Python程序设计与数据科学导论

胡俊峰 2022-02-21
北京大学计算机学院

课程介绍 —— 主要内容及安排（20次课）

- Python程序设计（5次课）
- Numpy、Pandas、Sicpy、Sklearn等数据分析、数据挖掘相关算法实践与对应的软件包（5次课）
- 时间序列、图像处理、自然语言处理中特征分析与提取模型介绍（3次课）
- Pytorch与深度学习模型简介（3次课）
- 图像处理、自然语言处理经典模型分析（2次课）
- 深度学习模型特征分析与验证*（2次课）

考试及成绩认定

- 平时作业成绩：30%
- 期中成绩（大作业）：30%
- 期末考试（笔试）：40%

课程软件环境：

- Anaconda + python + pytorch 及相关软件包
- 推荐使用pycharm或Vscode作为期中大作业的编程IDE
- 平时作业大多用Jupytor notebook文件形式布置和提交

jupyter 2021-numpy作业-1-old(1) 最新检查点: 2021年4月2日 (更改未保存)

Logout

File Edit View Insert Cell Kernel Widgets Help 不可信 Python 3

运行 Markdown

numpy矩阵操作

首先请你创建三个矩阵A, B, C, D，它们都是服从标准正态分布的矩阵，其中A的大小为20×40，B的大小为40×40，C的大小为40×1，D的大小为40×1.

In [45]:
```
import numpy as np
np.random.seed(1)
# 作业：添加你的代码
```

Q1.1 计算$A + A$, AA^T, A^TA, AB. 然后写一个函数对于输入参数λ，计算$A(B - \lambda I)$.

In [46]:
```
# 作业：添加你的代码
```

Q1.2 计算 $DB^{-1}x = C$

In [47]:
```
# 作业：添加你的代码
```

Q1.3 请通过numpy的条件约束实现RELU函数，并输出RELU(A)

RELU参考：https://zh.wikipedia.org/wiki/%E7%BA%BF%E6%80%A7%E6%95%B4%E6%B5%81%E5%87%BD%E6%95%B0

In [48]:
```
# 作业：添加你的代码
```

向量范数

课程基础参考资料：

- 3.10.2 Documentation (python.org) （Python官方文档及教程）
  - The Python Tutorial — Python 3.10.2 documentation
- https://github.com/jakevdp/PythonDataScienceHandbook（数据科学教程）
  - https://github.com/cuttlefishh/python-for-data-analysis （数据分析技术教程）
- https://pytorch.org/tutorials/beginner/basics/intro.html （Pytorch官方文档及教程）
- https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html
- …

课程不涉及的内容：Full Stack Python

课程先修要求

- 有程序设计与算法基础
- 学习过基础python编程
- 具备一定的高等代数（向量空间模型及线性变换），高等数学（导数、偏导数、卷积）及概率统计（概率、条件概率、贝叶斯分析、概率分布）等相关基础

本课程的定位：

- Python作为一种架构语言最需要深入理解的技术与具体实现
- 数据科学的基本方法与Python环境下的实现。几个代表性应用领域的问题建模方案与算法实现。
- 深度学习框架下的数据建模思想与实现

课程组织形式：

- 授课
- 作业-及作业讲评
- 课程群交流

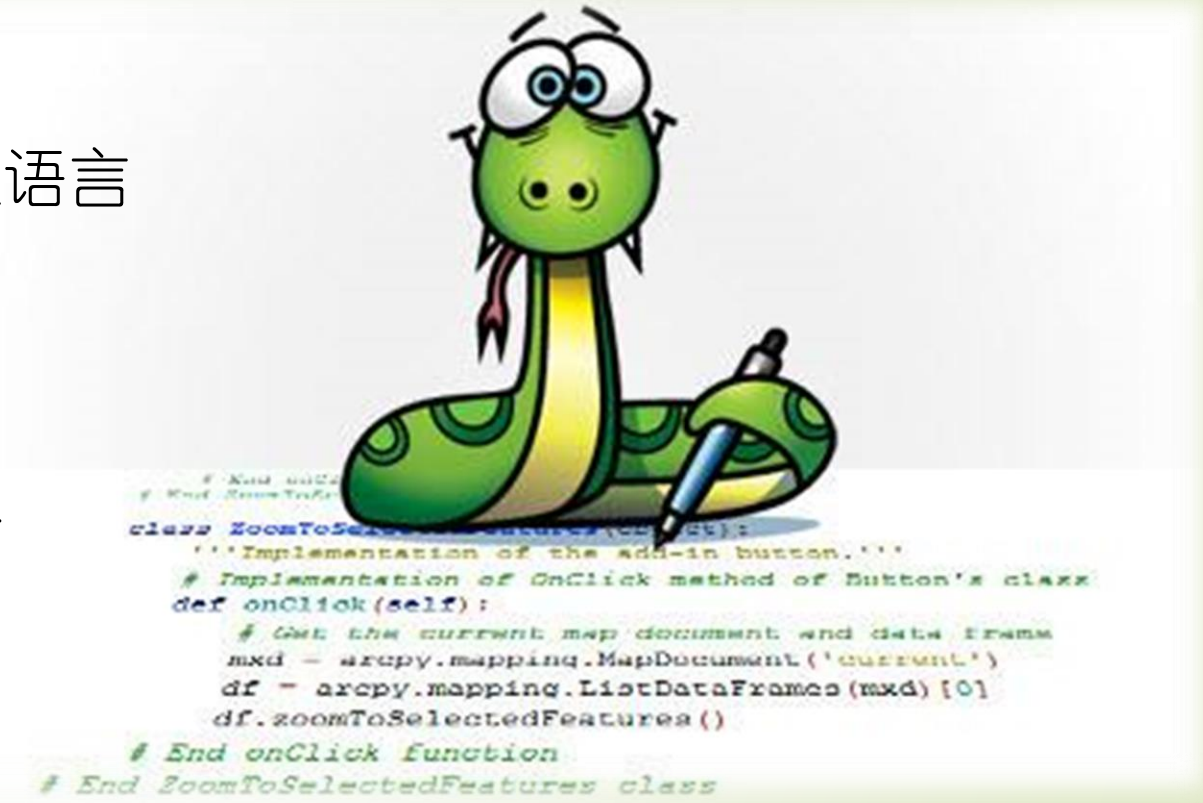
2022Python课程群

该二维码7天内(2月27日前)有效，重新进入将更新

Python语言基础 C01

Python

- 是解释型、函数式、面向对象编程语言
- 有很多功能强大的软件包
- 被广泛用于数据分析及处理
- 并不适合直接实现算法密集型任务
- 架构在各种类库上的胶水语言



