教育大数据应用技术国家工程实验室
NATIONAL ENGINEERING LABORATORY FOR EDUCATIONAL BIG DATA
国家数字化学习工程技术研究中心
NATIONAL ENGINEERING RESEARCH CENTER FOR E-LEARNING

Python数据处理

1. Python 语言基础

刘乐元 副教授

2021年春季

群名称:Python数据处理

群　号:983233851

- ✓ 答疑
- ✓ 交流
- ✓ 作业（群邮件）
- ✓ 通知 & 请假

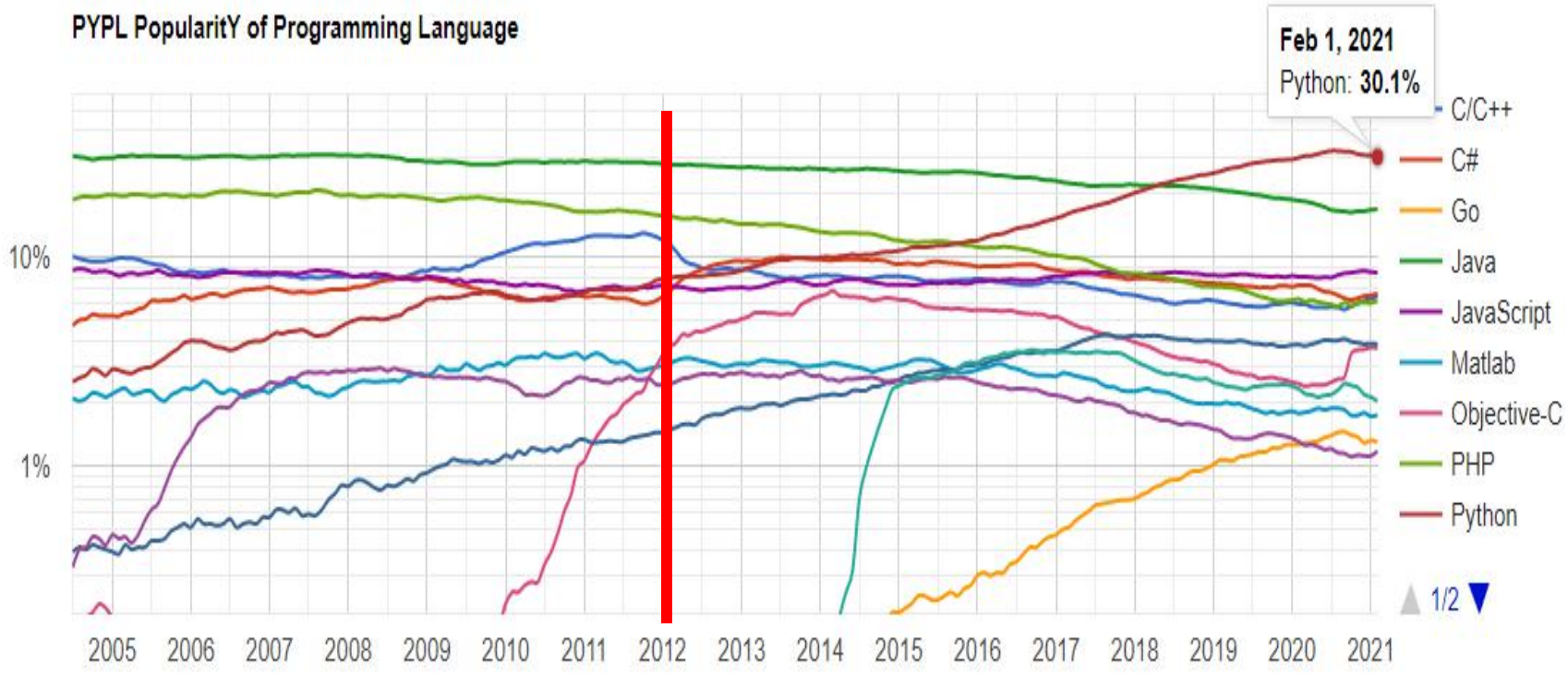
- ✓ 缺勤需要请假
- ✓ 手机静音
- ✓ 允许带笔记本电脑
- ✓ 按时交作业

1.1 Python 简介

python™

✓ Python 1991年面世，当前已经成长为**最流行**的解释型语言。

Guido van Rossum



https://pypl.github.io/PYPL.html

1.1 Python 简介

- ✓ 数据分析（NumPy、SciPy、Matplotlib、pandas）
- ✓ 人工智能（TransorFlow、PyTorch、Karas）
- ✓ 爬虫（urllib、Selenium 、BeautifulSoup 、Scrapy）
- ✓ 网页开发（Django、flask、TurboGears、web2py ）
- ✓ 办公自动化（ openpyxl， python-docx， python-pptx ）

1.1 Python 简介

- ✓ 解释型语言
- ✓ 脚本语言
- ✓ 胶水语言
- ✓ 简单易学
- ✓ 开源
- ✓ 高级语言
- ✓ 可移植性
- ✓ 强大的功能（社区）
- ✓ 可扩展性

