计算机视觉

张健

https://villa.jianzhang.tech/

信息工程学院
北京大学深圳研究生院

2022.09.14

作业情况

北京大学 PEKING UNIVERSITY

Open set semi-supervised learning	多模态
Deepfake	Human Object Interaction Detection
基于骨架的人体行为识别	神经视觉传感器的高效处理骨干网络
End-to-end Human-Object Interaction Detection	点云分类与分割
神经辐射场	基于diffusion模型的图像生成
Vision Transformer	Temporal Grounding
Video super resolution	few-shot image classification
弱监督物体定位	人脸重建
三维人脸合成及其动态化	Multi-view stereo
Medical Report Generation / Medical Image Caption	Deepfakes Detection
基于Transformer原理的图像分类	端到端图像压缩
医学报告生成	AI art
深度估计	3D human pose estimation, motion prediction
Object Detection	图像修复/编辑
人脸识别	行人重识别
graph convolutions in point cloud analysis	脉冲神经网络计算机视觉
三维人体合成	图像与视频压缩
Point Cloud	基于变分自编码器的端到端图像压缩
三维重建	网络安全与图像处理
Temporal Action Localization	多模态情绪识别
模型与数据结合的图像复原	Vision-Language Pretraining
端到端的点云压缩	大模型预训练
Image compression	人脸迁移

预热-自我介绍（一分钟）

- 姓名
- 班级，家乡
- 实验室，导师
- 研究方向
- 本课程所选择课题及课程目标

班级：计算机视觉

2. 基础知识

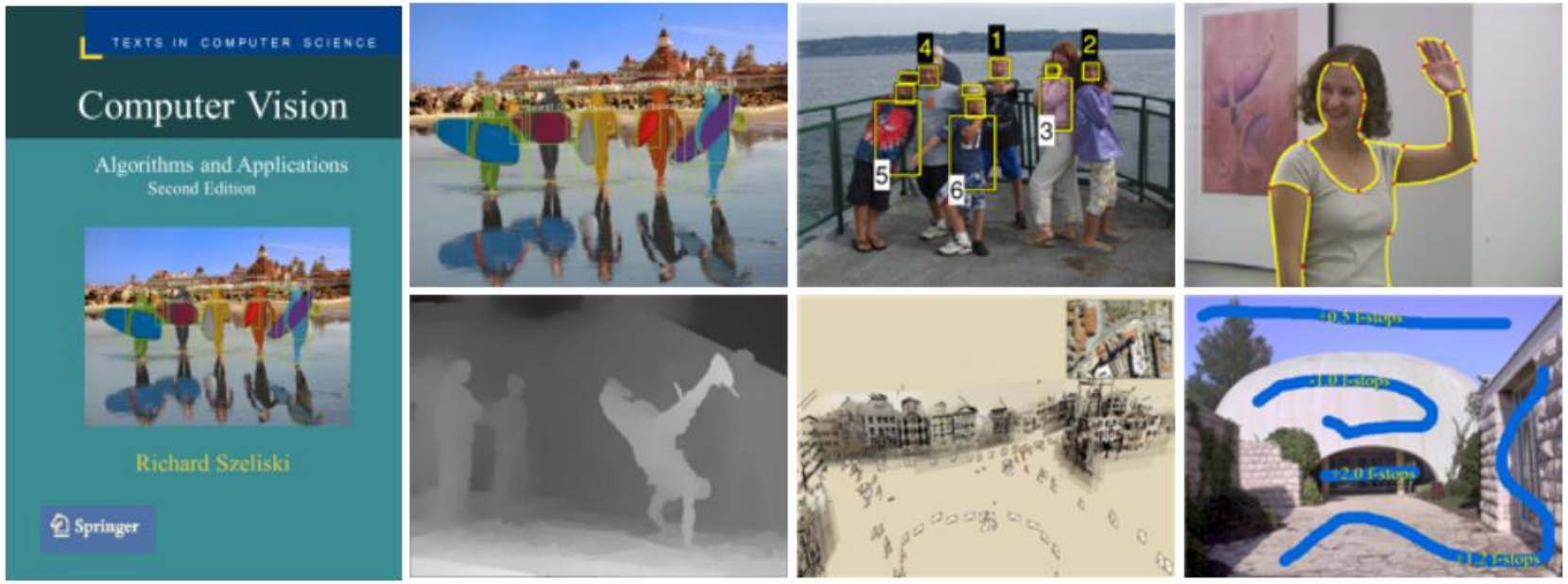
基础知识

- 平台工具安装使用
  - Anaconda (Jupyter Notebook, Jupyter Lab)
  - Python基础

推荐书籍

Computer Vision: Algorithms and Applications, 2nd ed.

