

SI100B Introduction to Information Science and Technology **(Python Programming)**

张海鹏 Haipeng Zhang

School of Information Science and Technology
ShanghaiTech University

Instructor for programming

- Instructor: 张海鹏 Haipeng Zhang
- Research
 - Data mining, finding insights from data
 - Fintech, applying technology to solve problems in finance

Office: Room 1C-303.C, SIST Building 1

Email: zhanghp@shanghaitech.edu.cn

Web: <http://sist.shanghaitech.edu.cn/2018/1018/c2739a34559/page.htm>



■ 研究方向：数据挖掘（金融数据、电商数据、社交媒体数据），金融科技

■ 教育背景

- 2009-2014



美国印第安纳大学

计算机科学 博士

- 2007.8-2007.12



香港科技大学

计算机科学 交换生

- 2005-2009



南京大学

软件工程 学士

■ 工作、实习经历

- 2018-至今



上海科技大学信息科学与技术学院

助理教授，研究员，博导

- 2015-2018



中国金融期货交易所

博士后研究员

- 2014-2015



IBM中国研究院

研究科学家

- 2013.6-9



三星研究院（硅谷）

移动数据挖掘

- 2013.2-5



微软研究院（剑桥）

社交媒体数据挖掘

- 2012.5-8



eBay研究院（硅谷）

电子商务数据挖掘

- 2010.6-8



国立情报学研究所（东京）

推荐系统研究



Instructor for signal processing

- Instructor: 娄鑫 Xin Lou
- Research
 - Integrated Circuits for Computer Vision
 - Digital VLSI

Office: Room 1D-303.E, SIST Building 1

Email: louxin@shanghaitech.edu.cn

Web: <https://sist.shanghaitech.edu.cn/2018/0502/c2739a24281/page.htm>

娄鑫

■ 研究方向：智能视觉芯片，超大规模数字集成电路

■ 教育背景

• 2012-2016	 NANYANG TECHNOLOGICAL UNIVERSITY	新加坡南洋理工大学	电子工程 博士
• 2010-2012	 KTH TEKNIKAL OCH KONST UNIVERSITET	瑞典皇家工学院	片上系统设计 硕士
• 2006-2010	 ZHEJIANG UNIVERSITY	浙江大学	电子信息技术及仪器 学士

■ 工作、实习经历

• 2017-至今	 上海科技大学 ShanghaiTech University	上海科技大学信息科学与技术学院	助理教授，研究员，博导
• 2016.4-2017.2		新加坡南洋理工大学	研究科学家

Instructor for electronics

- Instructor: 梁俊睿 Junrui Liang
 - Research
 - Self-powered Internet of Things 自供能物联网
 - Power electronics 电力电子
 - Mechatronics 机电一体化
- Office: Room 1D-303.D, SIST Building 1
- Email: liangjr@shanghaitech.edu.cn
- Web: <http://metal.shanghaitech.edu.cn>

梁俊睿

■ 研究方向：自供能物联网，电力电子，机电一体化

■ 教育背景

- 2007-2010 香港中文大学
 香港中文大學
The Chinese University of Hong Kong 香港中文大学
机械与自动化工程 博士
- 2004-2007 上海交通大学
 上海交通大学
SHANGHAI JIAO TONG UNIVERSITY 上海交通大学
精密仪器与机械 硕士
- 2000-2004 上海交通大学
 上海交通大学
SHANGHAI JIAO TONG UNIVERSITY 上海交通大学
仪器科学与技术 学士

■ 工作、实习经历

- 2013-至今 上海科技大学
 上海科技大学
ShanghaiTech University 上海科技大学
信息科学与技术学院
助理教授，研究员，博导
- 2015-2016 美国加州大学伯克利分校
 Berkeley
UNIVERSITY OF CALIFORNIA 美国加州大学伯克利分校
访问学者
- 2013 香港城市大学
 CityU
香港城市大學
City University of Hong Kong 香港城市大学
博士后研究员
- 2010-2012 香港中文大学
 香港中文大學
The Chinese University of Hong Kong 香港中文大学
博士后研究员

SI100B

- 24 lectures on Wed&Fri, for the first 12 weeks
 - **1. Python programming (PP)**, 8 lectures, **20%** of the total score
 - **2. Signal processing (SP)**, 8 lectures, **20%**,
 - **3. Electrical and electronic technologies (EE)**, 8 lectures, **20%**
- Project (PJ) during the last 4 weeks
 - Choose 1 from 4 projects, each one with a different focus: programming, signal processing, electronics
 - **39%**, details to be announced later in the semester
- Miscellaneous
 - **1%** for one logistics assignment on Academic Integrity
 - For PP, SP, EE, and PJ, each may include bonus scores up to 10% of the score for the corresponding part. E.g PP may take up to 22% of the total score
 - Additional points (up to 3%) may be given to students with good answers on Piazza, details to be announced.

Python Programming

- Goals
 - Learn Python as a computational and engineering tool.
 - Be successful in science, engineering, business, and other professions.
- Reference: www.python.org, **Python 3.7**
 - [The Python Tutorial](#)
 - [The Python Language Reference](#), **core** syntax and semantics
 - [The Python Standard Library](#), **exact** syntax and semantics
 - [The Python HOWTOs](#), guide for specific tasks
 - *Learning Python* (《Python学习手册》), *5th Edition*, Mark Lutz
- Slides, discussions, assignments, announcements on Piazza:
 - <https://piazza.com/shanghaitech.edu.cn/fall2020/si100b/>
 - Register using your real Chinese names and ShanghaiTech emails.
 - **Check Piazza and email daily. Don't miss out important messages.**
- General course information on course website <https://si100b.org/>
- Gradescope: www.gradescope.com (register with entry code **9D7J7V**), for submitting some paper&pen (non-programming) assignments and we will publish your scores on it.

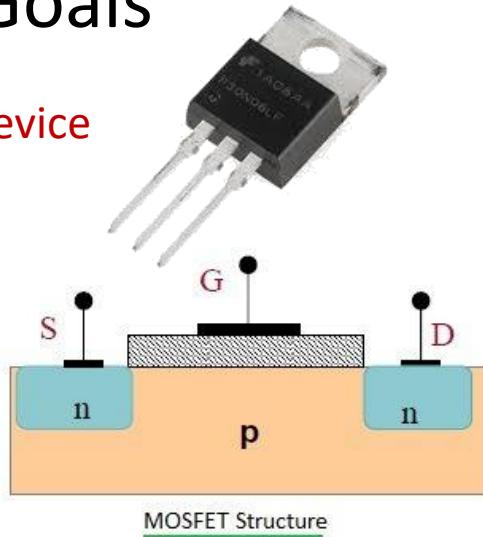
Signal Processing

- Goals
 - To provide an introduction to signal and signal processing
 - Familiarize the students with signal processing using MATLAB
- Reference books
 - Richard G. Lyons, Understanding Digital Signal Processing
 - Alan V. Oppenheim and Ronald W. Schafer, Discrete-Time Signal Processing, 3rd Edition
 - Sanjit K. Mitra, Digital Signal Processing: A Computer-Based Approach, 4th edition