Python之禅：

- Beautiful is better than ugly. 这个感觉会输给pascal
- Explicit is better than implicit. 于是类、对象的内容都是外部可见的？
- Simple is better than complex. 一切都是对象，按名字空间管理
- Complex is better than complicated. 比如多继承方案？
- Flat is better than nested. 类似Lambda演算的列表生成式
- Sparse is better than dense. 缩进~缩进~ 空行
- Readability counts. 列表生成式与高阶函数复合可以很酷很难读
- ……

Python很容易学么？

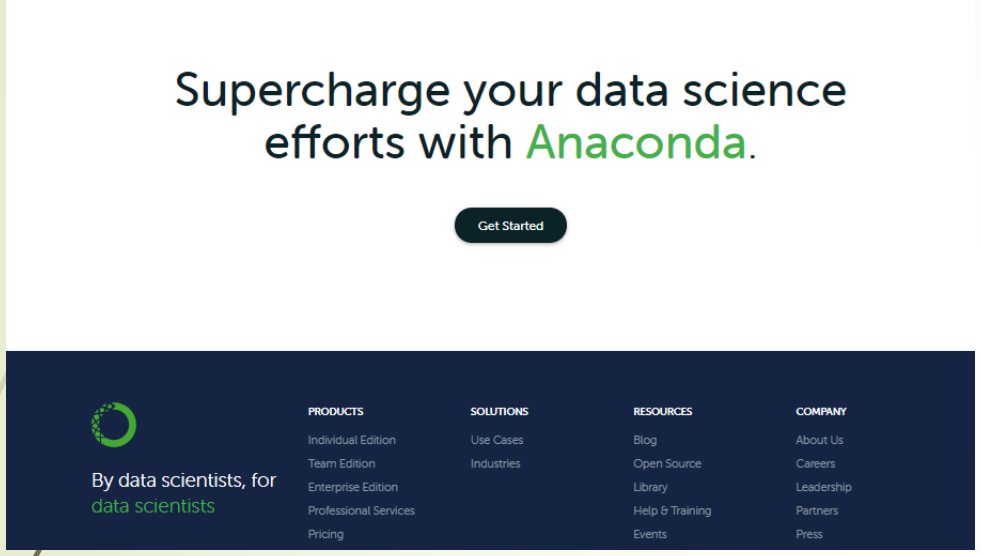
- 不，至少对新手来讲并不容易！
  但如果从学习一种语言架构的角度看，确实是简洁而优美！

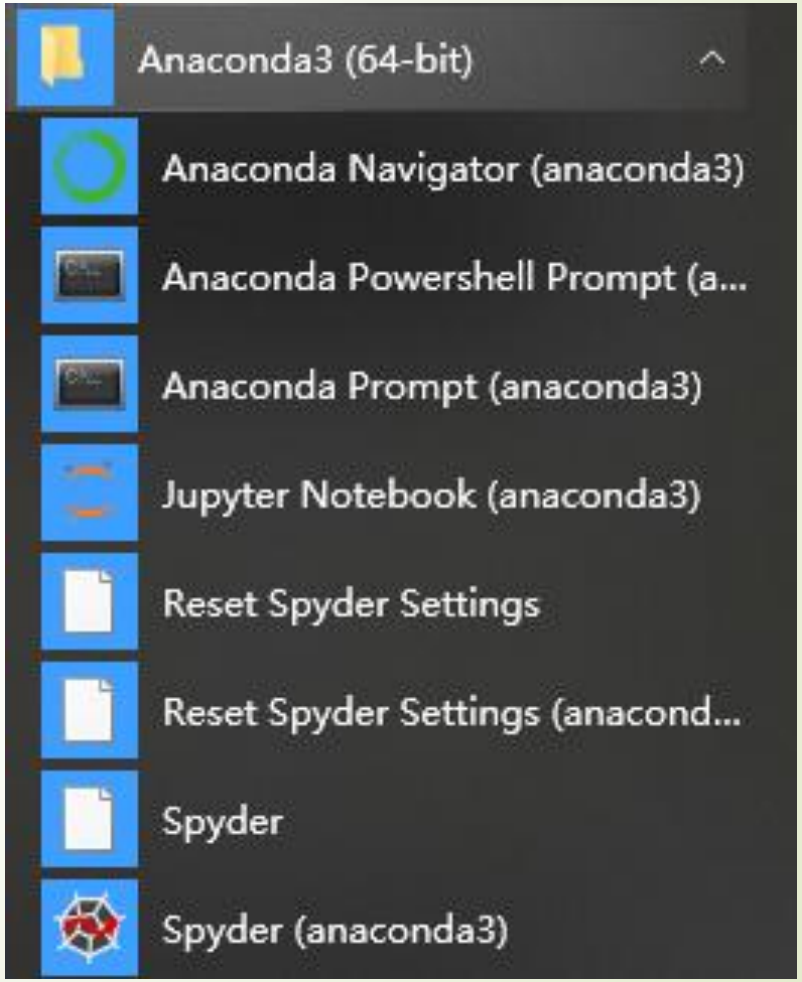
- 学会了Python语言并不等于可以熟练的用python进行编程实现。每个特定的领域，都需要具备相关的专业知识并熟悉软件环境。