1.2 Python 的安装

推荐使用Anaconda http://anaconda.com/downloads

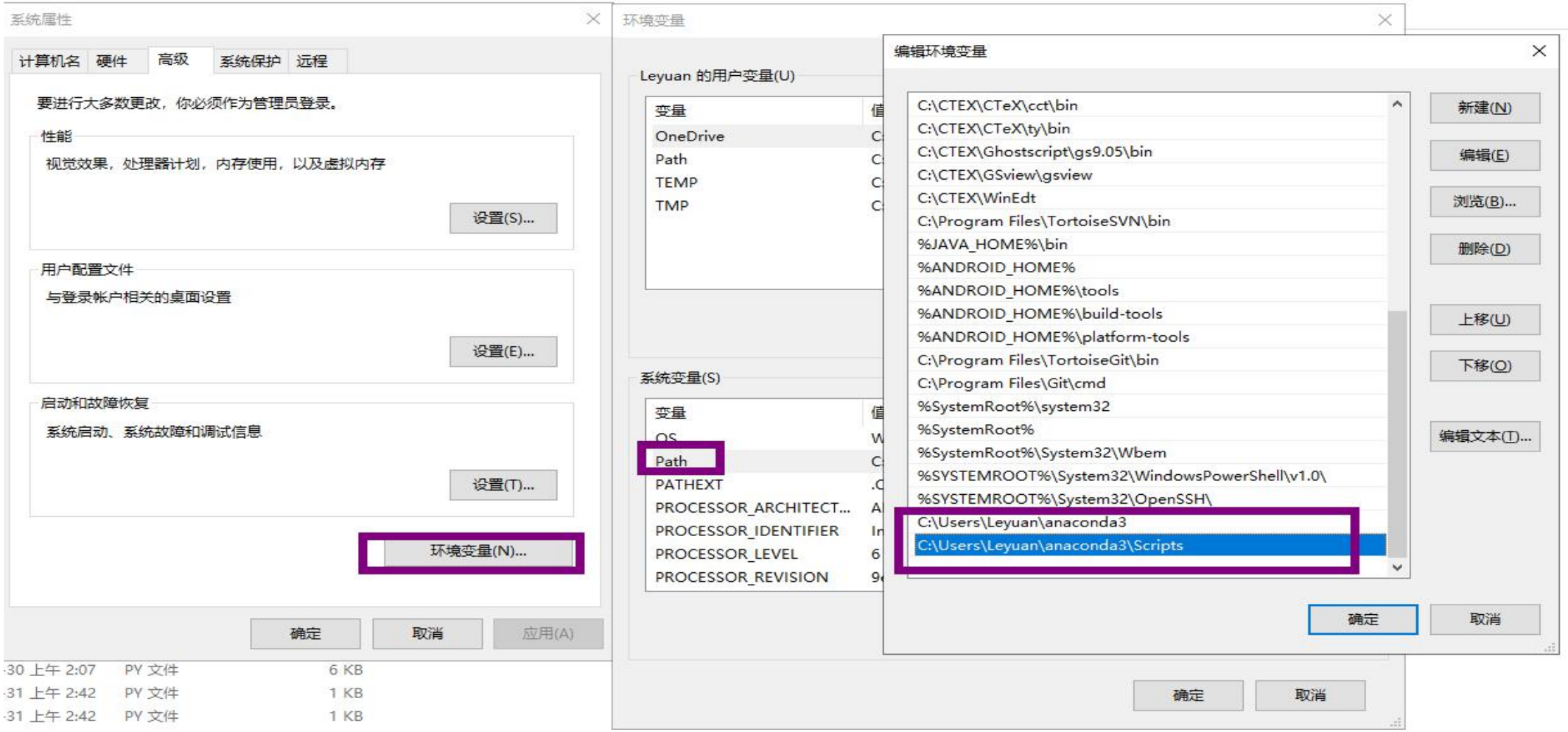
Open Source

Anaconda Individua
world's most popul
distribution platform
million users worldv
in our long-term co
supporting the Anac
ecosystem, the platf
Python data science

