模块的导入：如何使用其他人编写好的代码功能?

目录

模块的作用

- 什么是模块?

  模块是存放了函数的且以 .py 结尾的文件

- 模块的作用:

  将函数放在模块文件里，有助于隐藏代码细节，使程序的主要逻辑更清晰

导入模块的方法

- 导入整个模块

  import os

- 导入模块的特定方法

  from os import chdir

  from os import chdir, getcwd

导入模块的方法

- 多个模块放在一个文件夹中，该文件夹称作包
- 包的导入与模块的导入相同：

  “import 包”或“from 包 import 模块”

导入模块的方法

- 为导入的模块指定别名：

```
import numpy as np
```

- 不建议使用 * 导入模块所有函数：

```
from os import *
```

使用模块中定义的方法

不同的导入方法，使用函数的格式也不同，以使用 getcwd() 函数为例：

```
import os

os.getcwd()

from os import getcwd

getcwd()
```

使用模块中定义的方法

如果模块被放入包中，调用模块中的方法：

包.模块.方法

总结

1. 模块保存的文件是以 .py 结尾的文件
2. 模块导入可以使用 import 和 from … import 两种方法
3. 导入模块后，模块的函数可以采用“模块.函数”和“函数”两种方式调用

课后作业

为了获取今天的日期，需要使用 datetime 包里 date 模块中的 today() 方法。

today() 方法调用时可以使用下面三种格式，请问下面三种格式各自要采用什么语法来进行模块的导入？

格式一：datetime.date.today()

格式二：date.today()

格式三：today()

标准库：Python 默认提供的便捷功能有哪些?

目录

标准库中的常见组件

标准库中的所有组件目录：

https://docs.python.org/zh-cn/3.10/library/index.html

标准库中的常见组件

内置函数、类型、异常

文本处理

数字

文件和目录

通用操作系统

并发执行

网络和进程间通信

互联网协议

如何通过官方文档学习标准库

需求：

通过 Python 模拟浏览器访问 http://www.baidu.com

如何通过官方文档学习标准库

实现步骤：

1. 通过官方文档找到互联网协议和支持
2. 找到 HTTP、urllib 目录参考
3. 最终确定 urllib.request 可以实现相应功能
4. 参考文档中的“例子”和 urlopen() 函数的参数说明，实现模拟浏览器的目的

* 官方文档：

  https://docs.python.org/zh-cn/3.10/library/urllib.request.html#examples

总结

1. 标准库能支持各种扩展功能，并提高 Python 的开发效率

2. 标准库中的包非常多，不必逐一使用和熟练掌握，掌握文档的查阅方法即可

课后作业

某项目采用 zip 压缩格式，保存了文件 a.txt，它的目录结构如下：

```
a.zip
 ├── a.txt
 ├── b.txt
 ...
```

请参考官方文档，使用 Python 读取 a.txt 文件中的内容，并打印到终端。

自定义模块：如何编写一个完整功能?

目录

创建自定义模块

- 使用以 .py 为结尾的文件名保存模块
- 在文件中定义的属性、函数都可以被调用
- 在文件中定义的类可以在引用模块时进行实例化

自定义模块注意事项

1. 导入自定义模块时，需确保导入路径正确
2. 自定义模块的文件名称尽量避免特殊字符，文件名应避免和标准库重名
3. 自定义模块多次导入，模块中的代码也只能被执行一次
4. 自定义模块中应为函数定义，避免在模块中编写函数调用代码，或将函数调用代码放在 __name__ 代码块中

总结

1. 自定义模块使用以 .py 结尾的文件名作为模块的保存位置
2. 自定义模块名称不应和标准库名称重名

课后作业

请编写一个自定义模块，模块中定义 date() 和 time() 两个函数，分别用于显示当前的日期和系统时间。

第三方模块的使用：如何使用其他人编写的代码?