本次课内容提要

- Anaconda环境与Jupytor notebook介绍
- Python基本类型与表达式
- Python的容器类
- Python流程控制：条件分支与循环
- 容器类与迭代器
- 函数、生成器

Anaconda


Supercharge your data science efforts with Anaconda.
Get Started
By data scientists, for data scientists
PRODUCTS
Individual Edition
Team Edition
Enterprise Edition
Professional Services
Pricing
SOLUTIONS
Use Cases
Industries
RESOURCES
Blog
Open Source
Library
Help & Training
Events
COMPANY
About Us
Careers
Leadership
Partners
Press


Anaconda3 (64-bit)
Anaconda Navigator (anaconda3)
Anaconda Powershell Prompt (a...
Anaconda Prompt (anaconda3)
Jupyter Notebook (anaconda3)
Reset Spyder Settings
Reset Spyder Settings (anacond...
Spyder
Spyder (anaconda3)

jupyter Jupyter Notebook简介 最后检查: 1 天前 (自动保存)

Logout

File Edit View Insert Cell Kernel Widgets Help 不可信 Python 3

运行 标记

什么是Jupyter Notebook

Jupyter Notebook是基于**网页**的用于交互计算的应用程序。其可被应用于全过程计算：开发、文档编写、运行代码和展示结果。

优势

- 轻便
- 交互性好，同时支持文本和代码
- 可以在服务器上部署，远程登录使用

劣势

- 功能简单，不适合大规模工程项目(可通过安装拓展包弥补部分功能)

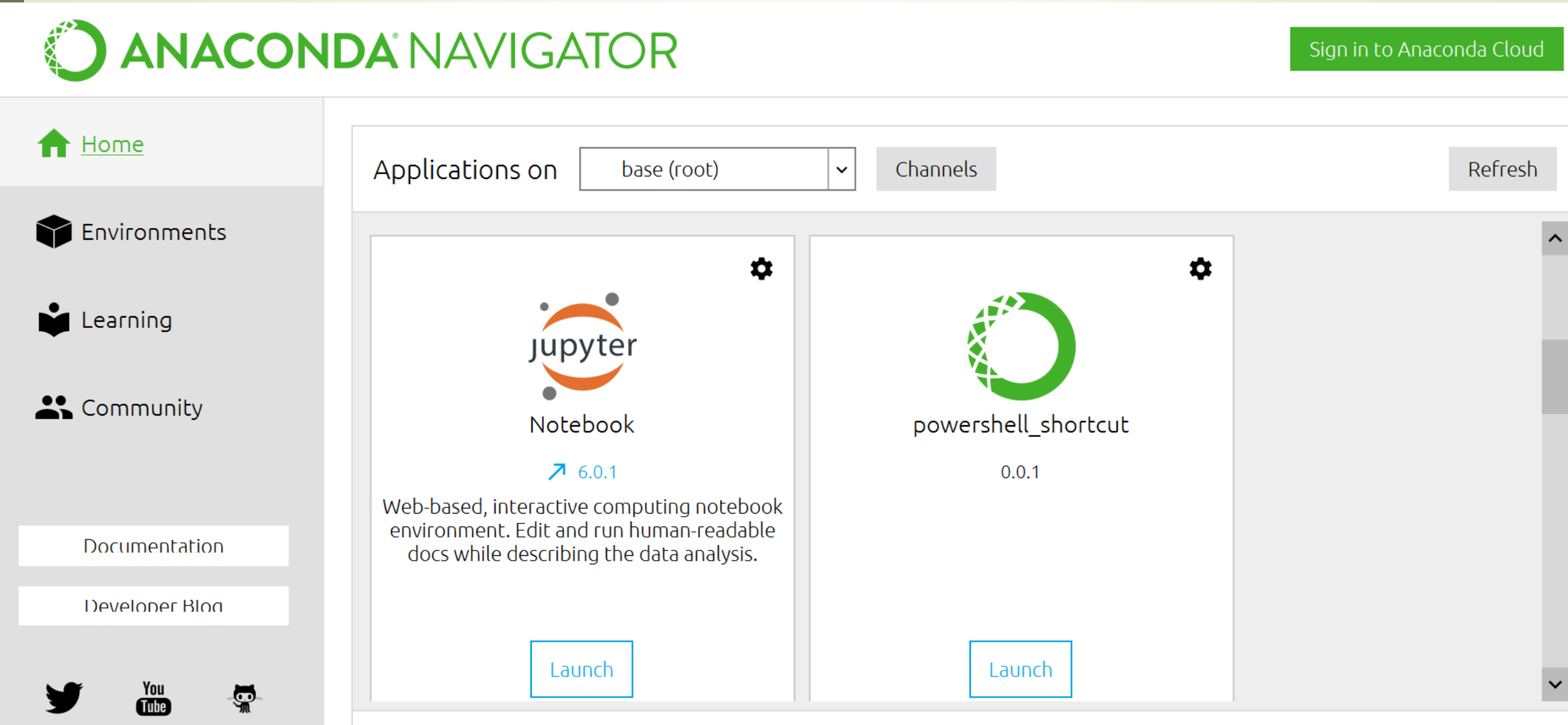
In [1]:
```
#演示交互性
string = '人生苦短，我学Python'
```

In [2]:
```
print(string)
```

```
人生苦短，我学Python
```

如何安裝

通过anaconda navigator默认安装（推荐）



命令行运行pip install jupyter

命令提示符

```
Microsoft Windows [版本 10.0.18362.592]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\lajiaojiang>pip install jupyter
```