Electronics

- Goals

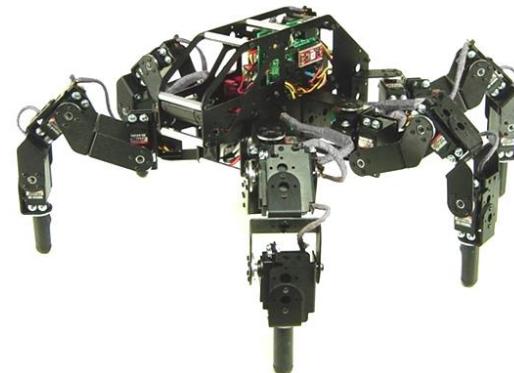
Device



Circuit



System



- Reference books

- Sarah Harris, David Harris, *Digital Design and Computer Architecture. ARM Edition*, Morgan Kaufmann, 2015, ISBN 978-0-12-800056-4.
(电子版<https://b-ok.as/book/2716132/efaeba?regionChanged>)

Programming Arrangement

- 8 lectures

Lecture	Content
1	Intro to course, computer basics, environment setup, basics about print, comments, and control flow
2	Variables (some on standard data types: numbers, string, list, tuple, dict), Expressions (some about operators), Statements
3	More on operators; condition, program structure
4	Loop, functions
5	Exception handling, file operations, modules, class and objects
5	Class and inheritance
7	NumPy, Pandas
8	Web crawler and data analysis example

- 2 programming assignments
 - HW1 30%, HW2 40%
 - Announce on Piazza
- Quizzes (30%), **in class**

PP Arrangement

- TAs for programming part
 - 刁子豪, diaozh@shanghaitech.edu.cn
 - 高子淇, gaozq@shanghaitech.edu.cn
 - 邱龙田, qiult@shanghaitech.edu.cn
 - 徐鸿图, xuht1@shanghaitech.edu.cn
 - 吴大千, wudq1@shanghaitech.edu.cn
 - 郭恺豪, guokh@shanghaitech.edu.cn
- Related inquiries
 - Google, StackOverflow
 - Ask on Piazza
 - Email, put **SI100B** in titles
- TA office hours
 - Time and location TBD
- TA tutorials
 - 1-1.5 hours each week
 - Time and location, **vote on the slot, via Piazza**

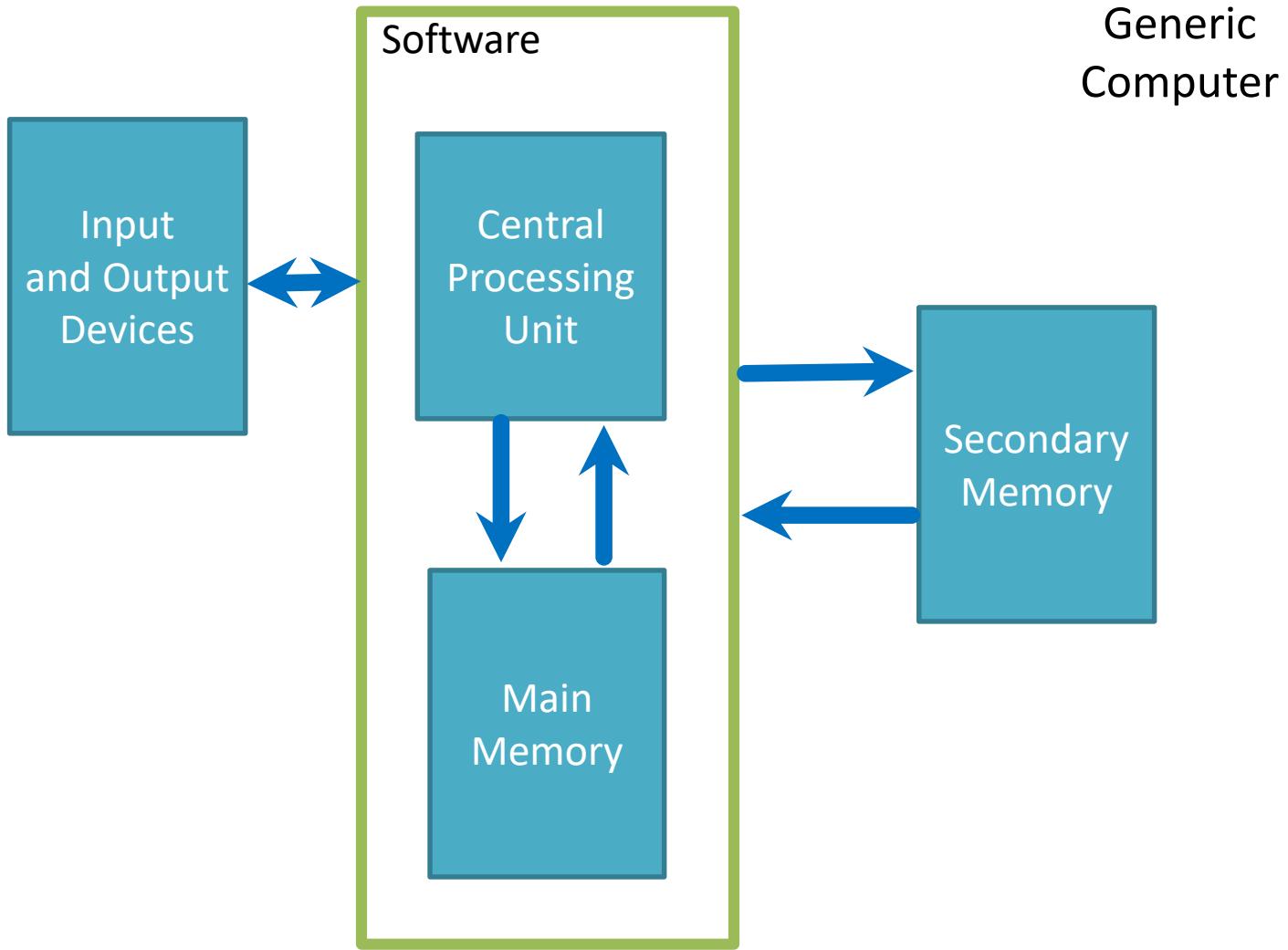
Academic Integrity

- The university code on academic integrity
 - <http://openinfo.shanghaitech.edu.cn/xswyhxgzdjy/list.htm>
- Academic integrity policy for SI100B
 - <https://si100b.org/resource-policy/#policies>
- We **DON'T** tolerate academic misconducts
 - ✗ 抄袭他人代码，抄袭网络源代码(Github, CSDN等)
 - ✗ 反编译别人的编译后文件(.pyc)
 - ✗ 散布源代码
 - ✗ 帮助或者寻求别人帮助修改代码问题(debug)
 - ✗ 代他人签到
 - 不允许提供抄袭，保护好自己的代码
 - 鼓励讨论，仅限于算法层面，不允许讨论与借鉴实现细节

Learning Objectives

- How a program runs on a basic computer
- What is Python
- How to get Python
- Run a Python program
- Basics about print and comments
- Basic control flow