© 2020 Richard Szeliski, Facebook

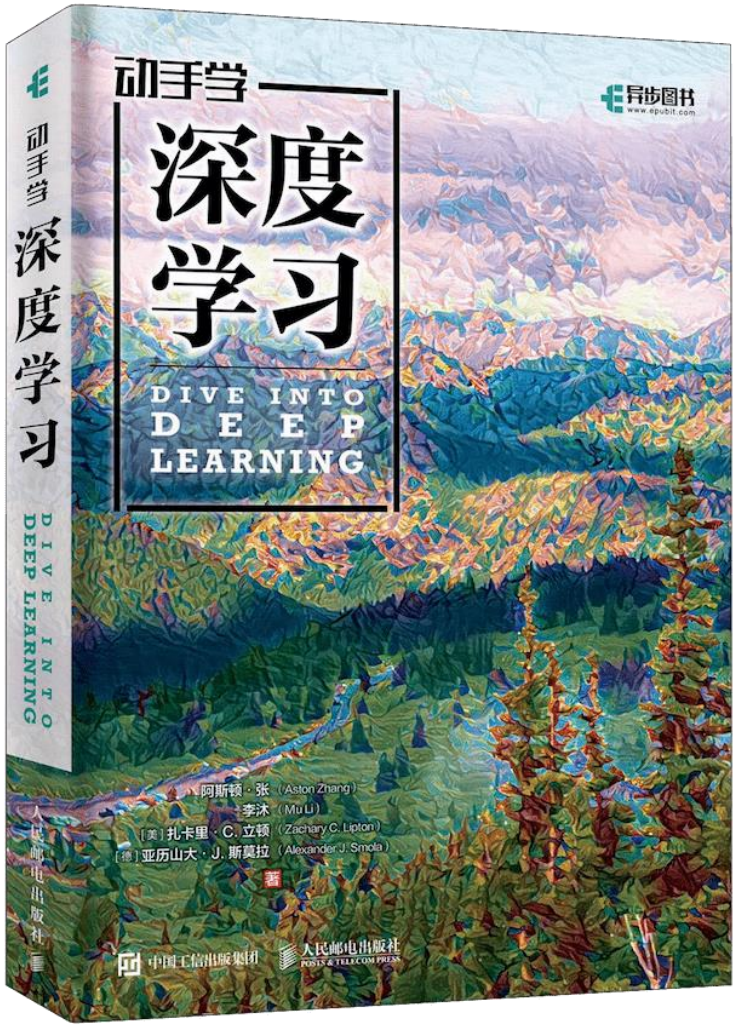


http://szeliski.org/Book/

推荐书籍

https://www.deeplearningbook.org/

推荐书籍

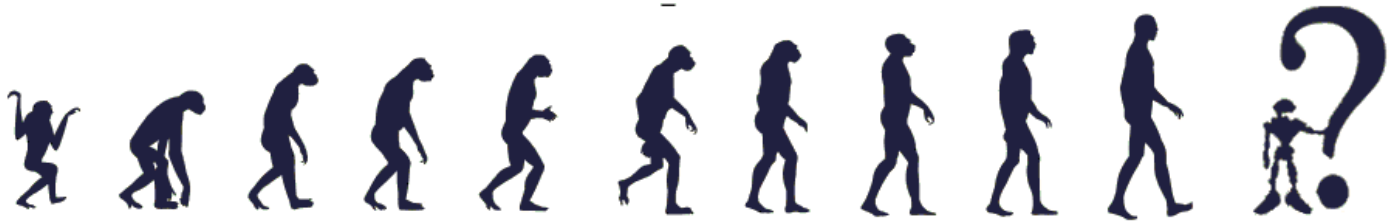


https://zh.d2l.ai/

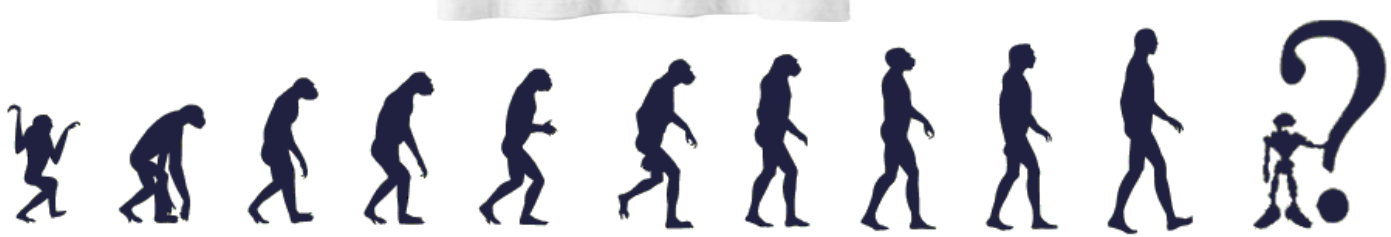
基础知识

- 平台工具安装使用
  - Anaconda (Jupyter Notebook, Jupyter Lab)
  - Python (Numpy, CV2, Matplotlib)
- 图像处理背景知识
  - 图像表示
  - 颜色空间 (RGB, HSI, YCbCr)
  - 基本操作（像素处理，缩放裁剪，滤波卷积）

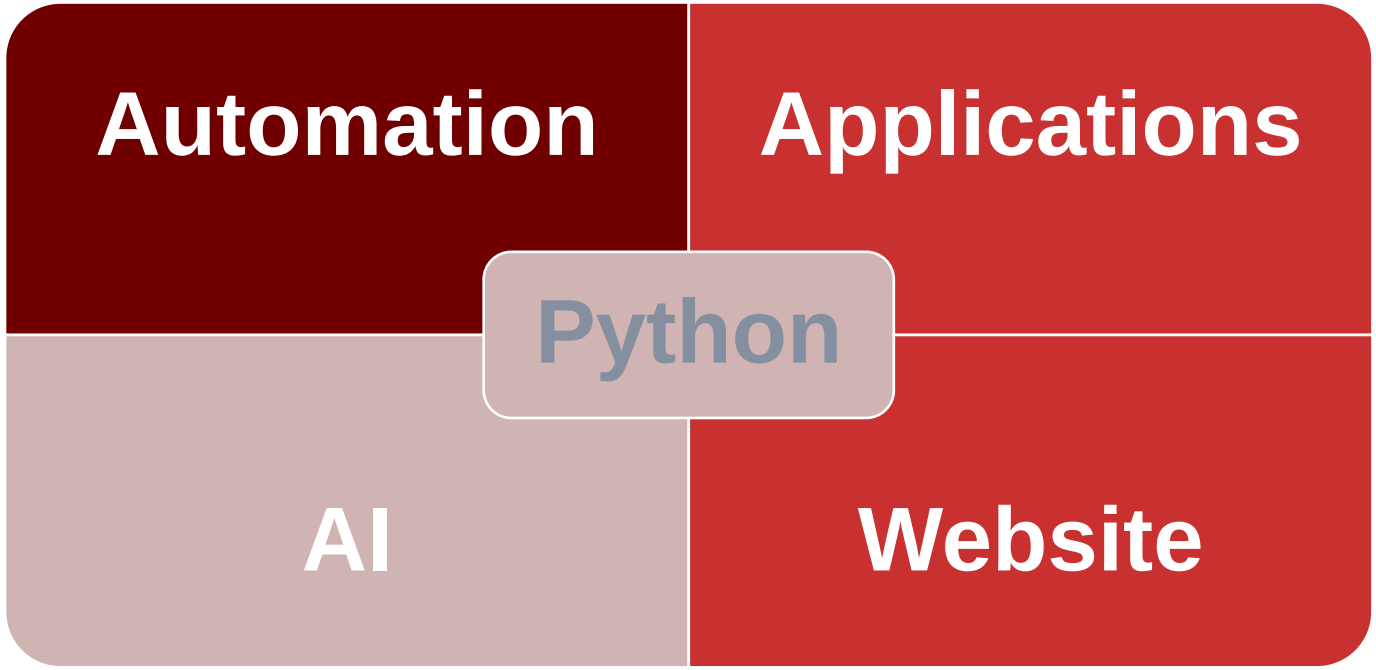
平台工具安装使用



平台工具安装使用

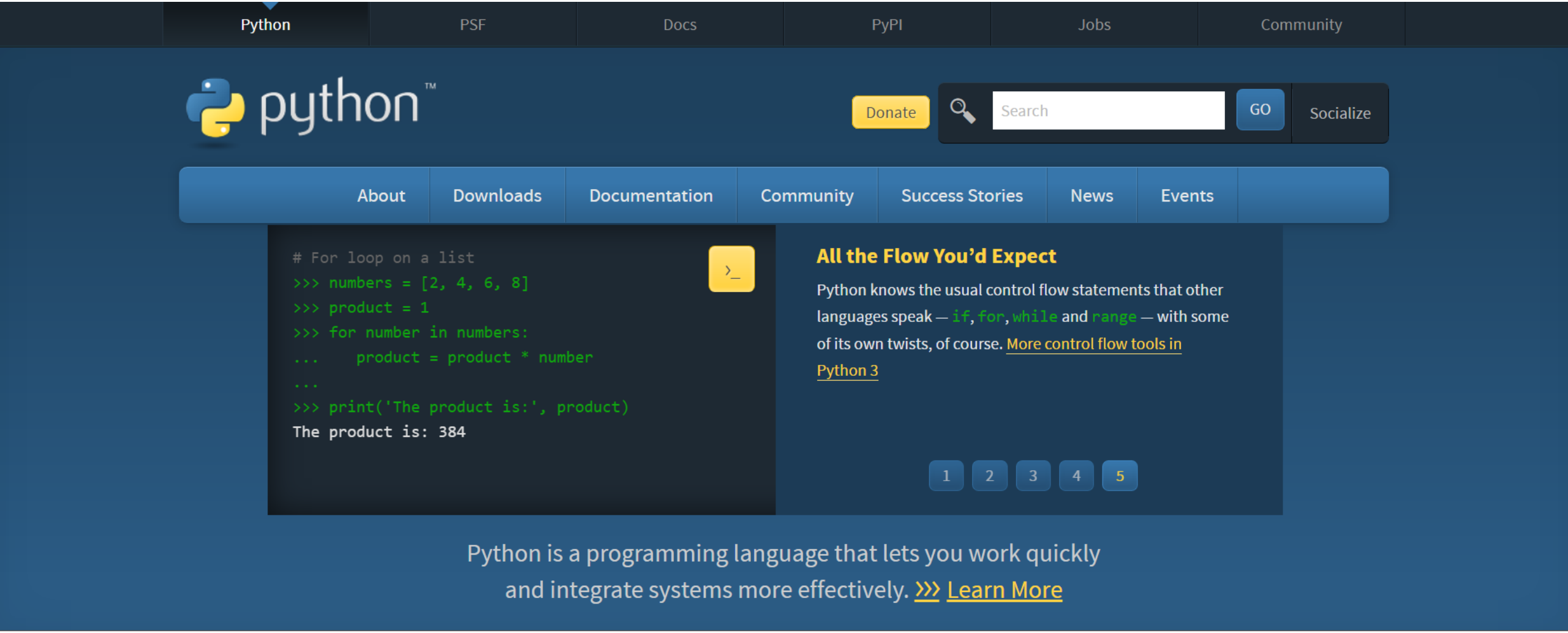


Python是什么?