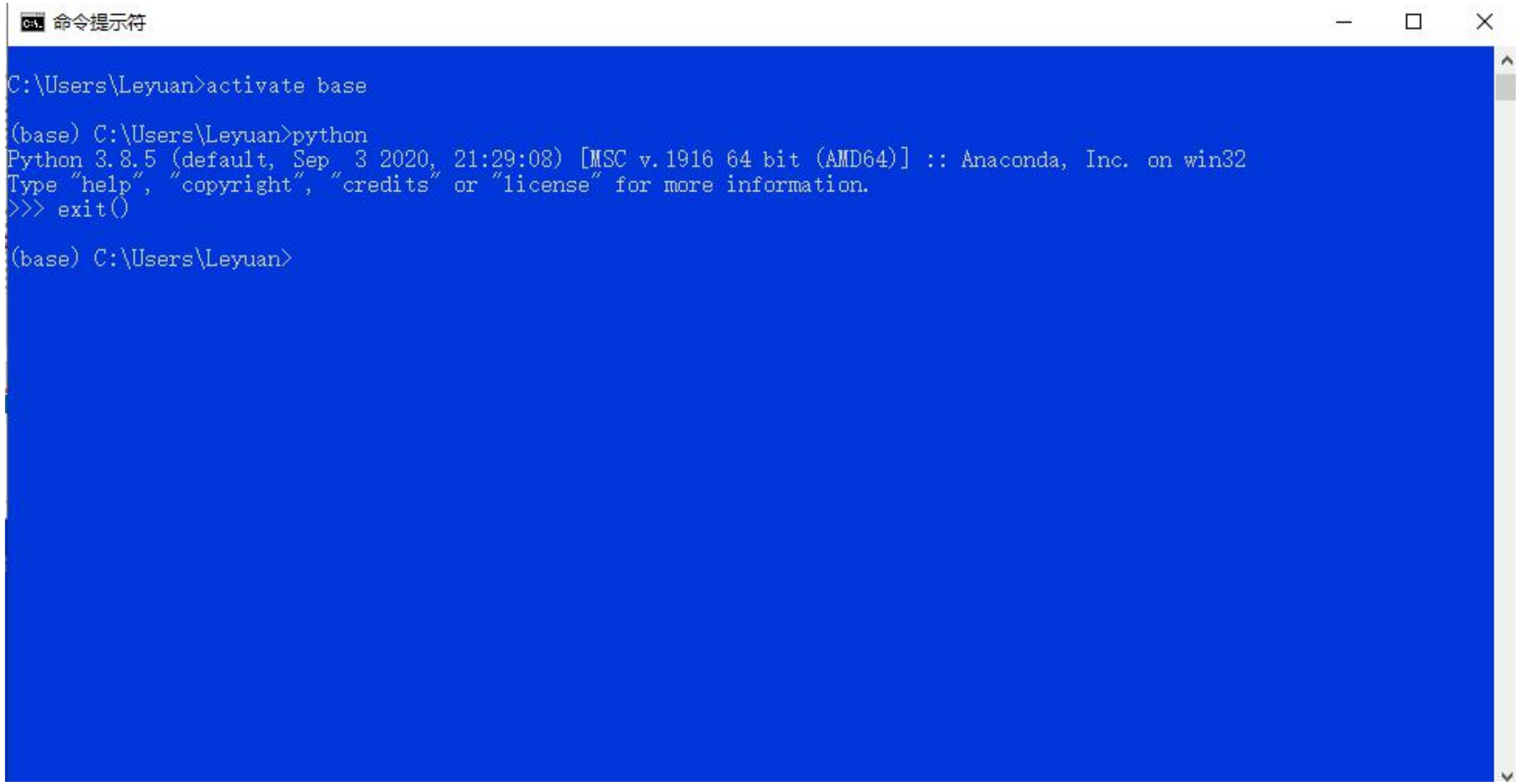
Manage Environments

Individual Edition is an open source, flexible solution that provides the utilities to build, distribute, install, update, and manage software in a cross-platform manner. Conda makes it easy to manage multiple data environments that can be maintained and run separately without interference from each other.

