：

```
>>> print(300)
300
>>> print(100 + 200)
300
```

因此，我们可以把计算`100 + 200`的结果打印得更漂亮一点：

```
>>> print('100 + 200 =', 100 + 200)
100 + 200 = 300
```

注意，对于`100 + 200`，Python解释器自动计算出结果`300`，但是，`'100 + 200 ='`是字符串而非数学公式，Python把它视为字符串，请自行解释上述打印结果。

输入

现在，你已经可以用`print()`输出你想要的结果了。但是，如果要让用户从电脑输入一些字符怎么办？Python提供了一个`input()`，可以让用户输入字符串，并存放到一个变量里。比如输入用户的名字：

```
>>> name = input()
Michael
```

当你输入`name = input()`并按下回车后，Python交互式命令行就在等待你的输入了。这时，你可以输入任意字符，然后按回车后完成输入。

输入完成后，不会有任何提示，Python交互式命令行又回到`>>>`状态了。那我们刚才输入的内容到哪去了？答案是存放到`name`变量里了。可以直接输入`name`查看变量内容：

```
>>> name
'Michael'
```

什么是变量？请回忆初中数学所学的代数基础知识：

设正方形的边长为`a`，则正方形的面积为`a x a`。把边长`a`看做一个变量，我们就可以根据`a`的值计算正方形的面积，比如：

若a=2，则面积为a x a = 2 x 2 = 4；

若a=3.5，则面积为a x a = 3.5 x 3.5 = 12.25。

在计算机程序中，变量不仅可以为整数或浮点数，还可以是字符串，因此，`name`作为一个变量就是一个字符串。

要打印出`name`变量的内容，除了直接写`name`然后按回车外，还可以用`print()`函数：

```
>>> print(name)
Michael
```

有了输入和输出，我们就可以把上次打印`'hello, world'`的程序改成有点意义的程序了：

```
name = input()
print('hello,', name)
```

运行上面的程序，第一行代码会让用户输入任意字符作为自己的名字，然后存入`name`变量中；第二行代码会根据用户的名字向用户说`hello`，比如输入`Michael`：

```
C:\Workspace> python hello.py
Michael
hello, Michael
```

但是程序运行的时候，没有任何提示信息告诉用户：“嘿，赶紧输入你的名字”，这样显得很不友好。幸好，`input()`可以让你显示一个字符串来提示用户，于是我们把代码改成：

```
name = input('please enter your name: ')
print('hello,', name)
```

再次运行这个程序，你会发现，程序一运行，会首先打印出`please enter your name:`，这样，用户就可以根据提示，输入名字后，得到`hello, xxx`的输出：

```
C:\Workspace> python hello.py
please enter your name: Michael
hello, Michael
```

每次运行该程序，根据用户输入的不同，输出结果也会不同。

在命令行下，输入和输出就是这么简单。

小结

- 任何计算机程序都是为了执行一个特定的任务，有了输入，用户才能告诉计算机程序所需的信息，有了输出，程序运行后才能告诉用户任务的结果。
- 输入是Input，输出是Output，因此，我们把输入输出统称为Input/Output，或者简写为IO。
- `input()`和`print()`是在命令行下面最基本的输入和输出，但是，用户也可以通过其他更高级的图形界面完成输入和输出，比如，在网页上的一个文本框输入自己的名字，点击“确定”后在网页上看到输出信息。

练习

请利用`print()`输出`1024 * 768 = xxx`：

```
# -*- coding: utf-8 -*-
```