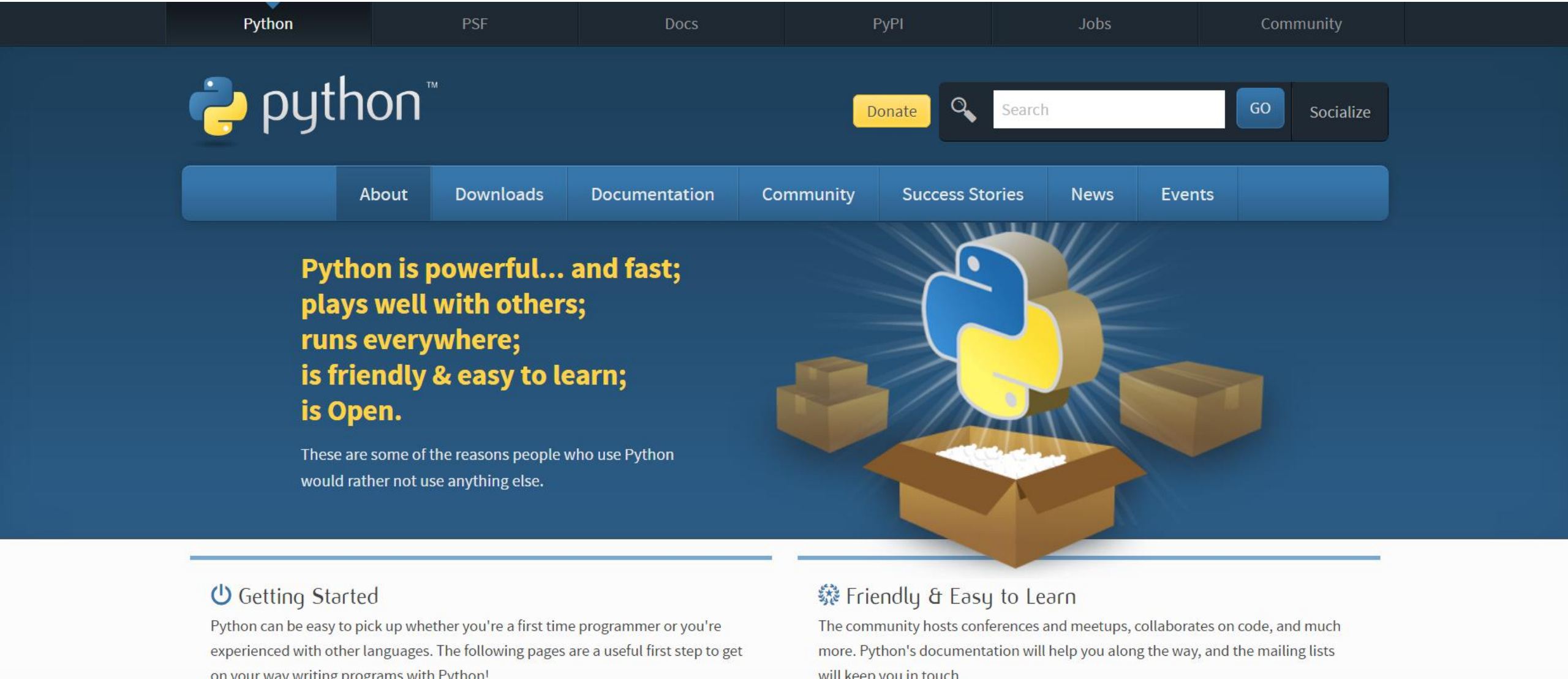
Python是什么?

https://www.python.org/

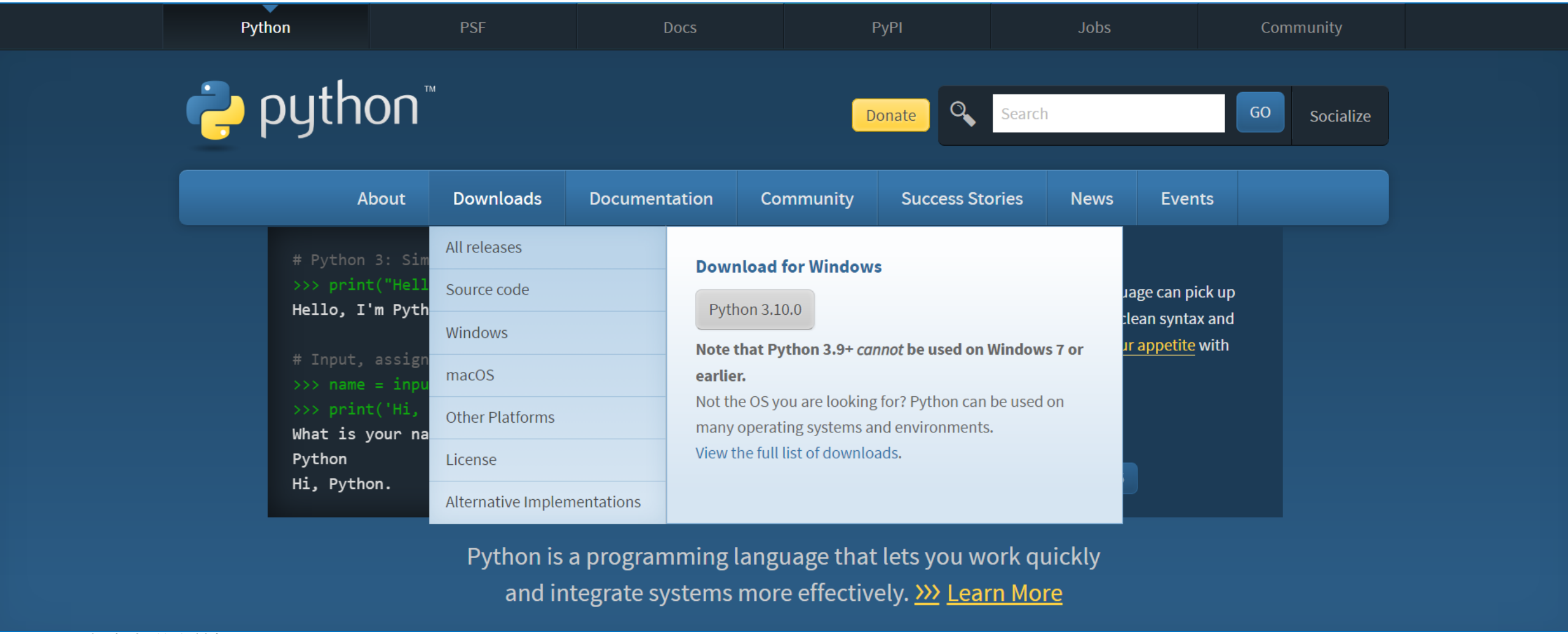


Python是什么?

https://www.python.org/



Python是什么?