1.2 Python 的安装

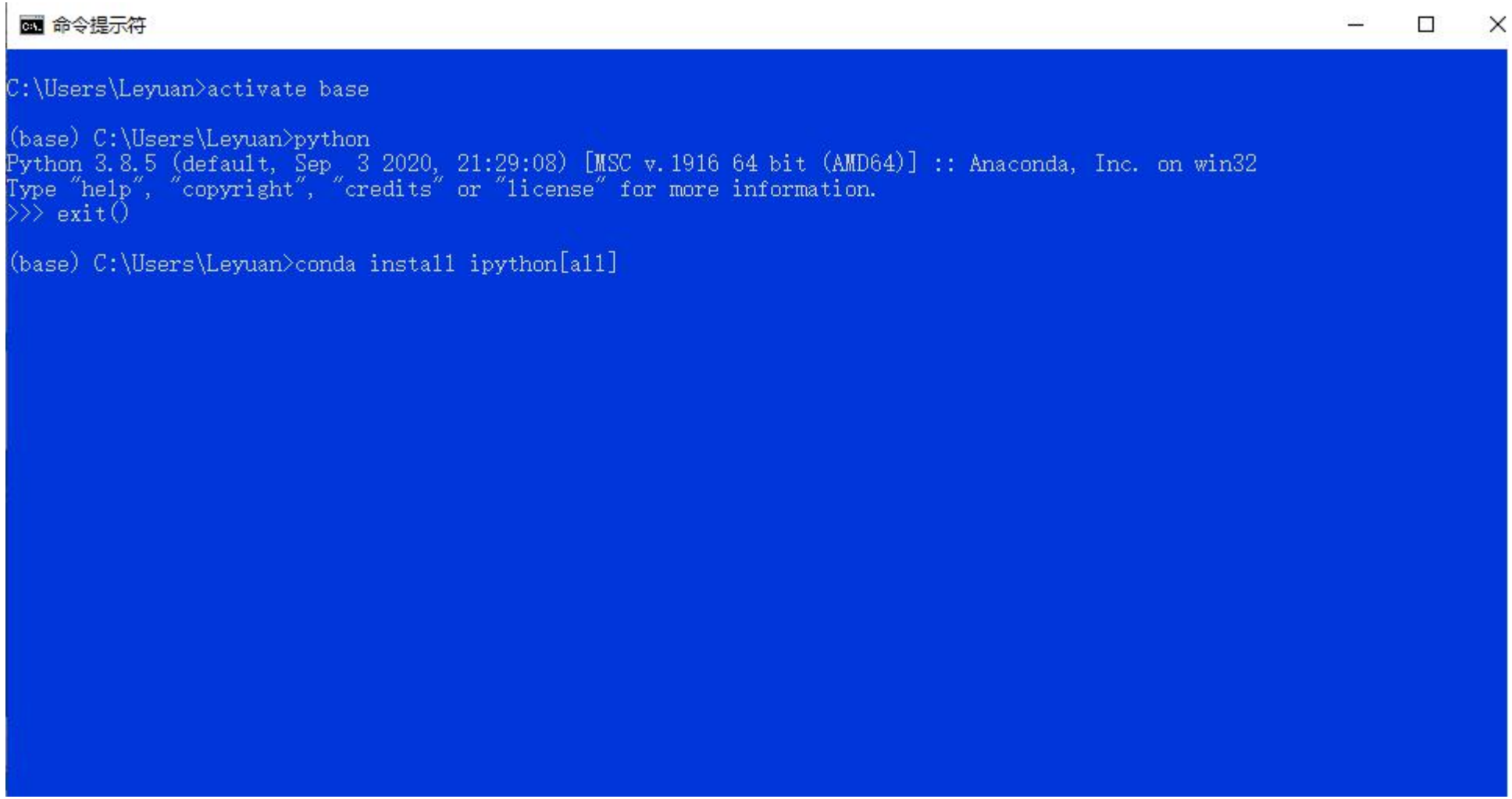


1.2 Python 的安装



1.2 Python 的安装

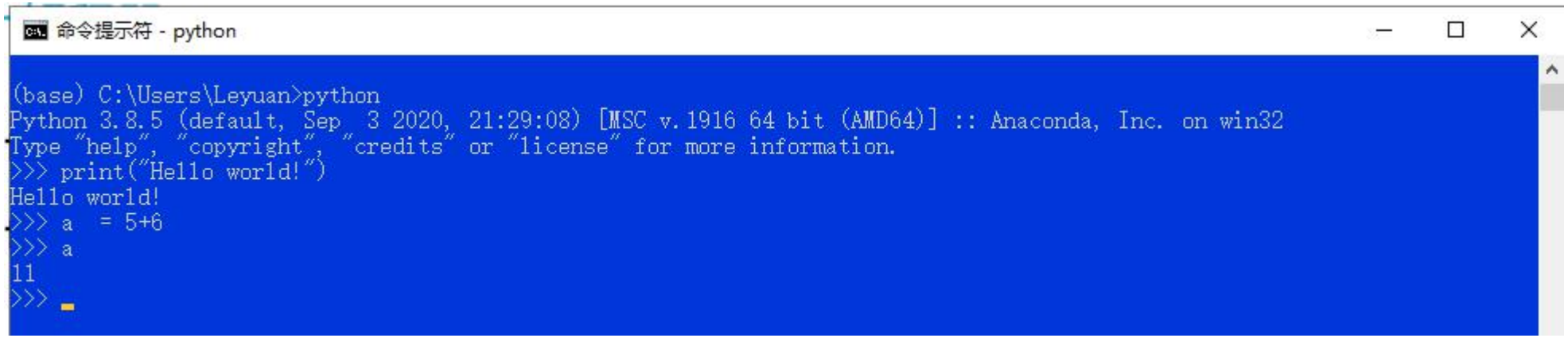
安装及更新Python包



1.2 Python 的安装

Python解释器

- ✓ Python是一种解释型语言。
- ✓ Python解释器通过一次执行一条语句来运行程序



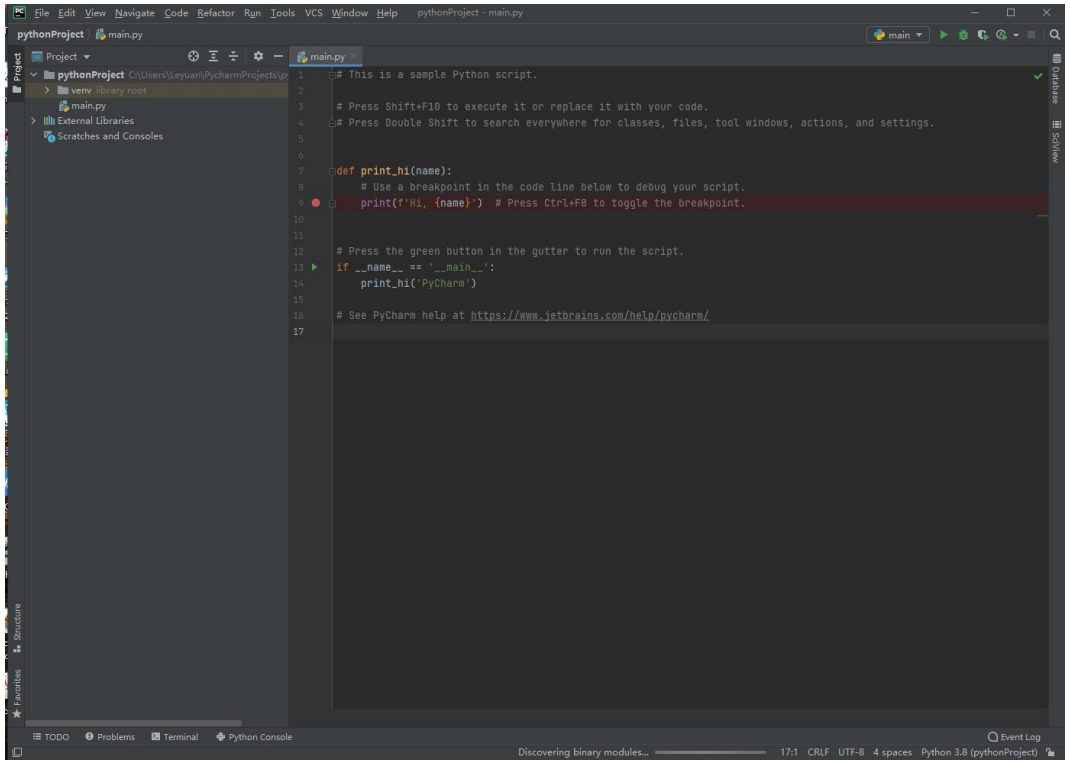
1.2 Python 的安装

Python解释器

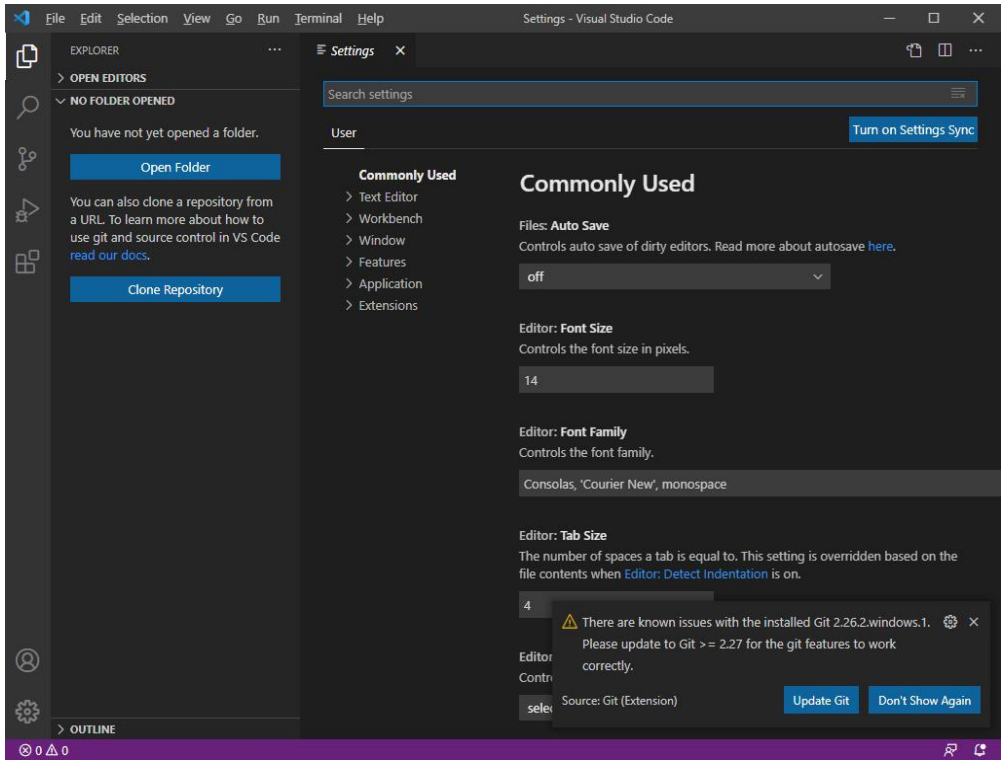
- ✓ 使用Python解释器运行写好的代码

1.2 Python 的安装

Python IDE



PyCharm



VS code

1.3 IPython与Jupyter Notebook