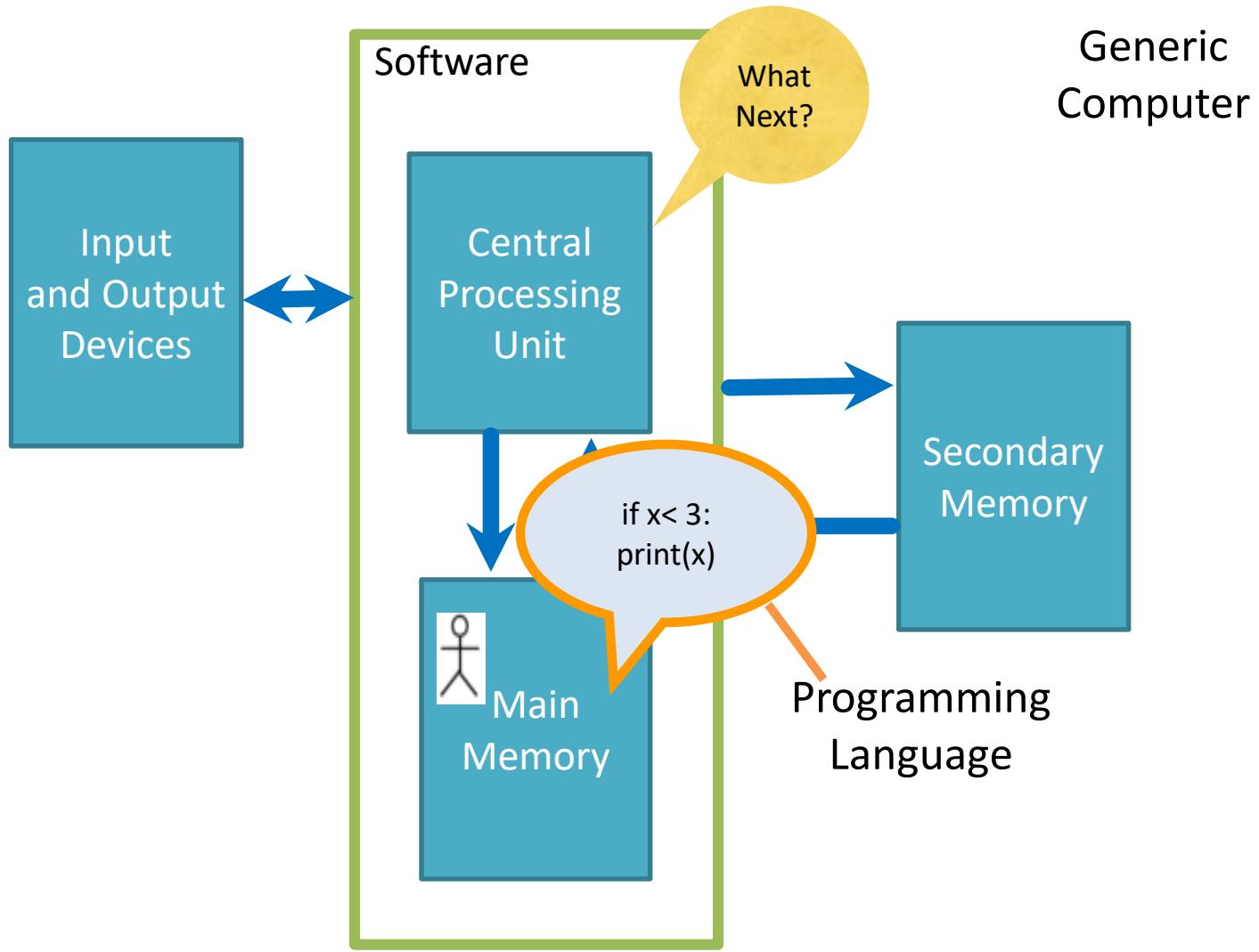
A basic computer



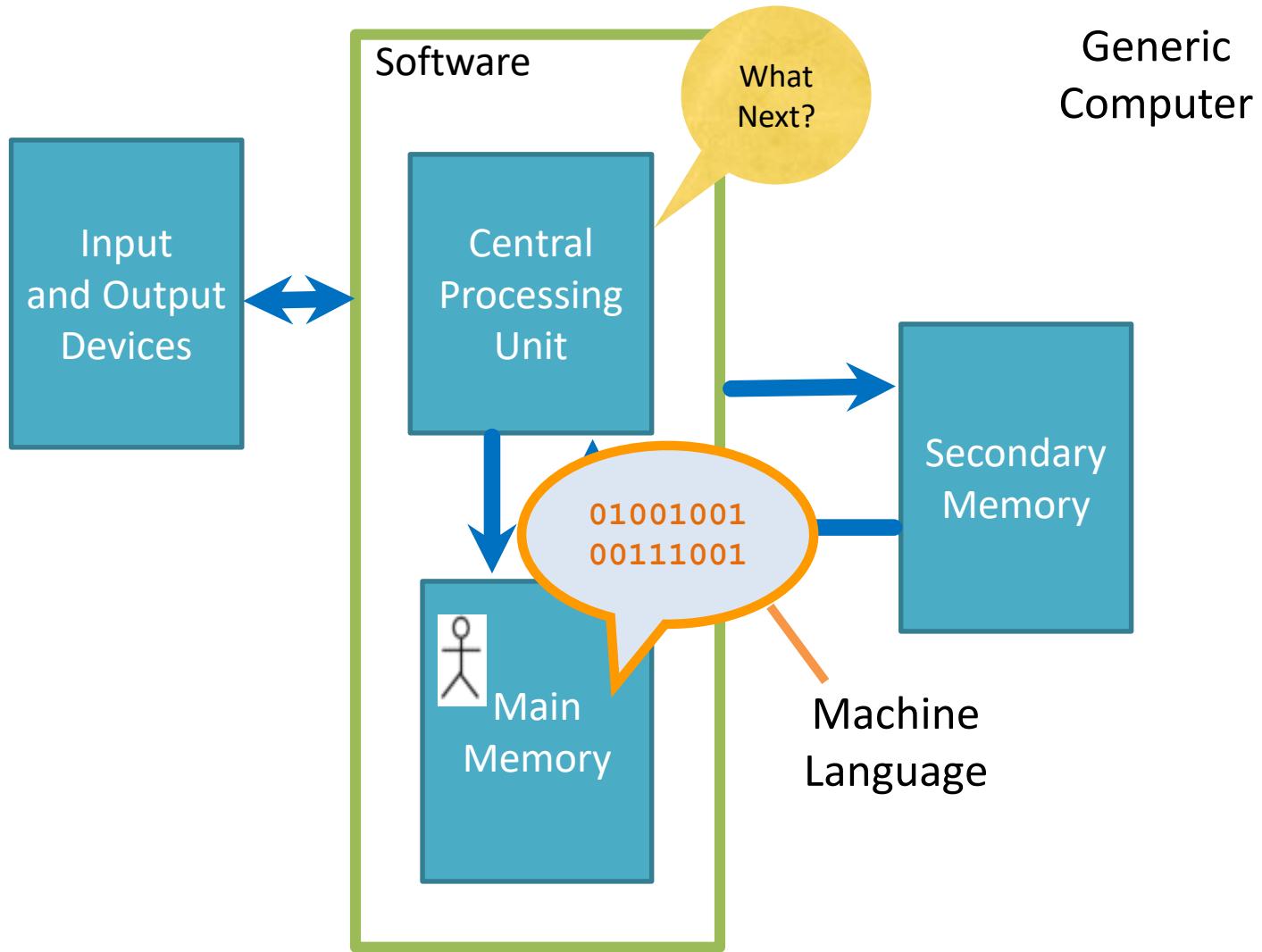
Definitions

- Central Processing Unit: Runs the Program - Executing simple instructions (add, multiply, etc), one by one.
- Input Devices: Keyboard, Mouse, Touch Screen
- Output Devices: Screen, Speakers, Printer, DVD Burner
- Main Memory: Fast small temporary storage - lost on reboot
- Secondary Memory: Slower large permanent storage - lasts until deleted - disk drive / memory stick

How a program runs on a basic computer



How a program runs on a basic computer