如何运行

命令行运行jupyter notebook

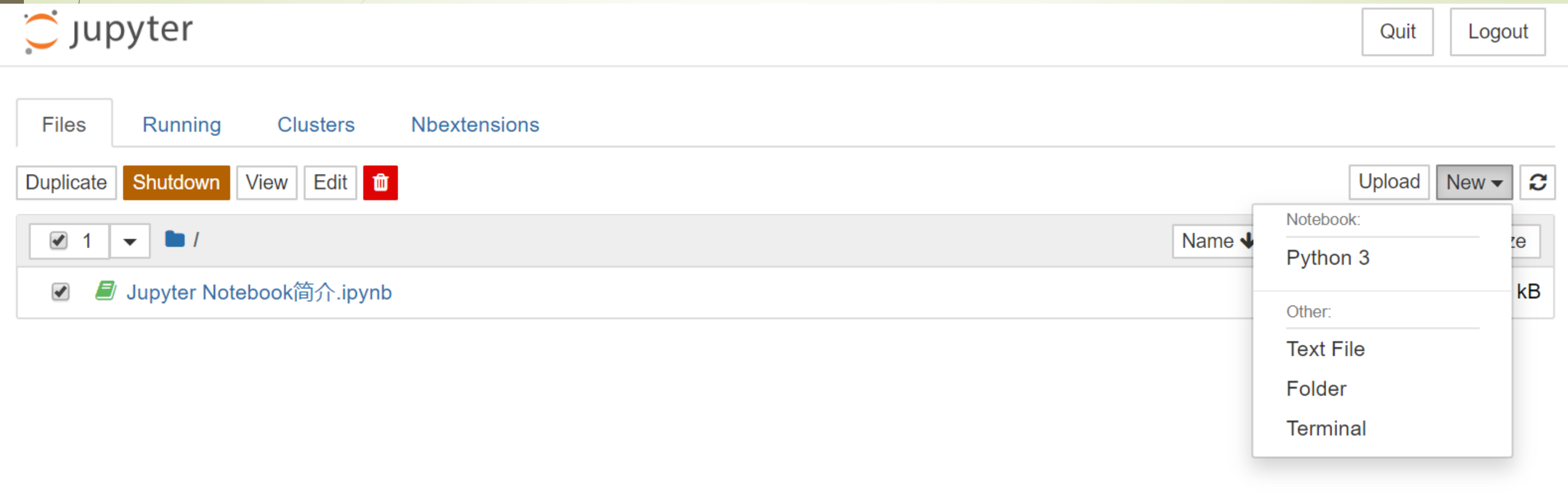
命令提示符 - jupyter notebook

```
Microsoft Windows [版本 10.0.18362.592]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\Users\lajiaojiang>jupyter notebook
[I 17:46:39.322 NotebookApp] [jupyter_nbextensions_configurator] enabled 0.4.1
[I 17:46:39.366 NotebookApp] JupyterLab extension loaded from D:\anaconda\lib\site-packages\jupyterlab
[I 17:46:39.366 NotebookApp] JupyterLab application directory is D:\anaconda\share\jupyter\lab
[I 17:46:39.370 NotebookApp] Serving notebooks from local directory: E:/Jupyter
[I 17:46:39.370 NotebookApp] The Jupyter Notebook is running at:
[I 17:46:39.370 NotebookApp] http://localhost:8888/?token=8b43fb6b40f87511a583e61df312717620612aaf1c8d3c87
[I 17:46:39.370 NotebookApp]  or http://127.0.0.1:8888/?token=8b43fb6b40f87511a583e61df312717620612aaf1c8d3c87
[I 17:46:39.370 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 17:46:39.414 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/lajiaojiang/AppData/Roaming/jupyter/runtime/nbserver-628-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=8b43fb6b40f87511a583e61df312717620612aaf1c8d3c87
     or http://127.0.0.1:8888/?token=8b43fb6b40f87511a583e61df312717620612aaf1c8d3c87
```

菜单介绍


jupyter
Quit
Logout
Files
Running
Clusters
Nbextensions
Duplicate
Shutdown
View
Edit
Upload
New
1
/
Name
Jupyter Notebook简介.ipynb
kB
Notebook:
Python 3
Other:
Text File
Folder
Terminal

jupyter Jupyter Notebook简介 最后检查: 1 天前 (自动保存)

Logout

File Edit View Insert Cell Kernel Widgets Help 可信的 Python 3

运行 标记

Cut Cells
Copy Cells
Paste Cells Above
Paste Cells Below
Paste Cells & Replace
Delete Cells
Undo Delete Cells

Split Cell
Merge Cell Above
Merge Cell Below

Move Cell Up
Move Cell Down

Edit Notebook Metadata

Find and Replace

Cut Cell Attachments

ter Notebook

是基于**网页**的用于交互计算的应用程序。其可被应用于全过程计算：开发、文档编写、运行代码和展示结果。

对支持文本和代码
上部署，远程登录使用

适合大规模工程项目(可通过安装拓展包弥补部分功能)

I

苦短，我学Python’

I

jupyter Jupyter Notebook简介 最后检查: 1 天前 (自动保存)

Logout

File Edit View Insert Cell Kernel Widgets Help 可信的 Python 3

运行

Interrupt
Restart
Restart & Clear Output
Restart & Run All
Reconnect
Shutdown

Change kernel

什么是Jupyter N

Jupyter Notebook是基于网 序。其可被应用于全过程计算：开发、文档编写、运行代码和展示结果。

优势

- 轻便
- 交互性好，同时支持文
- 可以在服务器上部署，远程登录使用

劣势

- 功能简单，不适合大规模工程项目(可通过安装拓展包弥补部分功能)