- ✓ IPython是一个Python的交互式shell，比默认的python shell具有更好的交互功能，支持变量自动补全，自动缩进，支持bash shell命令，内置了许多很有用的功能和函数。
- ✓ 学习IPython将会让我们以一种更高的效率来使用python。同时它也是利用Python进行科学计算和交互可视化的一个最佳的平台。

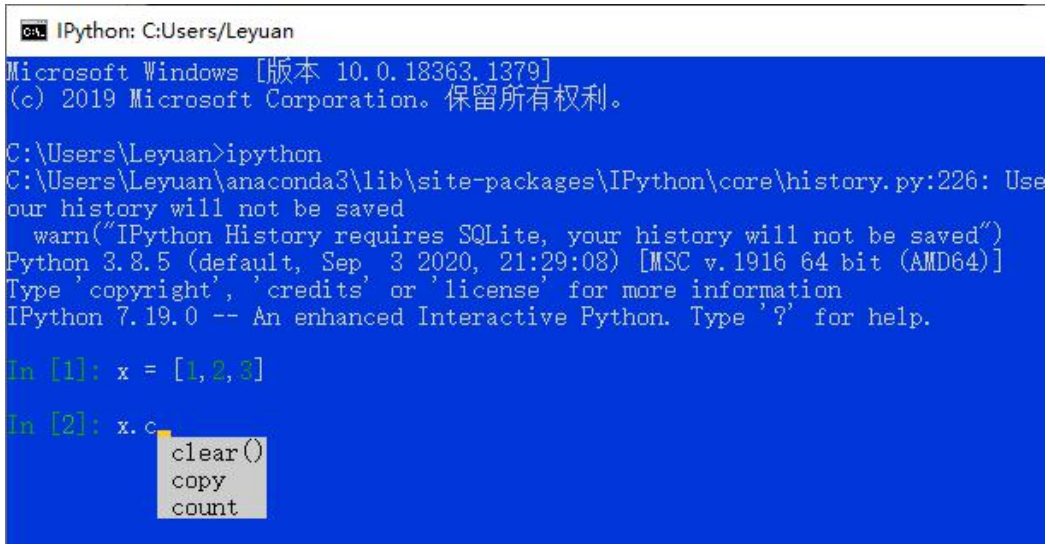
IPython提供两大组件：

- **一个强大的python交互式shell**
- **供Jupyter notebooks**

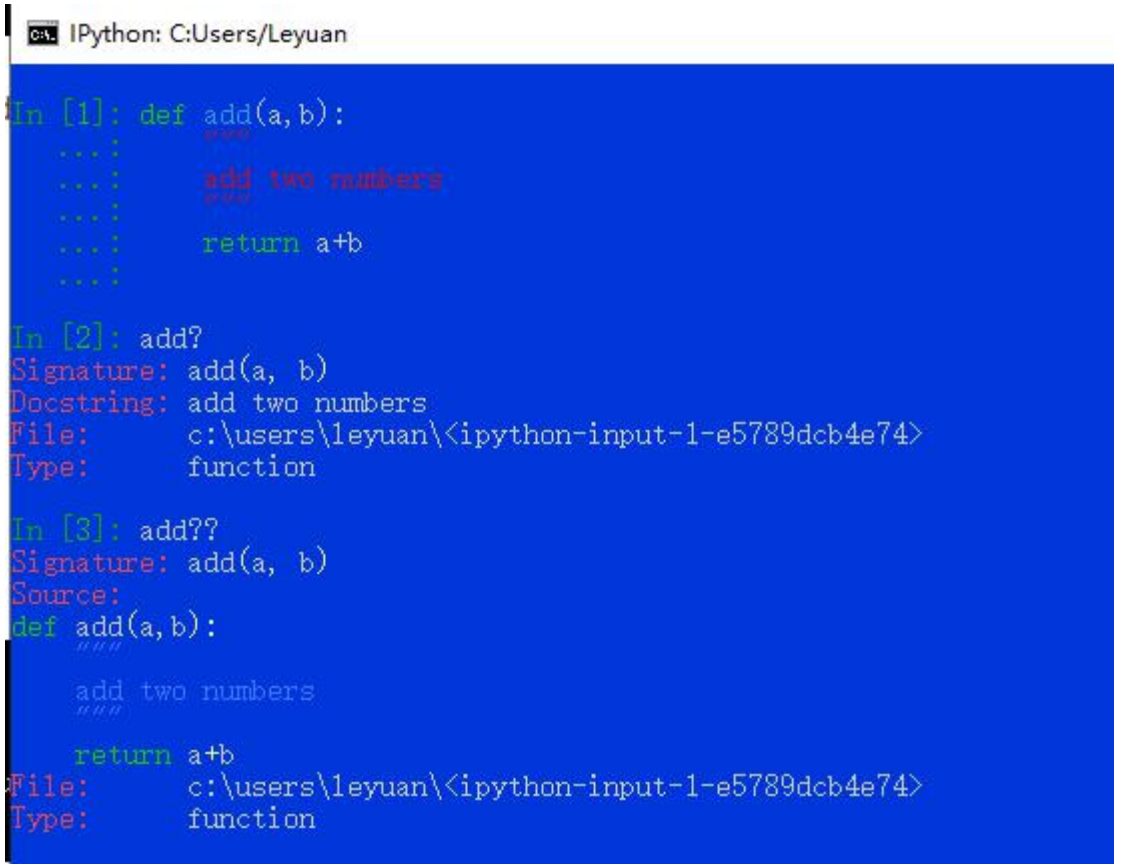
1.3 IPython与Jupyter Notebook

- ✓ Tab补全
- ✓ 内省
- ✓ 使用%run命令运行脚本
- ✓ 使用ipython作为系统shell
- ✓ 魔术命令
  - ✓ 使用%timeit命令快速测量时间
  - ✓ 使用%pdb命令快速debug
  - ✓ 使用%matplotlib进行交互计算