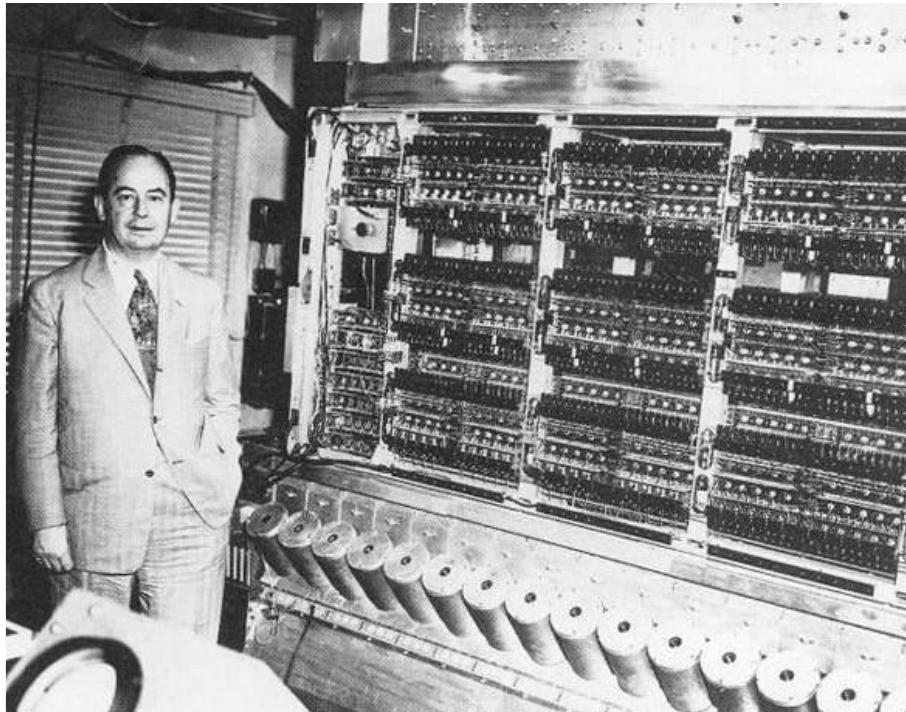
Basic Computer Architecture

- A very simple computer (RaspberryPi)



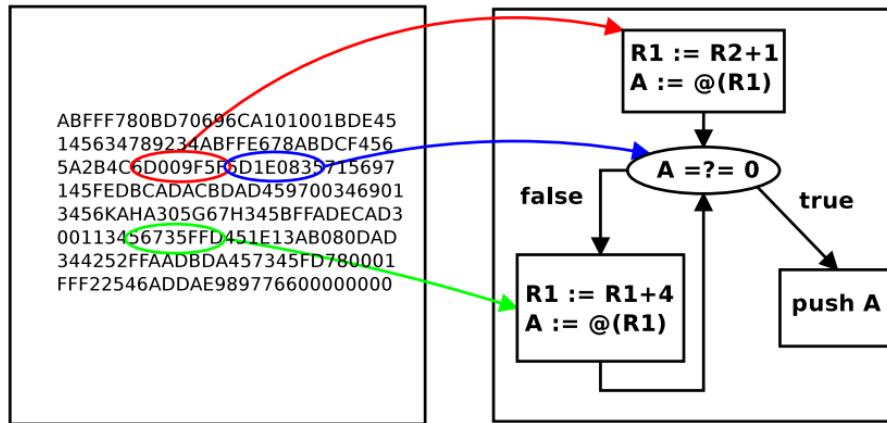
A brief history of programming languages

- In the late 1940s, the stored-program computer invented by John von Neumann, programs written in machine language, $\{0,1\}^+$



A brief history of programming languages

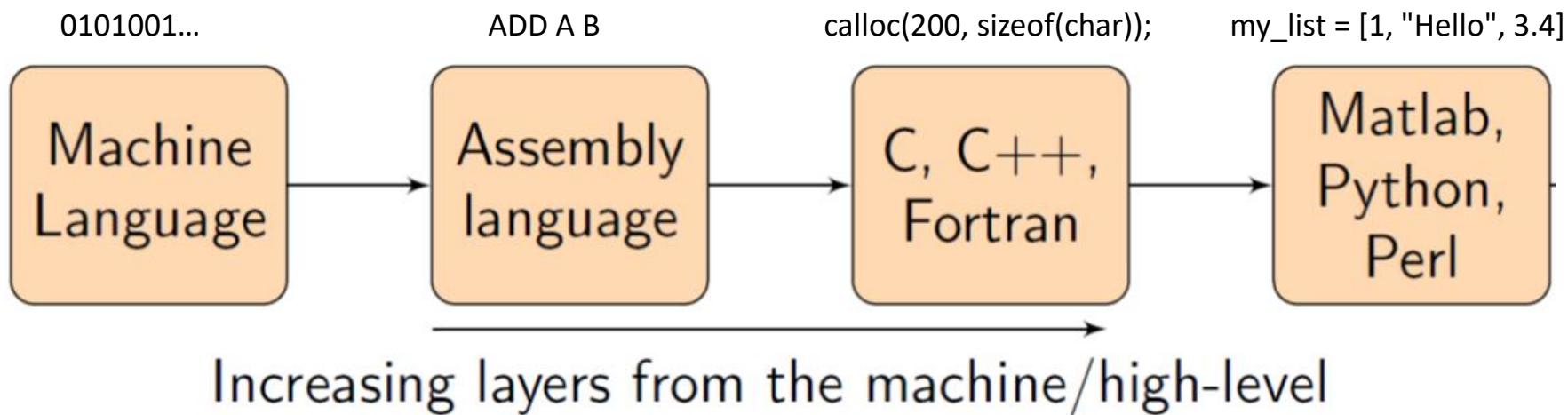
- Assembly language: in 1949 numeric codes were replaced by symbolic forms
 - Mov R, 2 (R=2)
- Assembler: translate assembly language into corresponding machine language.