Python
PSF
Docs
PyPI
Jobs
Community
python™
Donate
Search
GO
Socialize
About
Downloads
Documentation
Community
Success Stories
News
Events
# Python 3: Sim
>>> print("Hell
Hello, I'm Pyth
# Input, assign
>>> name = inpu
>>> print('Hi,
What is your na
Python
Hi, Python.
All releases
Source code
Windows
macOS
Other Platforms
License
Alternative Implementations
Download for Windows
Python 3.10.0
Note that Python 3.9+ cannot be used on Windows 7 or earlier.
Not the OS you are looking for? Python can be used on many operating systems and environments.
View the full list of downloads.
uage can pick up
clean syntax and
ur appetite with
Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Python是什么?


Python
PSF
Docs
PyPI
Jobs
Community
python™
GO
Socialize
About
Events
# Python 3: Fibon
>>> def fib(n):
>>> a, b = 0,
>>> while a <
>>> print
>>> a, b
>>> print()
>>> fib(1000)
0 1 1 2 3 5 8 13
functions.
ents, keyword
More about
Python 3.10.0 (64-bit) Setup
Install Python 3.10.0 (64-bit)
Select Install Now to install Python with default settings, or choose
Customize to enable or disable features.
Install Now
C:\Users\Jian\AppData\Local\Programs\Python\Python310
Includes IDLE, pip and documentation
Creates shortcuts and file associations
Customize installation
Choose location and features
python for windows
Install launcher for all users (recommended)
Add Python 3.10 to PATH
Cancel
Python is a programming language that lets you work quickly
and integrate systems more effectively. >>> Learn More

平台

https://www.anaconda.com/

ANACONDA. Products Pricing Solutions Resources Blog Company Get Started

Individual Edition

Your data science toolkit

With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.

Anaconda Individual Edition

Download

For Windows

Python 3.8 • 64-Bit Graphical Installer • 477 MB

Get Additional Installers

平台

https://www.anaconda.com/

Open Source

Anaconda Individual Edition is the world's most popular Python distribution platform with over 25 million users worldwide. You can trust in our long-term commitment to supporting the Anaconda open-source ecosystem, the platform of choice for Python data science.

Conda Packages

Search our cloud-based repository to find and install over 7,500 data science and machine learning packages. With the conda-install command, you can start using thousands of open-source Conda, R, Python and many other packages.

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.

平台

https://www.anaconda.com/

Build machine learning models

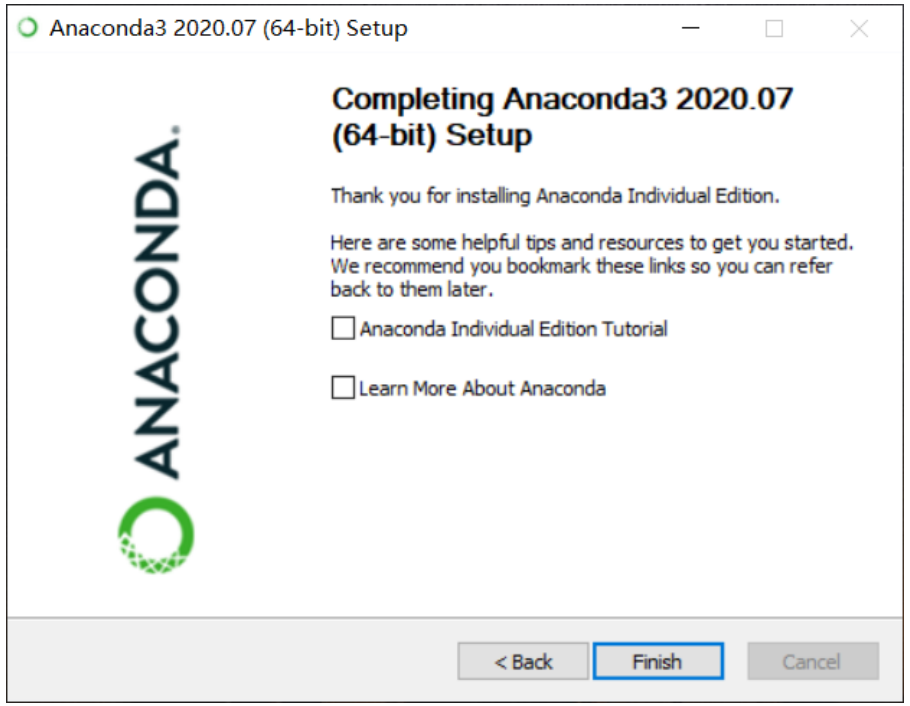
Build and train machine learning models using the best Python packages built by the open-source community, including scikit-learn, TensorFlow, and PyTorch.

Get Started

平台

Anaconda

https://www.anaconda.com/products/individual#windows



Anaconda Navigator

File Help

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home

Environments

Learning

Community

Documentation

Developer Blog

Applications on base (root) Channels Refresh

CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated
Launch

JupyterLab
2.1.5
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

Notebook
6.0.3
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated
Launch

Qt Console
4.7.5
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

Spyder
4.1.4
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch

Glueviz
0.15.2
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install

Orange 3
3.26.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install

RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Install

Anaconda Navigator

File Help

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home

Environments

Learning

Community

Documentation

Developer Blog

Search Environments

base (root)

Anaconda3

Create Clone Import Remove

Installed Channels Update index... Search Packages

Name	T	Description	Version
_ipyw_jlab_nb_ex...			0.1.0
alabaster			0.7.12
anaconda			2020.07
anaconda-client			1.7.2
anaconda-project			0.8.4
argh			0.26.2
asn1crypto			1.3.0
astroid			2.4.2
astropy			4.0.1.po...
atomicwrites			1.4.0
attrs			19.3.0
autopep8			1.5.3
babel			2.8.0
backcall			0.2.0
backports			1.0
backports.functoo...			1.6.1
backports.functoo...			1.6.1
backports.shutil-get-terminal-size			1.0.0