1.3 IPython交互式shell



Tab键自动补全

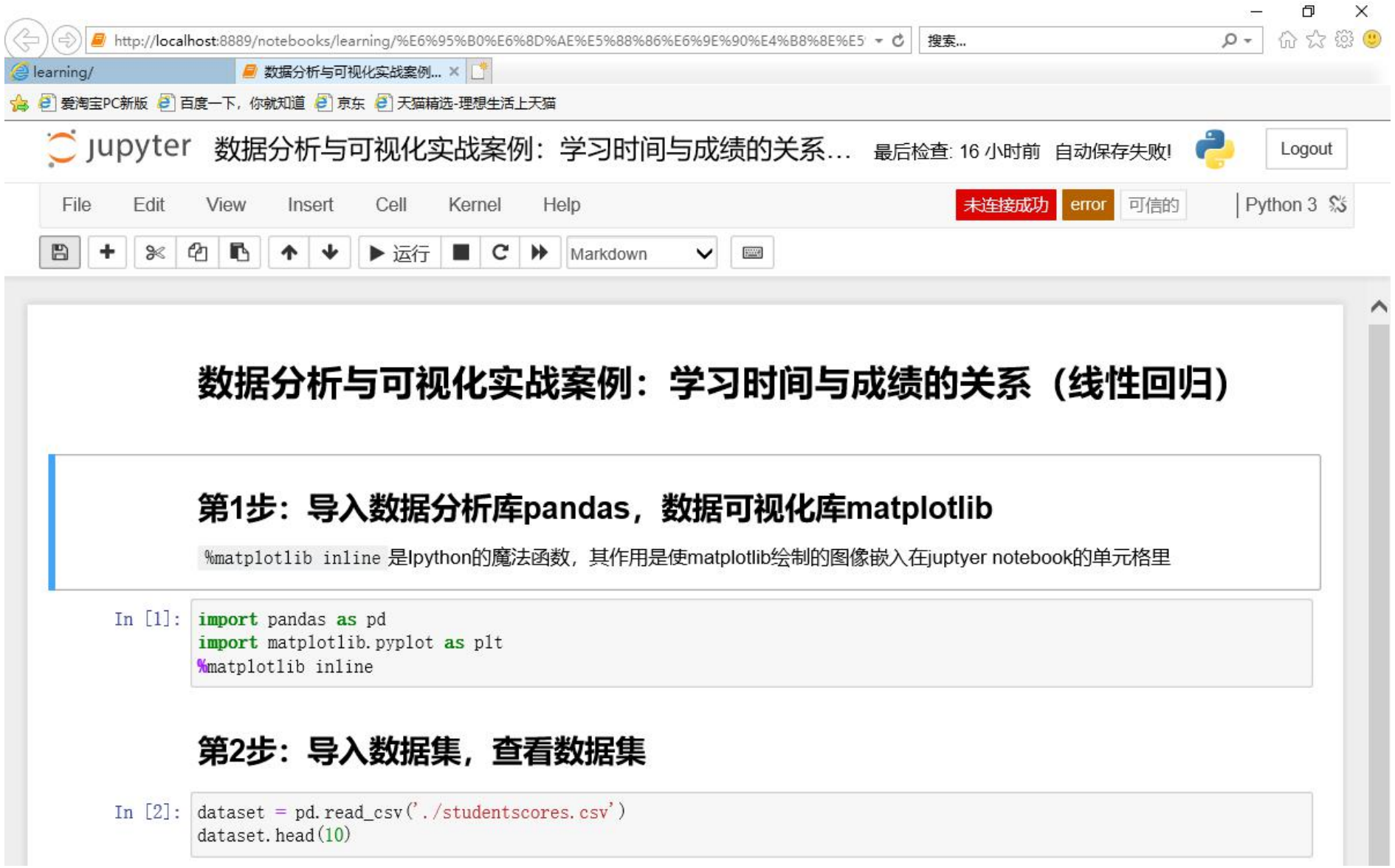


1.3 Jupyter Notebook

- ✓ Jupyter Notebook使用浏览器作为界面，向后台的IPython服务器发送请求，并显示结果。在浏览器的界面中使用单元(Cell)保存各种信息。Cell有多种类型，需要强调的是，它也支持MarkDown语法，所以可以有MarkDown格式化文本单元，也可以有表示代码的Code单元。
- ✓ Jupyter Notebook有一个重要的特点就是：可重复性的互动计算，这意味着我们可以重复更改并且执行曾经的输入记录。它可以保存成其他很多格式，比如Python脚本，HTML，PDF等，所以它可以记录我们的演算过程。很多课程，博客以及书籍都是用Notebook写的。