- FORTRAN language and its compiler: between 1954 and 1957, developed by IBM

Programming Languages

- Far from hardware, more human-friendly.



How to pick a language?

➤ Meet your application requirements

- Must it be efficient?
- Easy to understand
- Easy to write (time, size of programs)
- Easy to debug, maintain, and prevent errors
- Plenty of support

Why Python?

- ✓ **Efficiency**
 - efficient for programmers to write programs
 - but Python programs themselves are not efficient
- ✓ **Easy to learn**
- ✓ **Active user community**
- ✓ **Lots of handy packages**
- ✓ **Lots of open source projects**

And as a bonus

- Once you learn how to code in one programming language it will be easier to learn another programming language, and another, and another...
- C#, Java, C++, Perl...

A brief history

- Invented by Guido van Rossum, started in 1989
 - "hobby' programming project that would keep him occupied during the week around Christmas"
 - Top 5 popular programming languages
 - Ver 3.0 in 2008, limited backward compatibility



Growing popularity

- Among top 5 by popularity

Aug 2020	Aug 2019	Change	Programming Language	Ratings	Change
1	2	▲	C	16.98%	+1.83%
2	1	▼	Java	14.43%	-1.60%
3	3		Python	9.69%	-0.33%
4	4		C++	6.84%	+0.78%
5	5		C#	4.68%	+0.83%
6	6		Visual Basic	4.66%	+0.97%
7	7		JavaScript	2.87%	+0.62%
8	20	▲	R	2.79%	+1.97%
9	8	▼	PHP	2.24%	+0.17%
10	10		SQL	1.46%	-0.17%

<https://www.tiobe.com/tiobe-index/>

Who use Python?

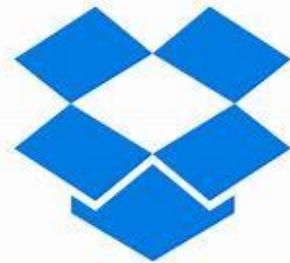
TensorFlow™



Google



PayPal™



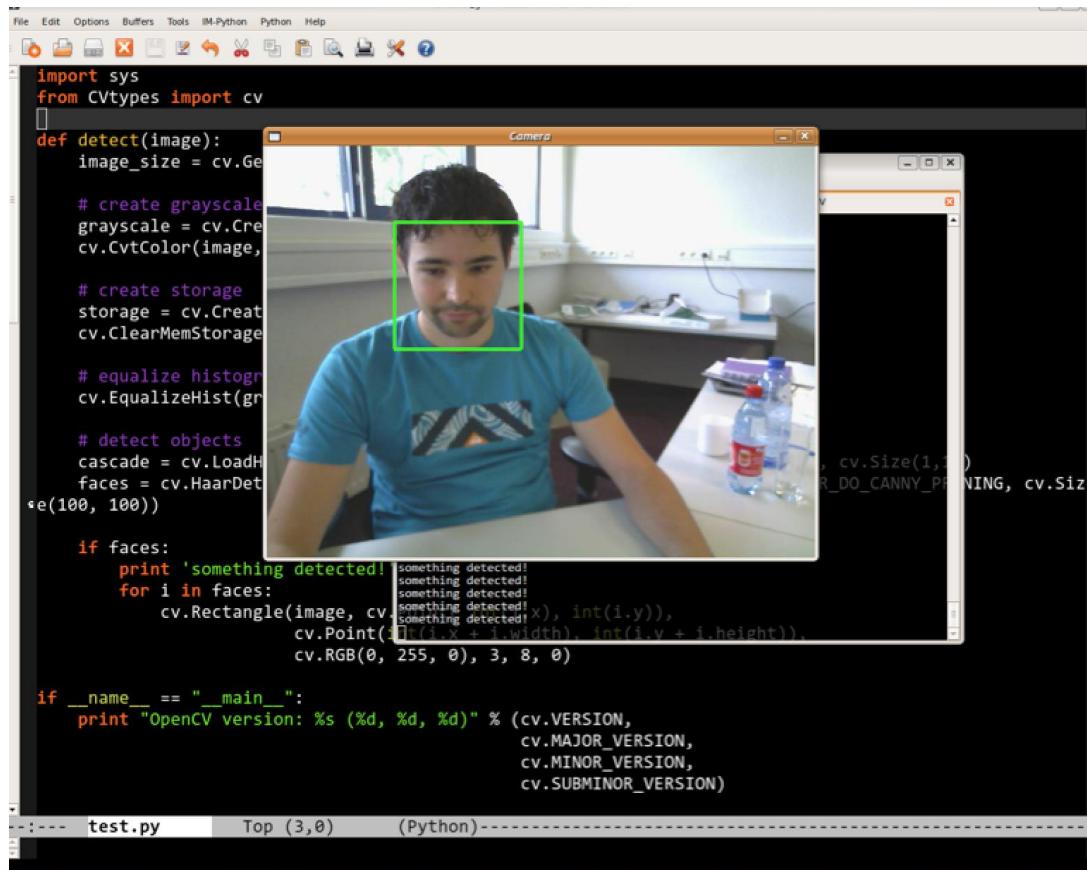
Dropbox

You Tube

Instagram

What can Python do?