Anaconda Navigator

File Help

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home

Environments

Learning

Community

Documentation

Developer Blog

Applications on base (root) Channels Refresh

CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated
Launch

JupyterLab
2.1.5
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

Notebook
6.0.3
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated
Launch

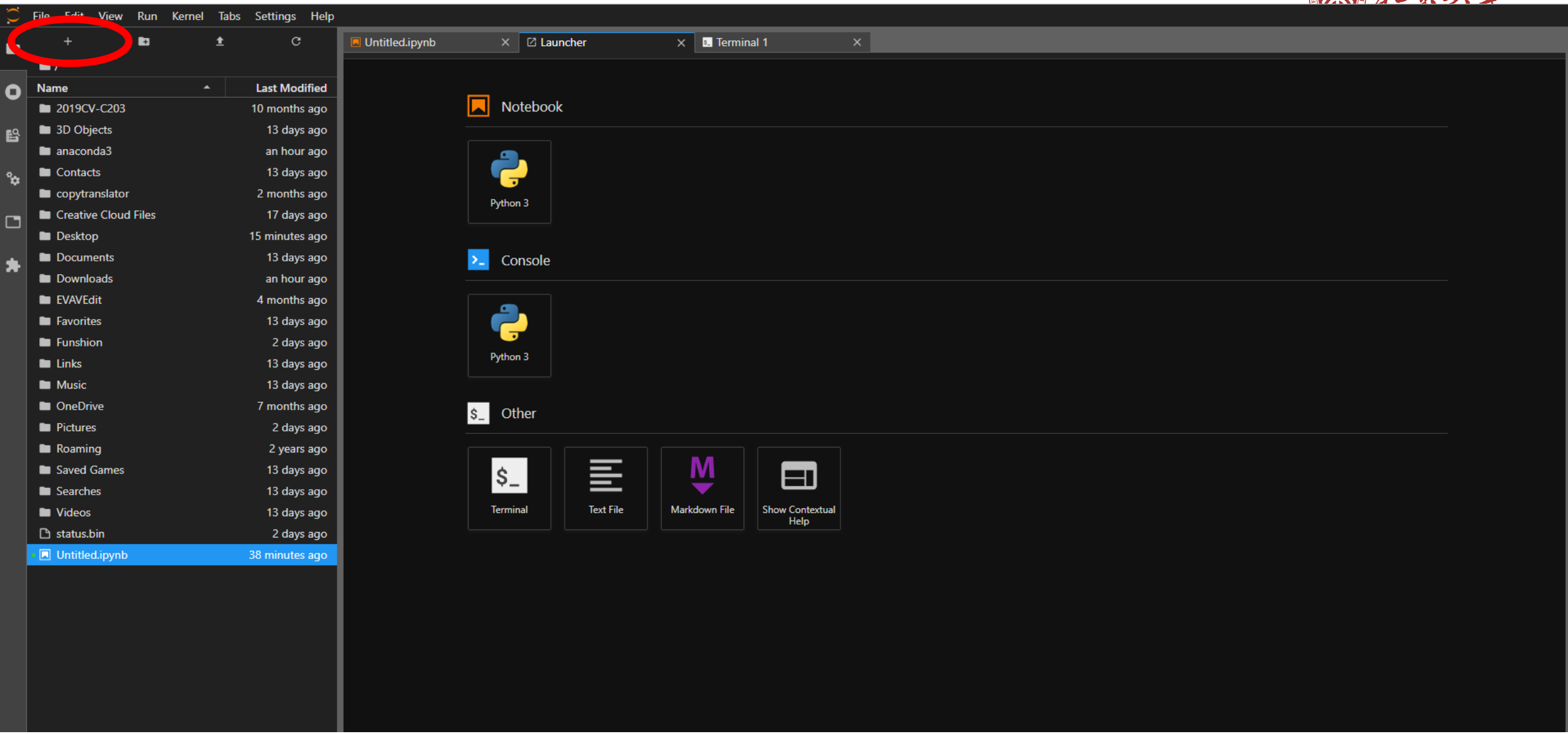
Qt Console
4.7.5
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

Spyder
4.1.4
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch

Glueviz
0.15.2
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install

Orange 3
3.26.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install

RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Install


File Edit View Run Kernel Tabs Settings Help
Untitled.ipynb
Launcher
Terminal 1
Name
Last Modified
2019CV-C203 10 months ago
3D Objects 13 days ago
anaconda3 an hour ago
Contacts 13 days ago
copytranslator 2 months ago
Creative Cloud Files 17 days ago
Desktop 15 minutes ago
Documents 13 days ago
Downloads an hour ago
EVAVEdit 4 months ago
Favorites 13 days ago
Funshion 2 days ago
Links 13 days ago
Music 13 days ago
OneDrive 7 months ago
Pictures 2 days ago
Roaming 2 years ago
Saved Games 13 days ago
Searches 13 days ago
Videos 13 days ago
status.bin 2 days ago
Untitled.ipynb 38 minutes ago
Notebook
Python 3
Console
Python 3
Other
Terminal
Text File
Markdown File
Show Contextual Help

File Edit View Run Kernel Tabs Settings Help

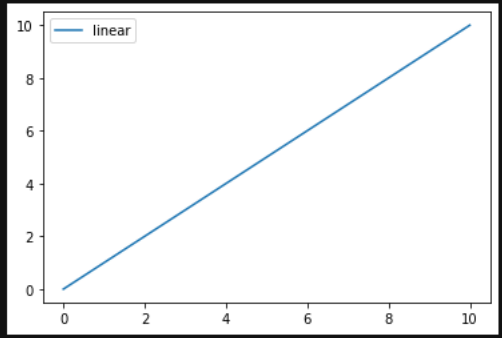
/

Name	Last Modified
2019CV-C203	10 months ago
3D Objects	13 days ago
anaconda3	an hour ago
Contacts	13 days ago
copytranslator	2 months ago
Creative Cloud Files	17 days ago
Desktop	14 minutes ago
Documents	13 days ago
Downloads	an hour ago
EVAVEdit	4 months ago
Favorites	13 days ago
Funshion	2 days ago
Links	13 days ago
Music	13 days ago
OneDrive	7 months ago
Pictures	2 days ago
Roaming	2 years ago
Saved Games	13 days ago
Searches	13 days ago
Videos	13 days ago
status.bin	2 days ago
Untitled.ipynb	38 minutes ago

Untitled.ipynb | Terminal 1

Code | Python 3

[1]:
```python
# Import the necessary packages and modules
import matplotlib.pyplot as plt
import numpy as np

# Prepare the data
x = np.linspace(0, 10, 100)

# Plot the data
plt.plot(x, x, label='linear')

# Add a legend
plt.legend()

# Show the plot
plt.show()
```



[2]:
```python
import numpy as np
```

[8]:
```python
a = np.random.randn(3,4)
print(a)
```

```
[[ 0.53871909  0.40661339  0.95665629 -0.30877368]
 [ 0.61670553 -0.19856635  0.92241214 -0.1236608 ]
 [ 2.1068687   0.80052442 -2.42887482 -0.27571512]]
```

[ ]:

File Edit View Run Kernel Tabs Settings Help

Files | Running | Commands | Cell Tools | Tabs

Name	Last Modified
House.ipynb	4 days ago
python_numpy_tuto...	7 days ago
python_numpy_tuto...	7 days ago
tensor_tutorial.ipynb	5 days ago
Untitled.ipynb	18 minutes ago
W2_OpenCV.ipynb	2 minutes ago
House256rgb.png	a year ago
HouseData.mat	5 days ago

W2_OpenCV. ●

Code　　Python 3

In [1]:
```
import cv2
```

利用opencv读取图片并显示

In [ ]:
```
img = cv2.imread('House256rgb.png', 1)
cv2.imshow('Image', img)
cv2.waitKey(0)
```

In [3]:
```
print('hello world')
```

```
hello world
```

要求：代码＋截图，带有运行结果

平台

https://www.jetbrains.com/pycharm/

平台

https://www.jetbrains.com/edu-products/download/#section=pycharm-edu

Education

Products for Learning Tools for Teaching

Get Your Educational Tool

Java Kotlin Python Scala JavaScript C/C++ Rust Go

PyCharm Edu is free & open source.

Licensed under Apache License, Version 2.0.

Download .exe or Install EduTools Plugin

PyCharm Edu

If you have already installed PyCharm Professional or Community

Version: 2021.2.3
Build: 212.5457.63
28 October 2021

Release notes ↗

System requirements

Installation Instructions

Other versions

Learn Python with JetBrains Academy ↗
Learn programming by creating working applications, read the JetBrains Academy guide ↗ for more details.