1.3Jupyter Notebook

1.3Jupyter Notebook

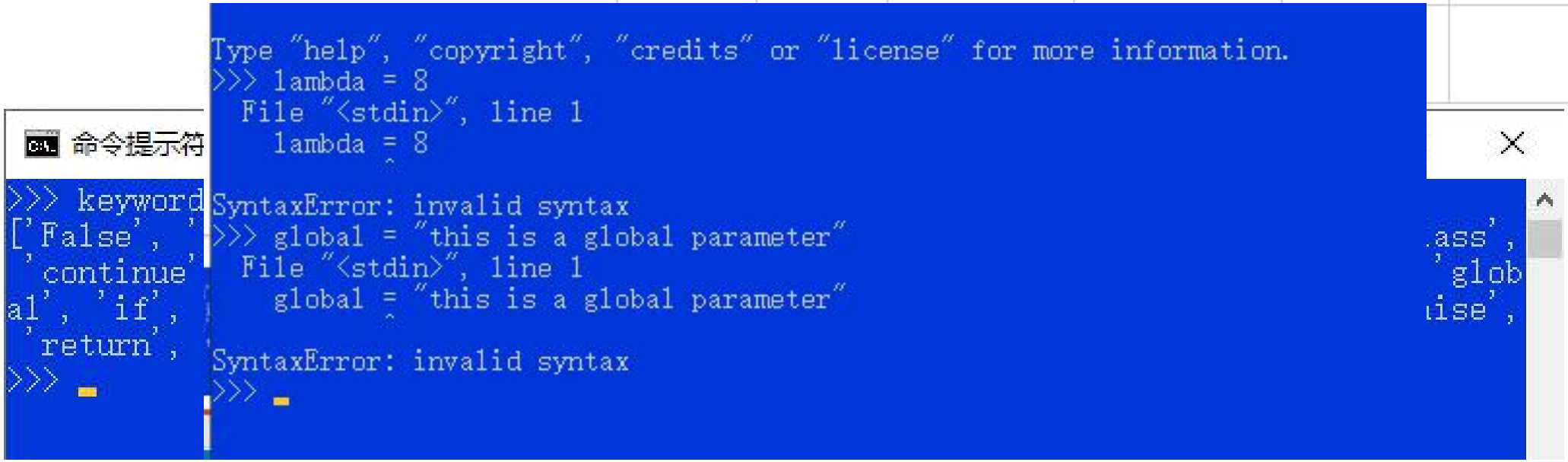


1.4 Python 语言语义

（1）关键字（保留字）

内置函数

and	as	assert	break	class	continue
def	del	elif	else	except	finally
for	from	False	global	if	import
in	is	lambda	nonlocal	not	None
or	pass	raise	return	try	True



1.4 Python 语言语义

（1）关键字（保留字）

内置函数

abs()	all()	any()	basestring()	bin()
bool()	bytearray()	callable()	chr()	classmethod()
cmp()	compile()	complex()	delattr()	dict()
dir()	divmod()	enumerate()	eval()	execfile()
file()	filter()	float()	format()	frozenset()
getattr()	globals()	hasattr()	hash()	help()
hex()	id()	input()	int()	isinstance()

1.4 Python 语言语义

（1）关键字（保留字）

内置函数

issubclass()	iter()	len()	list()	locals()
long()	map()	max()	memoryview()	min()
next()	object()	oct()	open()	ord()
pow()	print()	property()	range()	raw_input()
reduce()	reload()	repr()	reversed()	zip()
round()	set()	setattr()	slice()	sorted()

1.4 Python 语言语义

（1）关键字（保留字）

内置函数

staticmethod()	str()	sum()	super()	tuple()
type()	unichr()	unicode()	vars()	xrange()
Zip()	__import__()	apply()	buffer()	coerce()
intern				

1.4 Python 语言语义

（2）变量命名

命令提示符 - python

```
>>> first_str = "Hello world!"
>>> print(first_str)
Hello world!
>>> print  =9
>>> print(first_str)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'int' object is not callable
>>> type(print)
<class 'int'>
>>> if = 8
  File "<stdin>", line 1
    if = 8
       ^
SyntaxError: invalid syntax
```

覆盖掉）

组合（不允许使用特

- ✓ student_no(推荐使用)
- ✓ StudentNo

1.4 Python 语言语义

（3）使用缩进，而不是大括号

- Python使用缩进（Tab或空格）来组织代码，而不是像其他语言如C++，Java那样用大括号来组织代码。
- 一个冒号代表一个缩进代码块的开始。
- 单个代码块中的所有的代码必须保持相同的缩进，直到代码块结束。

```
for x in array:
    if x < pivot:
        less.append(x)
    else:
        greater.append(x)
```

1.3 Python 语言语义

（4）注释 （#）

- 所有写在#号之后的文本会被Python解释器所忽略。

```
results = []
    #逐行处理文件
    for line in file_handle:
        if len(line) == 0:
            results.append(x) #添加
```

1.3 Python 语言语义

（5）一切皆为对象

- Python的一个特征就是对象模型的一致性。
- 每个数值、字符串、数据结构、函数、类、模块，以及所有存在于Python解释器中的事物，都是Python对象。
- 每个对象都会关联到一种类型（例如，字符串、函数）和内部数据。

Python中的对象包含三个要素：id, type, value

其中：

- id：用来唯一标示一个对象
- type：表示对象的类型
- value：是对象的值

1.3 Python 语言语义

（6）变量和参数传递

- 在Python中对一个变量（或者变量名）赋值时，就是创建了一个指向等号右边对象的引用

```
>>> a
[1, 2, 3, 4]
>>> b
[1, 2, 3, 4]
>>> b = a.copy()
>>> b
[1, 2, 3, 4]
>>> a.append(5)
>>> a
[1, 2, 3, 4, 5]
>>> b
[1, 2, 3, 4]
>>> id(a)
2525585553472
>>> id(b)
2525585659584
2525585553472
```

```
>>> s1 = "string1"
>>> id(s1)
2525585460720
>>> s1 += "append"
>>> id(s1)
2525585480432
```

```
>>> i = 6
>>> j = i
>>> i = 9
>>> j
6
```

1.3 Python 语言语义

（7）动态引用和强类型

- Python中的对象引用并不涉及类型。
- Python是“非类型化语言”？

```
>>> a = 7
>>> type(a)
<class 'int'>
>>> a = 'foo'
>>> type(a)
<class 'str'>
>>> a = [1, 2, 3]
>>> type(a)
<class 'list'>
>>>
```

```
>>> c = 2.5
>>> type(a)
<class 'int'>
>>> type(c)
<class 'float'>
>>> d = a/b
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for /: 'int' and 'str'
>>> d = a/c
>>> d
2.0
>>> type(d)
<class 'float'>
>>>
```