- Face recognition with OpenCV library

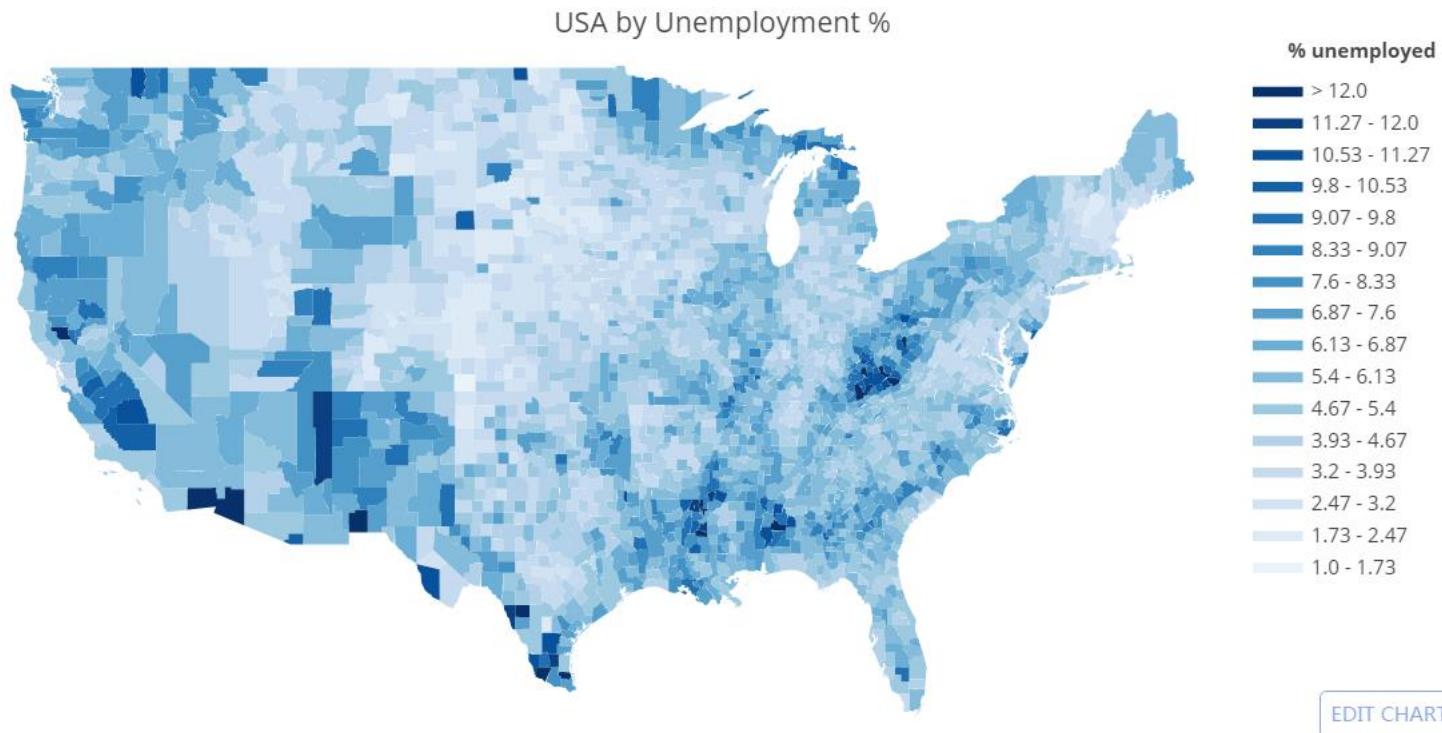


A screenshot of a Python IDE showing a script named `test.py`. The script uses the OpenCV library to detect faces in a video feed from a camera. A green bounding box highlights a detected face in the camera window. The terminal window shows the printed output of the detected faces.

```
File Edit Options Buffers Tools IM-Python Python Help
import sys
from CVtypes import cv
def detect(image):
    image_size = cv.GetSize(image)
    # create grayscale
    grayscale = cv.CreateImage(image_size, 8, 1)
    cv.CvtColor(image, grayscale, cv.CV_BGR2GRAY)
    # create storage
    storage = cv.CreateMemStorage(0)
    cv.ClearMemStorage(storage)
    # equalize histogram
    cv.EqualizeHist(grayscale, grayscale)
    # detect objects
    cascade = cv.LoadHaarClassifierCascade("haarcascade_frontalface_default.xml", -1)
    faces = cv.HaarDetectObjects(grayscale, cascade, storage, 1.1, 0, cv.CV_HAAR_DO_CANNY_PRUNING, cv.Size(1,1))
    if faces:
        print 'something detected!'
        for i in faces:
            cv.Rectangle(image, cv.Point(i[0][0], i[0][1]), cv.Point(i[0][0] + i[0].width, i[0][1] + i[0].height), cv.RGB(0, 255, 0), 3, 8, 0)
    if __name__ == "__main__":
        print "OpenCV version: %s (%d, %d, %d)" % (cv.VERSION,
                                                       cv.MAJOR_VERSION,
                                                       cv.MINOR_VERSION,
                                                       cv.SUBMINOR_VERSION)
:---- test.py Top (3,0) (Python)
```

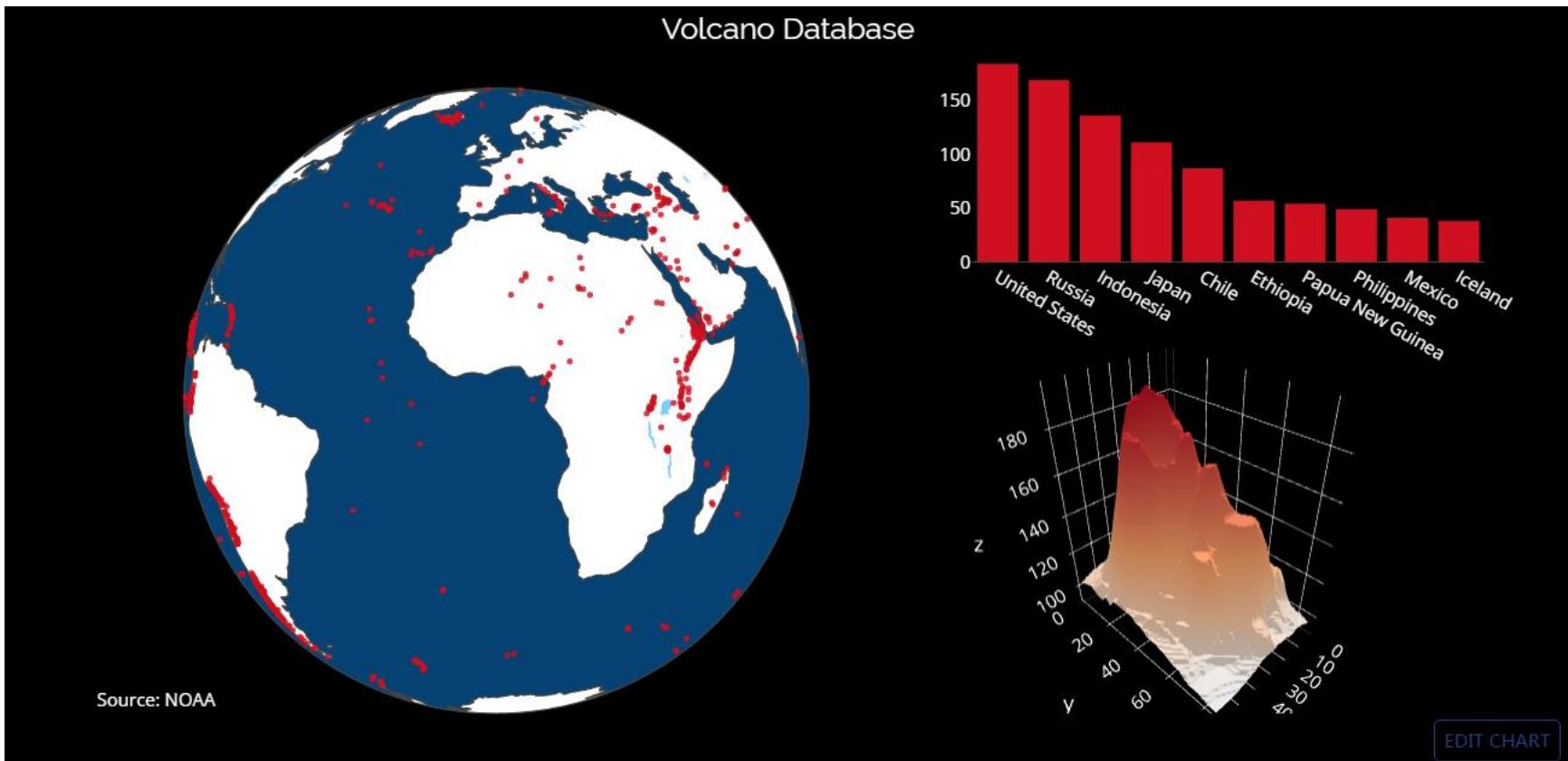
What can Python do?