Print 函数

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

```
>>> print(6)
6
```

```
>>> print(['hello', 3, 4.5])
['hello', 3, 4.5])
```

```
>>> print({'A':1, 2:'B'})
{'A':1, 2:'B'}
```

```
>>> print(6//4, 7.0//4, 9%4)
1 1.0 1
```

```
[1]: print('Welcome to Computer Vision C203!')

Welcome to Computer Vision C203!

[2]: print("Welcome to Computer Vision C203!")
     # print("Welcome to Computer Vision C203!")

Welcome to Computer Vision C203!

[3]: a = 3
     b = 4
     c = a + b
     print(c)
     print(type(c))

7
<class 'int'>

[5]: a = 3.
     b = 4.
     c = a + b
     print(c)
     print(type(c))

7.0
<class 'float'>
```

内置原子数据类型

- int float
- 布尔类型

内置原子数据类型 int float

北京大学 PEKING UNIVERSITY

int float 标准运算 ‘+’ ‘-’ ‘/’ '*' '%' '//'

```
>>> print(6//4, 7.0//4, 9%4, 6/4)
1 1.0 1 1.5
```

```
>>> print(4+5, 8.0-2, 3*4)
9 6.0 12
```

内置原子数据类型 布尔类型

北京大学 PEKING UNIVERSITY

标准运算 and or not

```
>>> False or True
True
```

```
>>> not (False or True)
False
```

```
>>> True and True
True
```

比较运算符

```
>>> print(3 > 2)
True
```

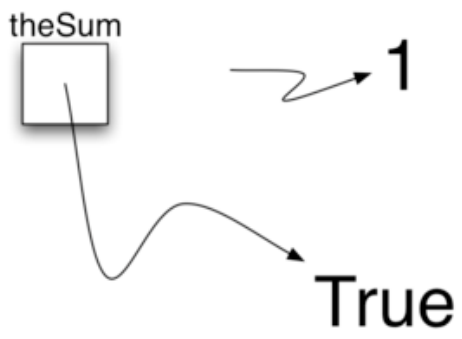
```
>>> print(4 == 4)
True
```

```
>>> print(2 <= 1)
False
```

Operation Name	Operator	Explanation
less than	$<$	Less than operator
greater than	$>$	Greater than operator
less than or equal	$<=$	Less than or equal to operator
greater than or equal	$>=$	Greater than or equal to operator
equal	$==$	Equality operator
not equal	$!=$	Not equal operator
logical and	*and*	Both operands True for result to be True
logical or	*or*	One or the other operand is True for the result to be True
logical not	*not*	Negates the truth value, False becomes True, True becomes False

赋值语句

```
>>> theSum = 0

>>> theSum
0

>>> theSum = theSum + 1

>>> theSum
1

>>> theSum = True

>>> theSum
True
```



内置集合数据类型

- 列表
- 元组
- 字典
- 集合
- 字符串

内置集合数据类型 列表

```
>>> [1,3,True,6.5]
[1, 3, True, 6.5]

>>> myList = [1,3,True,6.5]

>>> myList
[1, 3, True, 6.5]
```

列表是异构的

内置集合数据类型 列表

```
myList = [1,2,3,4]
A = [myList]*3
print(A)

myList[2]=45
print(A)

print(3 in myList)

print(myList[0:3])

myList2 = [4, 5]
print(myList + myList2)
```

Operation Name	Operator	Explanation
indexing	[ ]	Access an element of a sequence
concatenation	+	Combine sequences together
repetition	*	Concatenate a repeated number of times
membership	in	Ask whether an item is in a sequence
length	len	Ask the number of items in the sequence
slicing	[:]	Extract a part of a sequence

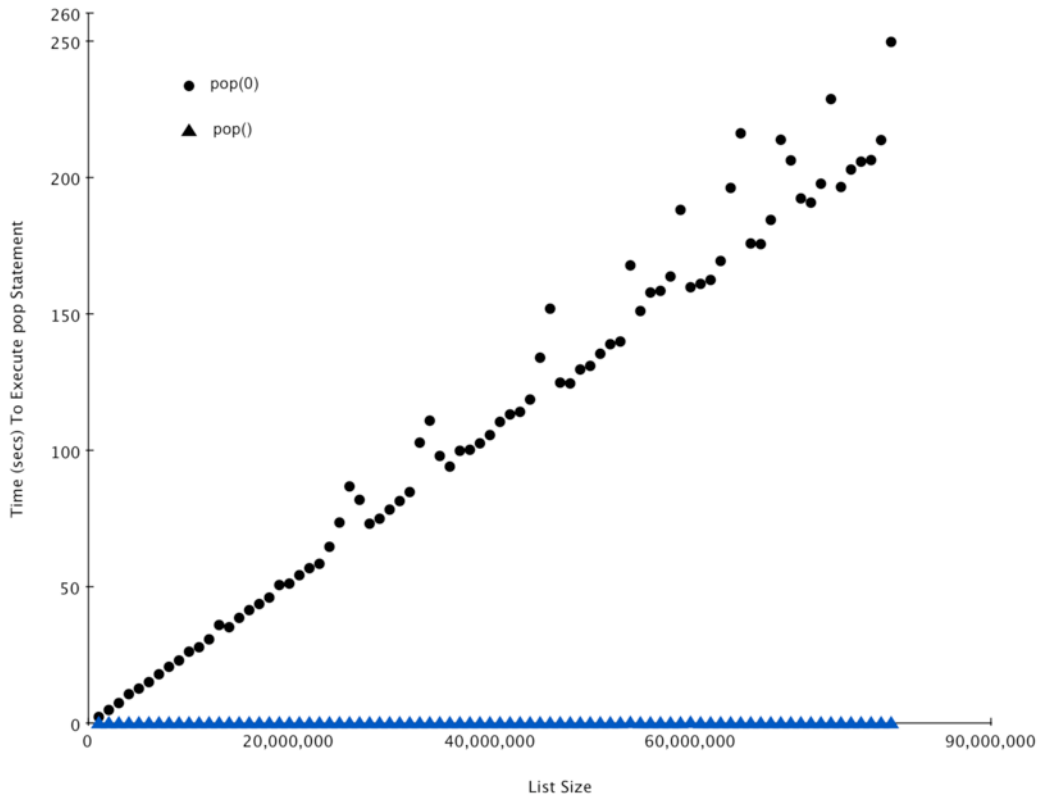
内置集合数据类型 列表

```
myList = [1024, 3, True, 6.5]
myList.append(False)
print(myList)
myList.insert(2,4.5)
print(myList)
print(myList.pop())
print(myList)
print(myList.pop(1))
print(myList)
myList.pop(2)
print(myList)
myList.sort()
print(myList)
myList.reverse()
print(myList)
print(myList.count(6.5))
print(myList.index(4.5))
myList.remove(6.5)
print(myList)
del myList[0]
```

Method Name	Use	Explanation
`append`	`alist.append(item)`	Adds a new item to the end of a list
`insert`	`alist.insert(i,item)`	Inserts an item at the ith position in a list
`pop`	`alist.pop()`	Removes and returns the last item in a list
`pop`	`alist.pop(i)`	Removes and returns the ith item in a list
`sort`	`alist.sort()`	Modifies a list to be sorted
`reverse`	`alist.reverse()`	Modifies a list to be in reverse order
`del`	`del alist[i]`	Deletes the item in the ith position
`index`	`alist.index(item)`	Returns the index of the first occurrence of `item`
`count`	`alist.count(item)`	Returns the number of occurrences of `item`
`remove`	`alist.remove(item)`	Removes the first occurrence of `item`

内置集合数据类型 列表

Operation	Big-O Efficiency
index []	O(1)
index assignment	O(1)
append	O(1)
pop()	O(1)
pop(i)	O(n)
insert(i,item)	O(n)
del operator	O(n)
iteration	O(n)
contains (in)	O(n)
get slice [x:y]	O(k)
del slice	O(n)
set slice	O(n+k)
reverse	O(n)
concatenate	O(k)
sort	O(n log n)
multiply	O(nk)



range 函数

```
>>> range(10)
range(0, 10)

>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

>>> range(5,10)
range(5, 10)

>>> list(range(5,10))
[5, 6, 7, 8, 9]

>>> list(range(5,10,2))
[5, 7, 9]

>>> list(range(10,1,-1))
[10, 9, 8, 7, 6, 5, 4, 3, 2]
```