1.4 Python 语言语义

（8）二元操作符

操作符	描述
a + b	a 加 b
a - b	a 减 b
a * b	a 乘 b
a / b	a 除以 b
a // b	a 整除 b
a ** b	a的b次
a & b	对于布尔值a与b; 对于整型值是按位与。 **and**
a | b	对于布尔值a或b; 对于整型值是按位或。 **Or**
a ^ b	对于布尔值a异或b; 对于整型值是按位异或
a == b	a是否等于b
a != b	a是否不等于b
a <= b, a<b	大小比较
a >=b, a>b	大小比较
a is b, a is not b	a和b是否是一个对象

1.4 Python 语言语义

- 二元操作符

```
a = [1, 2, 3]
b = a
c = list(a)
a is b
a is not c
```

```
a = None
a is None
```

1.5 Python 标量类型

- Python的标准库中拥有一个小的内奸类型集合，用来处理数值数据、字符串、布尔值、日期和时间。

类型	描述
None	Python的“null”值（单例）
str	字符串类型，包含Unicode（UTF-8编码）字符串
bytes	原生ASCII字节（或者Unicode编码字节）
float	双精度64位浮点数值
bool	True 或者 False
int	任意精度无符号整数

1.5 Python 标量类型

（1）数字类型

- Python的基础数字类型就是int 和 float

```
ival = 17239871
ival ** 6
fval = 7.243
fval2 = 6.78e-5
```

1.4 Python 标量类型

(2) 字符串

```
>>> str1 = 'this is a string'
>>> str2 = "this is another string"
>>> str3 = """
... this is a longer string
... that spans multiple lines
... """
...
>>> str1
'this is a string'
>>> str2
'this is another string'
>>> str3
'\nthis is a longer string\nthat spans multiple lines\n'
>>>
```

```
>>> str3
'\nthis is a longer string\nthat spans multiple lines\n'
>>> str3[5] = 'a'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>>
```

1.4 Python 标量类型

（4）布尔值

- True
- False

1.4 Python 标量类型

（5）类型转换

```
s = '3.14159'
fval = float(s)
type(fval)
int(fval)
bool(fval)
bool(0)
s + fval
```

1.4 Python 标量类型

（6）None

- None是Python的null值类型。如果一个函数没有显式地返回值，则它会返回None

```
a=None
a is None
b=5
B is not None
```

```
def add_and_maybe_multiply(a, b, c=None):
    result = a + b

    if c is not None:
        result = result * c

    return result
```

1.4 Python 标量类型

（7）日期和时间

python中内建的datetime模块，提供了datetime、date和time类型

```
from datetime import datetime, date, time
dt = datetime(2011, 10, 29, 20, 30, 21)
dt.day
dt.minute
dt.date()
dt.time()
```

1.4 Python 标量类型

（7）日期和时间

时间、日期与字符串的转换

```
dt.strftime('%m/%d/%Y %H:%M’ )
datetime.strptime('20091031', '%Y%m%d')
```

1.4 Python 标量类型

（7）日期和时间

转换符	描述
%Y	四位的年份
%y	两位位的年份
%m	两位的月值[01,12]
%d	两位的日值[01,31]
%H	小时值 [00,23]（24小时制）
%I	小时值 [01,12]（12小时制）
%M	分值[00,59]
%S	秒值[00,61] (60,61闰秒)
%w	星期，[0,6]
%U	一年中的第几个星期的值[0,53] (星期日是每周第一天)
%W	一年中的第几个星期的值[0,53] (星期一是每周第一天)
%z	UTC时区偏置，格式为+HHMM或-HHMM
%F	%Y-%m-%d的简写（2021-12-13）
%D	%m%d%y的简写（12/13/21）

1.4 Python 标量类型

（7）日期和时间

时间运算

```
dt1 = datetime(2021,6,17,15,0,0)
dt2 = datetime(2021, 3,2, 14,15,0)
delta = dt1 - dt2
datetime.timedelta(days=107, seconds=2700)
```

1.5 Python 控制流

（1）选择（if, elif, else）

```
a = 5; b = 7
c = 8; d = 4
if a < b or c > d:
    print('Made it')
```

```
a = 8; b = 7
c = 5; d = 4
if a > b > c > d:
    print('Made it')
```

1.5 Python 控制流

（2）循环（for）

```
sequence = [1, 2, None, 4, None, 5]
total = 0
for value in sequence:
    if value is None:
        continue
    total += value
```

```
sequence = [1, 2, 0, 4, 6, 5, 2, 1]
total_until_5 = 0
for value in sequence:
    if value == 5:
        break
    total_until_5 += value
```

1.5 Python 控制流

（2）循环（for）

```
for i in range(4):
    for j in range(4):
        if j > i:
            break
        print((i, j))
```

1.5 Python 控制流

(2) 循环 (while)

```
x = 256
total = 0
while x > 0:
    if total > 500:
        break
    total += x
    x = x // 2
```

1.4 Python 控制流

（3）pass

- 用于在代码中表示“什么都不做”。

```
if x < 0:
    print('negative!')
elif x == 0:
    # TODO: put something smart here
    pass
else:
    print('positive!')
```

1.4 Python 控制流

（4）range

- 用于产生一个迭代器。该迭代器生成一个等差整数序列

```
range(10)
list(range(10))
```

```
list(range(0, 20, 2))
list(range(5, 0, -1))
```

```
seq = [1, 2, 3, 4]
for i in range(len(seq)):
    val = seq[i]
```

```
sum = 0
for i in range(100000):
    # % is the modulo operator
    if i % 3 == 0 or i % 5 == 0:
        sum += i
```

1.4 Python 控制流

（5）三元表达式

```
x=5
y= 'pos' if x >0 else 'neg'
```

```
if x>0:
    y =    ‘pos’
else:
    y =    ‘neg’
```

作业

- ✓ 安装Anaconda
- ✓ 熟悉Jupyter Notebook，练习快捷键
- ✓ 使用Jupyter Notebook创建一个Python3文件（代码+MarkDown+LaTeX公式）

谢 谢!