- Interactive visualizations and data analysis, <https://plot.ly>



What can Python do?

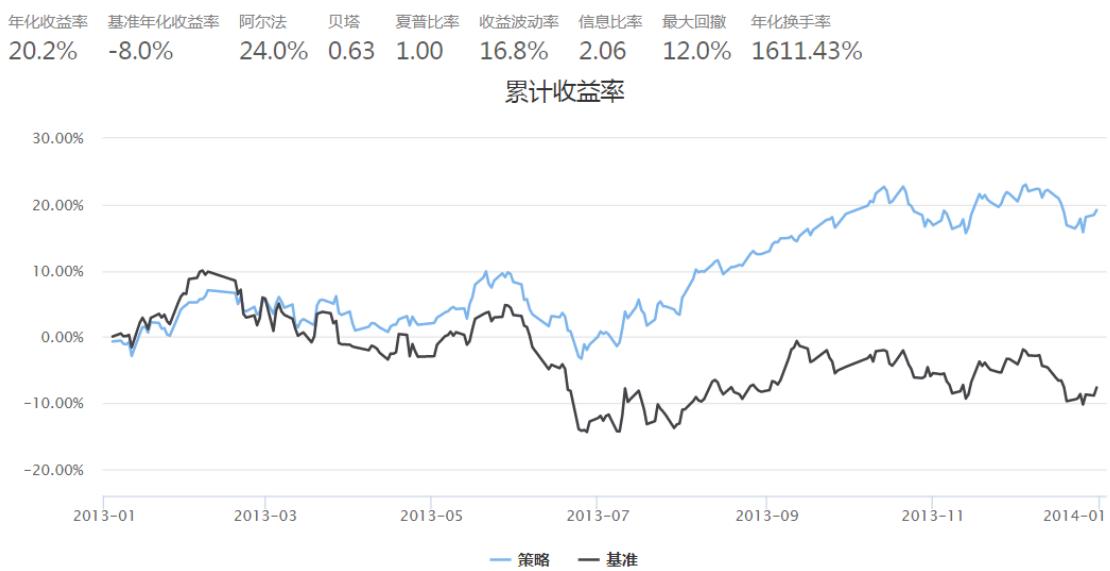
- Interactive visualizations and data analysis, <https://plot.ly>



What can Python do?

- Quantitative trading
 - <https://uquer.io>, write strategies and back-test them

```
1 import pandas as pd
2 import numpy as np
3 from sklearn.svm import SVC
4 import CAL
5 start = '2013-01-01'          # 回测起始时间
6 end = '2014-01-01'           # 回测结束时间
7 benchmark = 'HS300'           # 策略参考标准
8 universe = set_universe('HS300') # 证券池，支持股票和基金
9 capital_base = 100000         # 起始资金
10 freq = 'd'                   # 策略类型，'d'表示日间策略使用
11 refresh_rate = 20            # 调仓频率，表示执行handle_data
12
13 trainperiod='3M'#训练周期3个月
14 factor=['LFLO','NetProfitGrowRate']#选2个因子
15 cal = CAL.Calendar('China.SSE')#创建日历
16
17
18 def initialize(account):      # 初始化虚拟账户状态
19     pass
20
21 def handle_data(account):      # 每个交易日的买入卖出指令
22     preday=cal.advanceDate(account.current_date,trainperiod)
23
```



- A story about Bitcoin arbitrage system in 2014
 - <http://daily.zhihu.com/story/4831821>

What can Python do?

- Computational biology, bioinformatics
 - combines biology, computer science, information engineering, mathematics and statistics to analyze and interpret biological data
 - <https://biopython.org>, computations on gene sequences

What can Python do?

- Computational chemistry
 - uses methods of theoretical chemistry, incorporated into efficient computer programs, to calculate the structures and properties of molecules and solids.

Some active projects on Github

MDAnalysis / mdanalysis

★ 262

MDAnalysis is a Python library to analyze molecular dynamics trajectories.

molecular-dynamics python science trajectory-analysis molecular-simulation

Python

Updated Feb 12, 2019

16 issues need help

cclib / cclib

★ 115

Parsers and algorithms for computational chemistry logfiles

python computational-chemistry quantum-chemistry

Python

Updated Feb 12, 2019

Mariewelt / OpenChem

★ 96

OpenChem: Deep Learning toolkit for Computational Chemistry and Drug Design Research

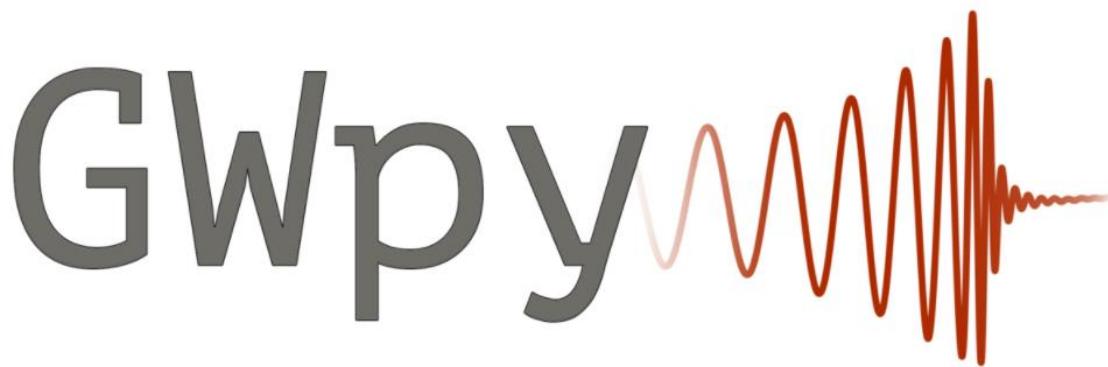
computational-chemistry computational-biology deep-learning deep-neural-networks



36

What can Python do?

- 2017 Nobel Prize in Physics
 - for decisive contributions to the LIGO detector and the observation of **gravitational waves**
 - <https://gwpy.github.io/> , tools and methods for studying data from gravitational-wave detectors



a python package for gravitational-wave astrophysics.

>>> help(gwpy)

What can Python do?