In [1]:

```
#演示交互性
string = '人生苦短，我学Python'
```

Notebook 编辑与常用快捷键

两种编辑模式：

内容编辑：

单元编辑：

ctrl+enter 运行当前cell；

shift+enter 运行当前cell并跳到下一个cell ；

ctrl+[向左缩进;

ctrl+] 向右缩进

安装拓展包

命令行运行

29

pip install jupyter_contrib_nbextensions

jupyter contrib nbextension install

```
Requirement already satisfied: pywin32>=1.0; sys_platform == "win32" in d:\anaconda\lib\site-packages (from jupyter-clie
nt>=5.3.1->notebook>=4.0->jupyter_contrib_nbextensions) (223)
Requirement already satisfied: python-dateutil>=2.1 in d:\anaconda\lib\site-packages (from jupyter-client>=5.3.1->notebo
ok>=4.0->jupyter_contrib_nbextensions) (2.8.0)
Requirement already satisfied: jedi>=0.10 in d:\anaconda\lib\site-packages (from ipython->jupyter-latex-envs>=1.3.8->jup
yter_contrib_nbextensions) (0.15.1)
Requirement already satisfied: backcall in d:\anaconda\lib\site-packages (from ipython->jupyter-latex-envs>=1.3.8->jupyt
er_contrib_nbextensions) (0.1.0)
Requirement already satisfied: colorama; sys_platform == "win32" in d:\anaconda\lib\site-packages (from ipython->jupyter
-latex-envs>=1.3.8->jupyter_contrib_nbextensions) (0.4.1)
Requirement already satisfied: pickleshare in d:\anaconda\lib\site-packages (from ipython->jupyter-latex-envs>=1.3.8->ju
pyter_contrib_nbextensions) (0.7.5)
Requirement already satisfied: prompt-toolkit<2.1.0,>=2.0.0 in d:\anaconda\lib\site-packages (from ipython->jupyter-late
x-envs>=1.3.8->jupyter_contrib_nbextensions) (2.0.10)
Requirement already satisfied: attrs>=17.4.0 in d:\anaconda\lib\site-packages (from jsonschema!=2.5.0,>=2.4->nbformat>=4
.4->nbconvert>=4.2->jupyter_contrib_nbextensions) (19.2.0)
Requirement already satisfied: pyrsistent>=0.14.0 in d:\anaconda\lib\site-packages (from jsonschema!=2.5.0,>=2.4->nbform
at>=4.4->nbconvert>=4.2->jupyter_contrib_nbextensions) (0.15.4)
Requirement already satisfied: parso>=0.5.0 in d:\anaconda\lib\site-packages (from jedi>=0.10->ipython->jupyter-latex-en
vs>=1.3.8->jupyter_contrib_nbextensions) (0.5.1)
Requirement already satisfied: wcwidth in d:\anaconda\lib\site-packages (from prompt-toolkit<2.1.0,>=2.0.0->ipython->jup
yter-latex-envs>=1.3.8->jupyter_contrib_nbextensions) (0.1.7)

C:\Users\lajiaojiang>jupyter contrib nbextension install
```

Files | Running | Clusters | Nbextensions

Configurable nbextensions

- [ ] disable configuration for nbextensions without explicit compatibility (they may break your notebook environment, but can be useful to show for nbextension development)

filter: by description, section, or tags

- [ ] (some) LaTeX environments for Jupyter
- [ ] Autopep8
- [ ] Cell Filter
- [x] Codefolding
- [ ] Collapsible Headings
- [ ] datestamper
- [ ] Execution Dependencies
- [ ] Export Embedded HTML
- [ ] Help panel
- [ ] Hide input all
- [x] Hinterland
- [x] jupyter-js-widgets/extension
- [ ] Limit Output
- [ ] 2to3 Converter
- [ ] AutoSaveTime
- [ ] Code Font Size
- [ ] Codefolding in Editor
- [ ] Comment/Uncomment Hotkey
- [ ] Equation Auto Numbering
- [ ] Exercise
- [ ] Freeze
- [ ] Hide Header
- [ ] Highlight selected word
- [ ] Initialization cells
- [ ] Keyboard shortcut editor
- [ ] Live Markdown Preview
- [ ] AddBefore
- [ ] Autoscroll
- [x] Code prettify
- [ ] CodeMirror mode extensions
- [x] contrib_nbextensions_help_item
- [ ] ExecuteTime
- [ ] Exercise2
- [ ] Gist-it
- [ ] Hide input
- [ ] highlighter
- [ ] isort formatter
- [ ] Launch QTConsole
- [ ] Load TeX macros

修改默认配置

命令行运行jupyter notebook --generate-config生成配置文件

命令提示符 - jupyter notebook --generate-config

```
Requirement already satisfied: python-dateutil>=2.1 in d:\anaconda\lib\site-packages (from jupyter-client>=5.3.1->notebo
ok>=4.0->jupyter_contrib_nbextensions) (2.8.0)
Requirement already satisfied: jedi>=0.10 in d:\anaconda\lib\site-packages (from ipython->jupyter-latex-envs>=1.3.8->jup
yter_contrib_nbextensions) (0.15.1)
Requirement already satisfied: backcall in d:\anaconda\lib\site-packages (from ipython->jupyter-latex-envs>=1.3.8->jupyt
er_contrib_nbextensions) (0.1.0)
Requirement already satisfied: colorama; sys_platform == "win32" in d:\anaconda\lib\site-packages (from ipython->jupyter
-latex-envs>=1.3.8->jupyter_contrib_nbextensions) (0.4.1)
Requirement already satisfied: pickleshare in d:\anaconda\lib\site-packages (from ipython->jupyter-latex-envs>=1.3.8->ju
pyter_contrib_nbextensions) (0.7.5)
Requirement already satisfied: prompt-toolkit<2.1.0,>=2.0.0 in d:\anaconda\lib\site-packages (from ipython->jupyter-late
x-envs>=1.3.8->jupyter_contrib_nbextensions) (2.0.10)
Requirement already satisfied: attrs>=17.4.0 in d:\anaconda\lib\site-packages (from jsonschema!=2.5.0,>=2.4->nbformat>=4
.4->nbconvert>=4.2->jupyter_contrib_nbextensions) (19.2.0)
Requirement already satisfied: pyrsistent>=0.14.0 in d:\anaconda\lib\site-packages (from jsonschema!=2.5.0,>=2.4->nbform
at>=4.4->nbconvert>=4.2->jupyter_contrib_nbextensions) (0.15.4)
Requirement already satisfied: parso>=0.5.0 in d:\anaconda\lib\site-packages (from jedi>=0.10->ipython->jupyter-latex-en
vs>=1.3.8->jupyter_contrib_nbextensions) (0.5.1)
Requirement already satisfied: wcwidth in d:\anaconda\lib\site-packages (from prompt-toolkit<2.1.0,>=2.0.0->ipython->jup
yter-latex-envs>=1.3.8->jupyter_contrib_nbextensions) (0.1.7)

C:\Users\lajiaojiang>jupyter notebook --generate-config
Overwrite C:\Users\lajiaojiang\.jupyter\jupyter_notebook_config.py with default config? [y/N]
```

```
## Dict of Python modules to load as notebook server extensions.Entry values can
#  be used to enable and disable the loading ofthe extensions. The extensions
#  will be loaded in alphabetical order.
#c.NotebookApp.nbserver_extensions = {}

## The directory to use for notebooks and kernels.
c.NotebookApp.notebook_dir = 'E:/Jupyter'

## Whether to open in a browser after starting. The specific browser used is
```