目录

第三方模块的安装

- 安装第三方模块使用 pip 命令

  pip3.10 install —— 第三方模块名称

  python3.10 –m pip install —— 第三方模块名称

* https://pypi.org/project/requests/

虚拟环境

虚拟环境的用途：

- 解决多个模块依赖的问题
- 一次性安装多个指定版本的模块
- 避免对默认环境造成污染

虚拟环境

创建虚拟环境：

```
python –m venv myvenv
```

- venv 虚拟环境模块
- myvenv 保存虚拟环境的文件夹

虚拟环境

- 将当前安装的包及其版本保存到文件中：

  pip3.10 freeze > requirements.txt

- 激活虚拟环境：

  source myvenv/bin/activate

- 在虚拟环境中导入指定的包：

  pip3.10 install –r requirements.txt

- 离开虚拟环境：

  deactivate

加速第三方模块安装

- 临时加速

  pip install –i https://pypi.tuna.tsinghua.edu.cn/simple package_name

- 永久加速

  cat ~/pip.conf

  [global]
  index–url = http://mirrors.aliyun.com/pypi/simple/
  [install]
  trusted–host=mirrors.aliyun.com

总结

1. 第三方模块使用 pip 命令进行安装

2. 为了避免污染默认环境，应尽量使用虚拟环境安装第三方模块

3. 由于 pip 默认使用国外的源，下载安装包会出现超时的问题，可以通过参数 –i 或修改配置文件加速下载过程

课后作业

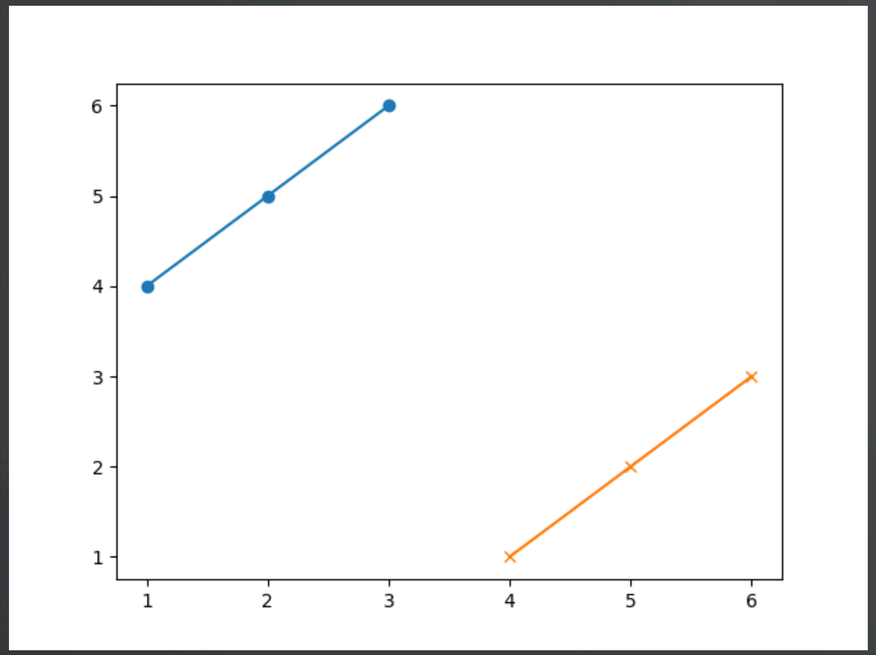
请安装第三方模块 Matplotlib, 安装后运行下方代码并显示执行结果。

代码如下：

```
import matplotlib.pyplot as plot
x = [ 1, 2, 3 ]
y = [ 4, 5, 6 ]
plot.plot(x, y, marker = 'o' )
plot.plot(y, x, marker = 'x' )
plot.show()
```

课后作业

代码执行结果如下：



小试牛刀：如何使用 Python 为函数求导?

目录

需求分析

需求：

使用 Python 计算函数的导数

需求分析

需求解析：

1. 找到实现 Python 求导的第三方模块
2. 安装第三方模块并测试
3. 使用第三方模块计算函数的导数
4. 安装绘图模块并测试
5. 绘制函数图像

使用第三方模块实现函数求导

- 安装第三方模块：

  pip3.10 install sympy

- 使用第三方模块：

  from sympy import Derivative

使用第三方模块实现函数求导

- 测试第三方模块：

```
from sympy import Symbol
x = Symbol('x')
d = Derivative(y, x)
d.doit()
d.doit().subs({t:1})
```

编写程序并测试

- 编写程序

```
from sympy import Symbol
x = Symbol('x')
y = f(x)
d = Derivative(y, x)
d.doit()
d.doit().subs({t:1})
```

总结

1. Python 可以借助 SymPy 解决数学问题
2. Python 可以借助 Matplot 进行绘图
3. 要实现复杂的功能，需要多个模块配合

课后作业

请使用 SymPy 模块计算，当 n 趋近无穷大时，(1+1/n) 的 n 次方的极限。

THANKS