内置集合数据类型 字符串

```
>>> "David"
'David'

>>> myName = "David"
>>> myName[3]
'i'

>>> myName*2
'DavidDavid'

>>> len(myName)
5
```

内置集合数据类型 字符串

```
>>> myName
'David'

>>> myName.upper()
'DAVID'

>>> myName.center(10)
'  David  '

>>> myName.find('v')
2

>>> myName.split('v')
['Da', 'id']
```

内置集合数据类型 字符串

```
>>> myList
[1, 3, True, 6.5]

>>> myList[0]=2**10

>>> myList
[1024, 3, True, 6.5]

>>>
>>> myName
'David‘

>>> myName[0]='X'

TypeError: object doesn't support item assignment
```

内置集合数据类型 元组

```
>>> myTuple = (2,True,4.96)

>>> myTuple
(2, True, 4.96)

>>> len(myTuple)
3

>>> myTuple[0]
2

>>> myTuple * 3
(2, True, 4.96, 2, True, 4.96, 2, True, 4.96)

>>> myTuple[0:2]
(2, True)

>>> myTuple[1]=False
```

内置集合数据类型 集合

```
>>> {3,6,"cat",4.5,False}
{False, 4.5, 3, 6, 'cat'}

>>> mySet = {3,6,"cat",4.5,False}

>>> mySet
{False, 4.5, 3, 6, 'cat'}

>>> len(mySet)
5

>>> False in mySet
True

>>> "dog" in mySet
False
```

内置集合数据类型 字典

```
>>> capitals = {'Heilongjiang':'Harbin', 'Guangdong':'Guangzhou'}

>>> capitals
{'Heilongjiang': 'Harbin', 'Guangdong': 'Guangzhou'}

>>> capitals['Hunan']='Changsha'
>>> print(capitals)
{'Heilongjiang': 'Harbin', 'Guangdong': 'Guangzhou', 'Hunan': 'Changsha'}

>>> print(len(capitals))
3

>>> for k in capitals:
>>>         print(capitals[k]," is the capital of ", k)
Harbin  is the capital of  Heilongjiang
Guangzhou  is the capital of  Guangdong
Changsha  is the capital of  Hunan
```

内置集合数据类型 字典

Operator	Use	Explanation
`[]`	`myDict[k]`	Returns the value associated with `k`, otherwise its an error
`in`	`key in adict`	Returns `True` if key is in the dictionary, `False` otherwise
`del`	del `adict[key]`	Removes the entry from the dictionary

内置集合数据类型 字典

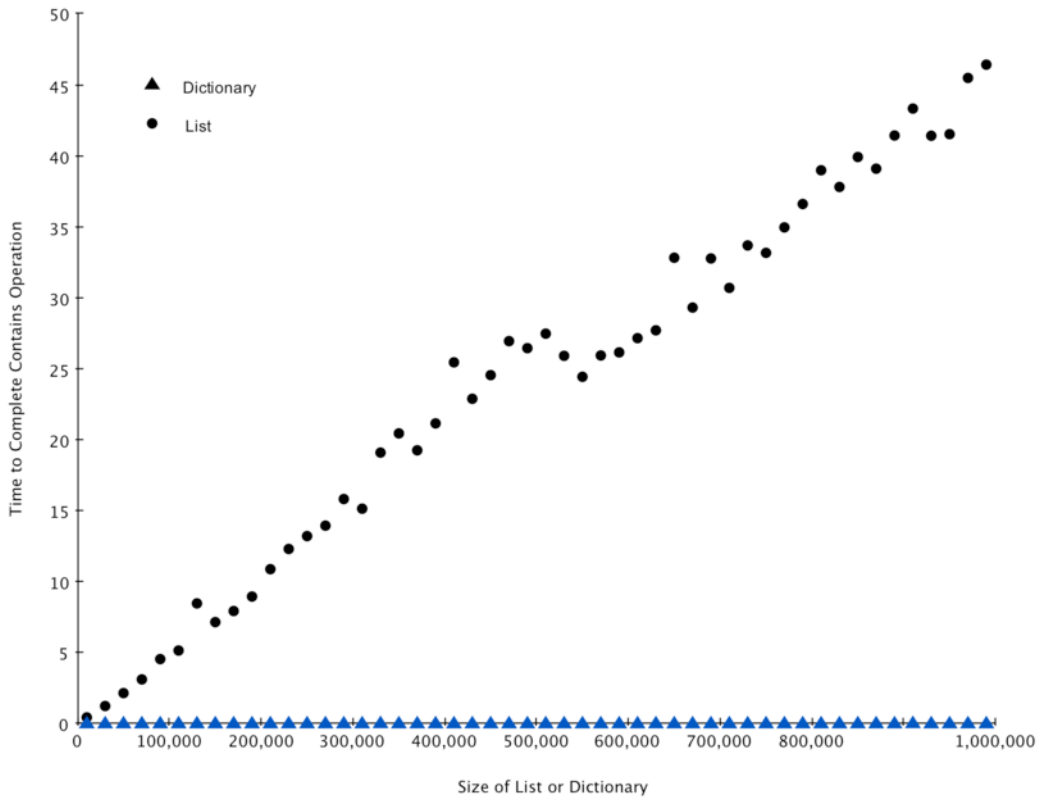
```
>>> phoneext={'david':1410,'brad':1137}

>>> phoneext
{'brad': 1137, 'david': 1410}

>>> phoneext.keys()
dict_keys(['brad', 'david'])

>>> list(phoneext.keys())
['brad', 'david']

>>> phoneext.values()
dict_values([1137, 1410])
```

¶

Method Name	Use	Explanation
`keys`	`adict.keys()`	Returns the keys of the dictionary in a dict_keys object
`values`	`adict.values()`	Returns the values of the dictionary in a dict_values object
`items`	`adict.items()`	Returns the key-value pairs in a dict_items object
`get`	`adict.get(k)`	Returns the value associated with `k`, `None` otherwise
`get`	`adict.get(k,alt)`	Returns the value associated with `k`, `alt` otherwise

内置集合数据类型 字典

operation	Big-O Efficiency
copy	O(n)
get item	O(1)
set item	O(1)
delete item	O(1)
contains (in)	O(1)
iteration	O(n)



内置集合数据类型-测试

- 列表
- 元组
- 字典
- 集合
- 字符串

```
a = [5, 3, 9, 4]
print(a.index(3))
```

```
a = [1, 2, 3, 4]
print(a[2])
```

```
a = [1, 2, 3, 4]
print(3 in a)
```

```
a = [1, 2, 3, 4]
a.append(6)
print(a)
```

```
a = [1, 2, 3, 4]
print(a.pop())
print(a)
```

```
a = [1, 2, 3, 4]
print(a.pop(0))
print(a)
```

```
a = [1, 2, 3, 4]
print(a.insert(0, 8))
print(a)
```

```
a = [1, 3, 2, 4]
print(a.sort())
print(a)
```

```
b = (1, 2, 3, 4)
b[2] = 0
print(b)
```

内置集合数据类型-测试

- 列表
- 元组
- 字典
- 集合
- 字符串

```
print(list(range(1,5)))

print(list(range(5,1,-1)))
```

```
phoneext={'david':1410,'brad':1137}
print('david' in phoneext)
```

```
phoneext={'david':1410,'brad':1137}
print(1410 in phoneext)
```

```
print(len(phoneext))
```

```
print(phoneext[0])
```

```
phoneext['david']
```

```
ss = 'abc'
print(list(ss))
```

```
ss = 'abc'
print(ss.upper())
```