进入python的内容：

- 基本数据类型
- 表达式
- 流程控制

Python的基础数据类型与算数表达式

- 整数、浮点数、（复数）
- 字符串

整数

```
i1 = 9999999999999999999999999999999    # 整数常量和变量
i2 = 101
i1 + i2                                 # 可以形象理解为用字符串实现的大整数运算
```

```
10000000000000000000000000100
```

```
type(i1)
```

```
int
```

```
print(i2 == 101)   # 整数相等的判定通过hash操作实现
```

```
True
```

整数支持位运算：

x | y	x 和 y 按位 *或*	负数的二进制采用有符号位补码表示
x ^ y	x 和 y 按位 *异或*	
x & y	x 和 y 按位 *与*	
x << n	x 左移 *n* 位	负数不支持移位运算
x >> n	x 右移 *n* 位	
~x	x 逐位取反	

浮点数

```
num1 = 101.1                    # 64位浮点数编码
num2 = 100.000000000000001      # 会产生精度误差
print(num1-num2)
num1 = 101.000000000000001
num2 = 100.0000000000000001     # 会有截断误差
print(num1-num2)
num2 = 1E-200                   # 科学计数法
print(num1 / num2)
```

```
1.0999999999999943
1.0
1.01e+202
```

算术表达式运算：

表达式计算

```
(10 + 12) // 4 % 3   # 计算器  // 表示整除  % 取模
```

2

变量类型？

变量类型不需要预先定义

```
a = 2 + 10                # 变量名直接使用，解释性语言特色
print("a=", (a + 1) /3)   # 表达式做函数参数
b = 3**2 + 1.3            # ** 乘方运算
print("b=", b)
```

```
a= 4.333333333333333
b= 10.3
```

Integers可以任意长度，由内存大小决定

floating point number精确到15位十进制小数

内置算数运算

	结果	注释	完整文档
`x + y`	*x* 和 *y* 的和		
`x - y`	*x* 和 *y* 的差		
`x * y`	*x* 和 *y* 的乘积		
`x / y`	*x* 和 *y* 的商		
`x // y`	*x* 和 *y* 的商数	(1)	
`x % y`	`x / y` 的余数	(2)	
`-x`	*x* 取反		
`+x`	*x* 不变		
`abs(x)`	*x* 的绝对值或大小		`abs()`
`int(x)`	将 *x* 转换为整数	(3)(6)	`int()`
`float(x)`	将 *x* 转换为浮点数	(4)(6)	`float()`
`complex(re, im)`	一个带有实部 *re* 和虚部 *im* 的复数。*im* 默认为0。	(6)	`complex()`
`c.conjugate()`	复数 *c* 的共轭		
`divmod(x, y)`	`(x // y, x % y)`	(2)	`divmod()`
`pow(x, y)`	*x* 的 *y* 次幂	(5)	`pow()`
`x ** y`	*x* 的 *y* 次幂	(5)	

- https://docs.python.org/zh-cn//3/library/stdtypes.html#numeric-types-int-float-long-complex

字符串

```
'''
单引号-双引号 表达字符串常量
连续三个引号开启一段注释
'''
st1 = '"Hello"'
st2 = "Python!"
print(st1+ ' '+ st2)
```

```
"Hello" Python!
```

```
str = 'Do U have a nice vacation?'
str += '\n' + "No I don\'t."        # 反斜杠转义字符
print (str)
```

字符流与简单输入输出操作：

```
print("Hello, I'm iPython!")

# Input, assignment
name = input('What is your name?\n')
print('Hi, %s.' % name)
```

控制台输出

```
Hello, I'm iPython!

What is your name?
Jeff Hu
```

交互输入

```
Hi, Jeff Hu.
```

条件表达式与分支（if-elif）语句

- 布尔类型与逻辑表达式
  - not 非； and 与； or 或
- 比较运算
- 分支语句

布尔类型与布尔达式运算：

Booleans 布尔类型

```
t, f = True, False
print (type(t)) # Prints "<type 'bool'>"
```

多赋值操作，

```
<class 'bool'>
```

```
print (t and f)    # Logical AND;
print (t or f)     # Logical OR;
print (not t)      # Logical NOT;
print (t != f)     # Logical XOR;
```

```
False
True
False
True
```

布尔表达式以及短路运算

```
a = 0     # 类型可以自定义 __bool__()返回的布尔值，len() = 0 对应 false，其他True
print(not a)
```

```
True
```

```
s1 = '123'
s2 = s1 or 'abc'   # or, and 为短路运算 不强制返回布尔值
print(s2)

s1 = ''
s2 = s1 or 'abc'
s2
```

```
123

'abc'
```

比较（关系）运算

<	严格小于
<=	小于或等于
>	严格大于
>=	大于或等于
==	等于
!=	不等于
is	是同一个对象（标识）
is not	不是同一个对象（标识）

比较（关系）运算与分支流程控制：


```
x = int(input("Please enter an integer: "))
if x < 0:
    x = 0
    print('Negative changed to zero')
elif x == 0:
    print('Zero')
elif x%2 == 1:
    print('Single')
else:
    print('Other')
    print("in the block")
print('out of the block')
```

冒号：开启一个程序块

缩进-程序块延续

Tab或四个空格

语句间隔是直接换行，句尾没有分号

```
Please enter an integer: 11
Single
out of the block
```

Python的复合类型

- 序列类型：
  - 列表 List，元组tuple，range，字符串 string
  - 迭代器，生成器
- 集合 set
- 词典 dict

列表 List

- 对应物理概念：系列、序列
- 特点：类型兼容性好、结构灵活、可容纳复合类型
  引用主导，数据隔离性不好，容易出错

List（列表）

不同类型能混在一起

```
a = [1, 2, 'aaa', 1.23]    # 可以大体理解为一个泛类型的数据对象数组

print('a = ',a)
print("a[1] =", a[1], "; a[2] = ", a[2])
print('len(a) =', len(a))
```

一切皆是引用！

双引号单引号对都表示字符串

```
a =  [1, 2, 'aaa', 1.23]
a[1] = 2 ; a[2] =  aaa
len(a) = 4
```

List对象常用方法：增、删、查、改

添加元素 append()

```
a = []          # 空列表

a.append(20.5)
a.append(30)
a.append('hello wrold!')
print(a)
```

尾部添加元素

```
[20.5, 30, 'hello wrold!']
```

插入元素 insert()

```
a.insert(2,'beautiful')
print(a)

a.insert (-1,24)
print(a)
```

第3个位置（下标从0开始）处插入元素

倒数第2个位置处插入元素

```
[20.5, 30, 'beautiful', 'hello wrold!']
[20.5, 30, 'beautiful', 24, 'hello wrold!']
```

List对象常用方法：删、改

```
a = [1, 2, 3, 4]     #新列表

del a[2]    # 删除下标为2的元素
a[2] = 10
print(a)

a.clear()   # 清空列表
print(a)

del a       # 删除列表对象
print(a)
```

按下标直接读写访问

```
[1, 2, 10]
[]

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-34-5cac9423a08a> in <module>()
      9
     10 del a       # 删除列表对象
---> 11 print(a)

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

判断是否在list中

```
a = [1, 3, 5, 7, 9]
print(1 in a)

d = a.index(5)
print ('a.index(5) = ', d)
# print ('a.index(4) = ', a.index(4)) # ValueError: 4 is not in list
```

```
True
a.index(5) =  2
```

```
#### count()
```

```
fruits = ['orange', 'apple', ['apple', 'banana'], 'kiwi', 'apple', 'banana
fruits.count('apple')
```

```
2
```

Tuple（元组）

- 和List不同在于，一旦初始化后不可更改
- 元组虽不可变，但其中嵌套元素可变（因为本身只是引用）

```
t = (1,2)

print(t)

print(t[1])
```

```
(1, 2)
2
```

```
t = ('a', 'b', ['A', 'B'])

t[2][0] = 'X'

print(t)

# t[0]  = 'd'   # this may cause an error
```

引用了一个list对象

```
('a', 'b', ['X', 'B'])
```

Tuple的运算


```
#元组展开
a = tuple('1234')   # 展开为一个tuple
print(a)

c, d,*_ = a    # unpack
c,d
```

```
('1', '2', '3', '4')
('1', '2')
```

```
# 元组 合并

t2 = (3,4)
t = (1,2,3)

print(t+t2)
```

```
(1, 2, 3, 3, 4)
```

List切片

```
a = [1, 2, 3, 4, 5]

b = a[1:3]
c = a[1:-1]
d = a[1:]

print(b)
print(c)
print(d)
```

```
[2, 3]
[2, 3, 4]
[2, 3, 4, 5]
```

```
d = a[::2]

d
```

```
[1, 3, 5]
```

str切片

- 字符串不可更改内容

```
string = "ursoooo cute!"

string[1:4]
```

```
'rso'
```

tuple切片

```
t = (1, 2, 3, 4, 5, 6)

t1 = t[1:3]

t1
```

```
(2, 3)
```

string对象常用方法：string对象可读，可下标访问，但不可修改

```
str = 'abcd'
print (str[1])
str[1] = 'q'
```

```
b

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-75-5b315d8a9b2a> in <module>()
      1 str = 'abcd'
      2 print (str[1])
----> 3 str[1] = 'q'

TypeError: 'str' object does not support item assignment
```

string对象常用方法： https://docs.python.org/3/library/stdtypes.html#string-methods

```
str = 'Do U have a nice sleep?'
str += '\n' + "No I don\'t."
print (str)
```

反斜杠的用法和C语言一致

```
Do U have a nice sleep?
No I don't.
```

```
s = "    11123123abc"      #str.strip([chars]) 去前导空格以及特殊符号
s = s.strip(' ')
print(s)
print(s.strip('1'))

print('s.find = ',s.find('23'))
i = s.find('23', 4)        # 从特定下标开始定位串
print (i, s.count('11'))
```

这里s的前导‘1’并没有真正去除，表达式是一个新对象

结果说明是find()函数的复用

```
11123123abc
23123abc
s.find =  3
6 1
```

str.split(sep=None, maxsplit=-1)

- Return a **list** of the words in the string, using sep as the delimiter string. If maxsplit is given, at most maxsplit splits are done (thus, the list will have at most maxsplit+1 elements). If maxsplit is not specified or -1, then there is no limit on the number of splits (all possible splits are made).