控制结构

- 迭代
  - while语句
  - for语句
- 选择
  - if, elif, else语句

控制结构 while语句

```
>>> counter = 1

>>> while counter <= 5:
        ... print("Hello, world")
        ... counter = counter + 1

Hello, world
Hello, world
Hello, world
Hello, world
Hello, world
```

```
while counter <= 10 and not done:
...
```

控制结构 for语句

```
>>> for item in [1,3,6,2,5]:
    ... print(item)
    ...

1
3
6
2
5
```

```
>>> for item in range(5):
    ... print(item**2)
    ...
0
1
4
9
16
```

```
wordlist = ['cat','dog','rabbit']
letterlist = [ ]
for aword in wordlist:
    for aletter in aword:
        letterlist.append(aletter)
print(letterlist)
```

控制结构 break语句

```
for letter in 'Python':     # 第一个实例
    if letter == 'h':
        break
    print('当前字母 :', letter)

var = 10                    # 第二个实例
while var > 0:
    print('当前变量值 :', var)
    var = var -1
    if var == 5:
        break

print("Good bye!")
```

```
当前字母 : P
当前字母 : y
当前字母 : t
当前变量值 : 10
当前变量值 : 9
当前变量值 : 8
当前变量值 : 7
当前变量值 : 6
Good bye!
```

控制结构 continue语句

```
for letter in 'Python':     # 第一个实例
    if letter == 'h':
        continue
    print('当前字母 :', letter)

var = 10                    # 第二个实例
while var > 0:
    var = var -1
    if var == 5:
        continue
    print('当前变量值 :', var)

print("Good bye!")
```

```
当前字母 : P
当前字母 : y
当前字母 : t
当前字母 : o
当前字母 : n
当前变量值 : 9
当前变量值 : 8
当前变量值 : 7
当前变量值 : 6
当前变量值 : 4
当前变量值 : 3
当前变量值 : 2
当前变量值 : 1
当前变量值 : 0
Good bye!
```

控制结构 if,elif,else语句

```
if score >= 90:
    print('A')
else:
    if score >=80:
        print('B')
    else:
        if score >= 70:
            print('C')
        else:
            if score >= 60:
                print('D')
            else:
                print('F')
```

```
if score >= 90:
    print('A')
elif score >=80:
    print('B')
elif score >= 70:
    print('C')
elif score >= 60:
    print('D')
else:
    print('F')
```

```
if n<0:
    n = abs(n)
print(math.sqrt(n))
```

控制结构 创建列表

```
sqlist=[]
for x in range(1,11):
    sqlist.append(x*x)
sqlist
```

```
sqlist=[x*x for x in range(1,11)]
sqlist
```

```
sqlist=[x*x for x in range(1,11) if x%2 != 0]
sqlist
```

```
[ch.upper() for ch in 'comprehension' if ch not in 'aeiou']
```

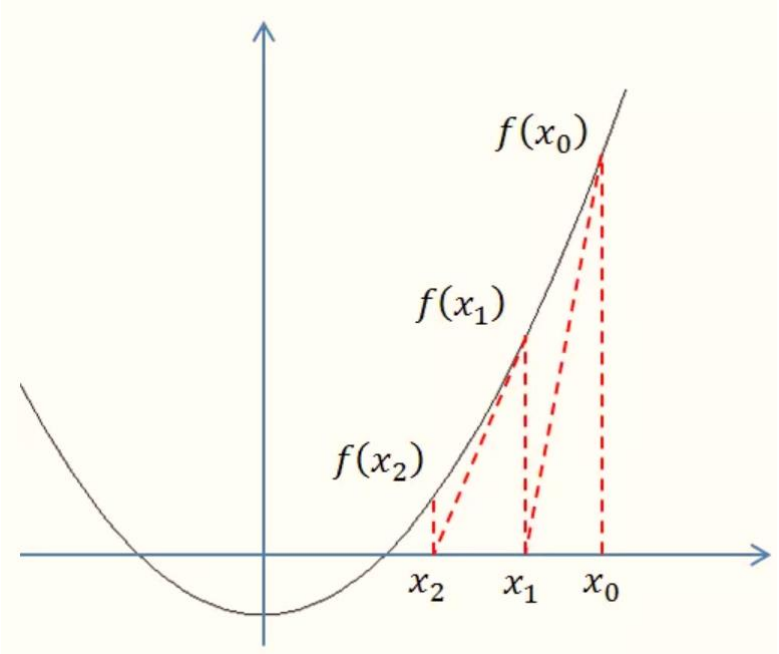
定义函数

```
>>> def square(n):
...         return n**2
...

>>> square(3)
9

>>> square(square(3))
81
```

定义函数

- 平方根函数
- 牛顿法

牛顿法是一种在实数域和复数域上近似求解方程的方法。方法使用函数f(x)的泰勒级数的前面几项来寻找方程f(x)=0的根。



$$0 = (x - x_0) \cdot f'(x_0) + f(x_0)$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

推导

root = (1/2)*(root + (input_num / root))

定义函数

- 平方根函数
- 牛顿法

```
def squareroot(input_num):
    root = input_num/2
    for k in range(20):
        root = (1/2)*(root + (input_num / root))
    return root

>>> squareroot(3)
1.7320508075688772
```

面向对象 定义类

- 创建类

```
class ClassName:
   '类的帮助信息'   #类文档字符串
   class_suite  #类体
```

- 创建实例对象

```
"创建 Employee 类的第一个对象"
emp1 = Employee("Zara", 2000)
"创建 Employee 类的第二个对象"
emp2 = Employee("Manni", 5000)
```

- 访问属性

```
emp1.displayEmployee()
emp2.displayEmployee()
print("Total Employee %d" % Employee.empCount)
```

- 实例

```
class Employee:
   '所有员工的基类'
   empCount = 0

   def __init__(self, name, salary):
      self.name = name
      self.salary = salary
      Employee.empCount += 1

   def displayCount(self):
      print("Total Employee %d" % Employee.empCount)

   def displayEmployee(self):
      print("Name : ", self.name,  ", Salary: ", self.salary)
```

面向对象 定义类

```
class Employee:
   '所有员工的基类'
   empCount = 0

   def __init__(self, name, salary):
      self.name = name
      self.salary = salary
      Employee.empCount += 1

   def displayCount(self):
      print("Total Employee %d" % Employee.empCount)

   def displayEmployee(self):
      print("Name : ", self.name,  ", Salary: ", self.salary)

# "创建 Employee 类的第一个对象"
emp1 = Employee("Zara", 2000)
# "创建 Employee 类的第二个对象"
emp2 = Employee("Manni", 5000)

emp1.displayEmployee()
emp2.displayEmployee()
print("Total Employee %d" % Employee.empCount)
```

```
Name :  Zara , Salary:  2000
Name :  Manni , Salary:  5000
Total Employee 2
```

面向对象 定义类

```
class Employee:
   '所有员工的基类'
   empCount = 0

   def __init__(self, name, salary):
      self.name = name
      self.salary = salary
      Employee.empCount += 1

   def displayCount(self):
      print("Total Employee %d" % Employee.empCount)

   def displayEmployee(self):
      print("Name : ", self.name,  ", Salary: ", self.salary)

# "创建 Employee 类的第一个对象"
emp1 = Employee("Zara", 2000)
# "创建 Employee 类的第二个对象"
emp2 = Employee("Manni", 5000)

emp1.displayEmployee()
emp2.displayEmployee()
print("Total Employee %d" % Employee.empCount)
```

```
Name :  Zara , Salary:  2000
Name :  Manni , Salary:  5000
Total Employee 2
```

今天的作业

- 安装Anaconda
- 安装虚拟环境，删除
- 运行课件附带所有的 .ipynb文件

交流&问题?