```
s = 'ab,cde,fgh,ijk'

print(s.split(','))  # 切分开逗号分割的串
print(s.split(',', maxsplit= 2))
```

```
['ab', 'cde', 'fgh', 'ijk']
['ab', 'cde', 'fgh,ijk']
```

str.join(iterable)

- Join a list of words into a string.

```
delimiter = ':'

mylist = ['Brazil', 'Russia', 'India', 'China']

print(delimiter.join(mylist))
```

```
Brazil:Russia:India:China
```

迭代器与循环

- Iter类型

Range(start, end, step) 左闭右开

```
sum = 0
for i in range(0, 101):
    sum += i                    # += 运算sum必须先定义
    j = i
print(sum, j)
```

冒号+缩进 声明语句块

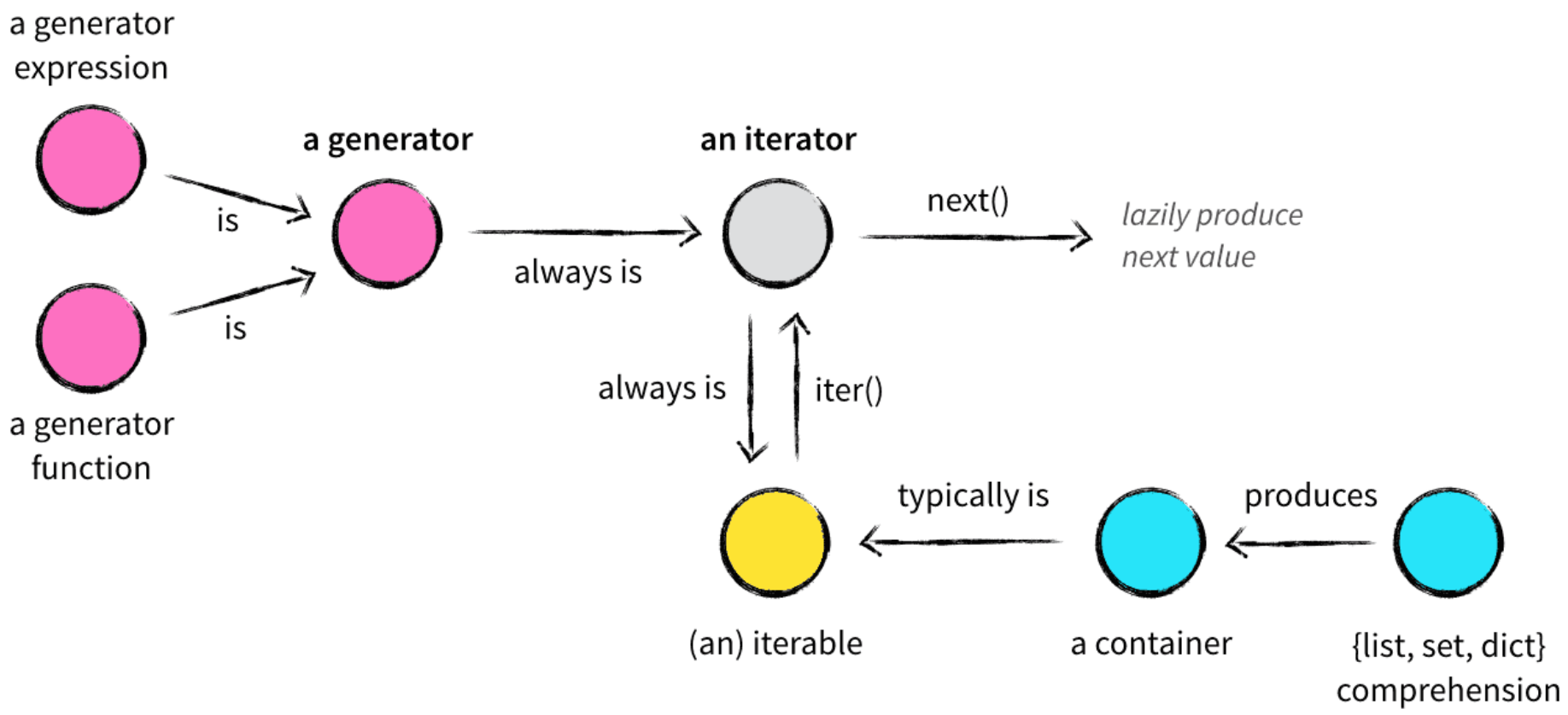
取消缩进，退出循环体

```
5050 100
```

语句块（for）并不产生独立的命名空间

生成器、迭代器

- 在Python中，依托循环结构与生成表达式惰性产生数据的对象被称为生成器：**generator**
- Python的**Iterator**对象表示的是一个数据流，Iterator对象可以被next()函数调用并不断返回下一个数据



生成器


```python
g = (x * x for x in range(4))

print(g)
# print(g[3])  $'generator' object is not subscriptable
```

```
<generator object <genexpr> at 0x0000019DEB305E58>
```

```python
for n in g:
    print(n)
```

```
0
1
4
9
```

迭代器：

迭代器不但可以作用于for循环，还可以被next()函数不断调用并返回下一个值，直到最后抛出StopIteration错误表示无法继续返回下一个值。

```
x = [1, 2, 3]

y = iter(x)

print(next(y),next(y))
```

```
1 2
```

```
type(x)
```

```
list
```

```
type(y)
```

```
list_iterator
```

```
x = [1, 2, 3]
x1 =[]          # if x1 not define
y = iter(x)

for i in y:
    x1.append(i)
print (x1)
```

```
[1, 2, 3]
```

How for loop actually works

```
# create an iterator object from that iterable
iter_obj = iter(iterable)

# infinite loop
while True:
    try:
        # get the next item
        element = next(iter_obj)
        # do something with element
    except StopIteration:
        # if StopIteration is raised, break from loop
        break
```

课后作业见课程网