- Crawling web data
 - BeautifulSoup, Scrapy
 - E.g. Get rent price from 链家 and analyze the price trend
- Statistics, scientific computing
 - statistics, numpy, scipy, pandas
- Natural language processing
 - NLTK, Jieba; to understand human language
- Machine learning/ Deep learning
 - SKLearn, Tensorflow, Pytorch

Social media data mining and e-commerce campaign

Twitter

 Retweeted by Chelsea FC
 André Schürrle @Andre_Schuerlle - Aug 18
 Great start in the saison...come on blues!!! #cfc



eBay

chelsea t-shirt

All Listings Auction Buy It Now

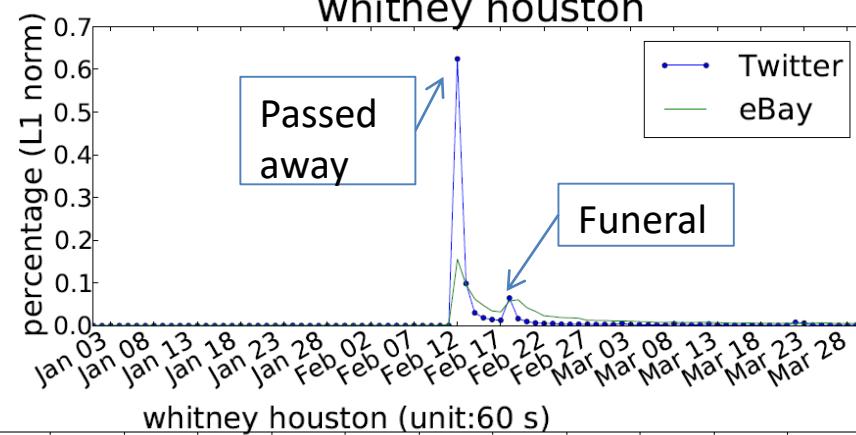
Sort: Best Match View: 

3,311 results for chelsea t-shirt 

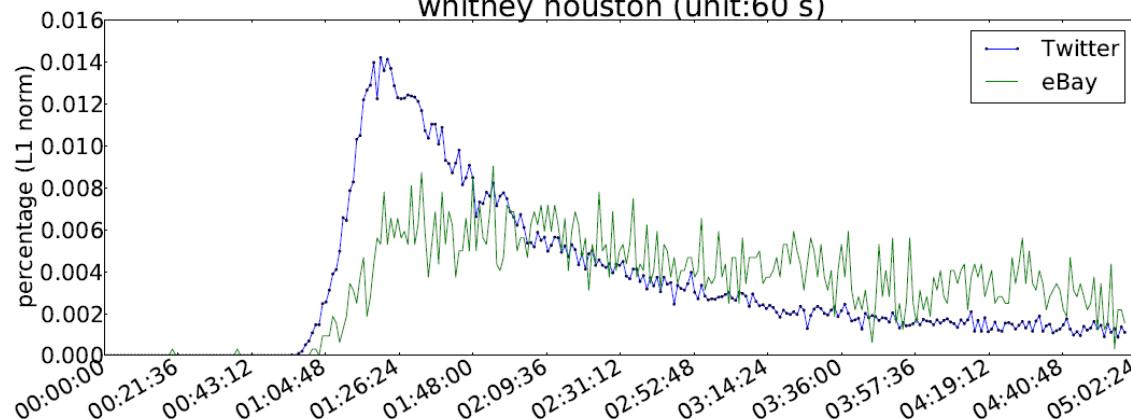
	7 Photos	Chelsea FC Football Soccer T Shirt J...
	5 Photos	Adidas Chelsea Away Soccer Jersey ...
	7 Photos	Chelsea FC Football Soccer T Shirt J...

\$17.99 to \$22.99 Buy It Now \$13.00 0 bids \$16.99 to \$22.99 Buy It Now

correlations

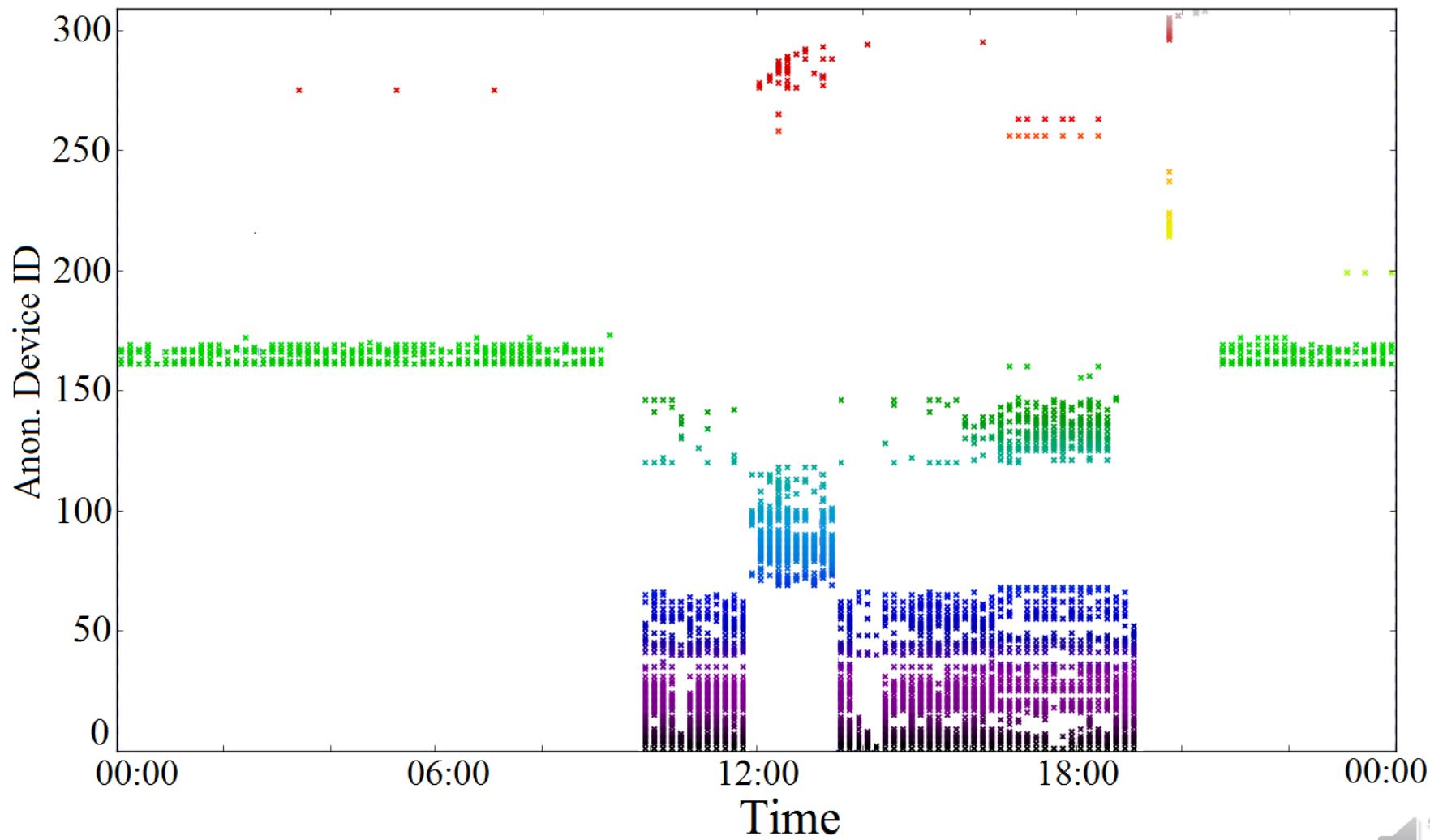


lags
(4.8 hours)

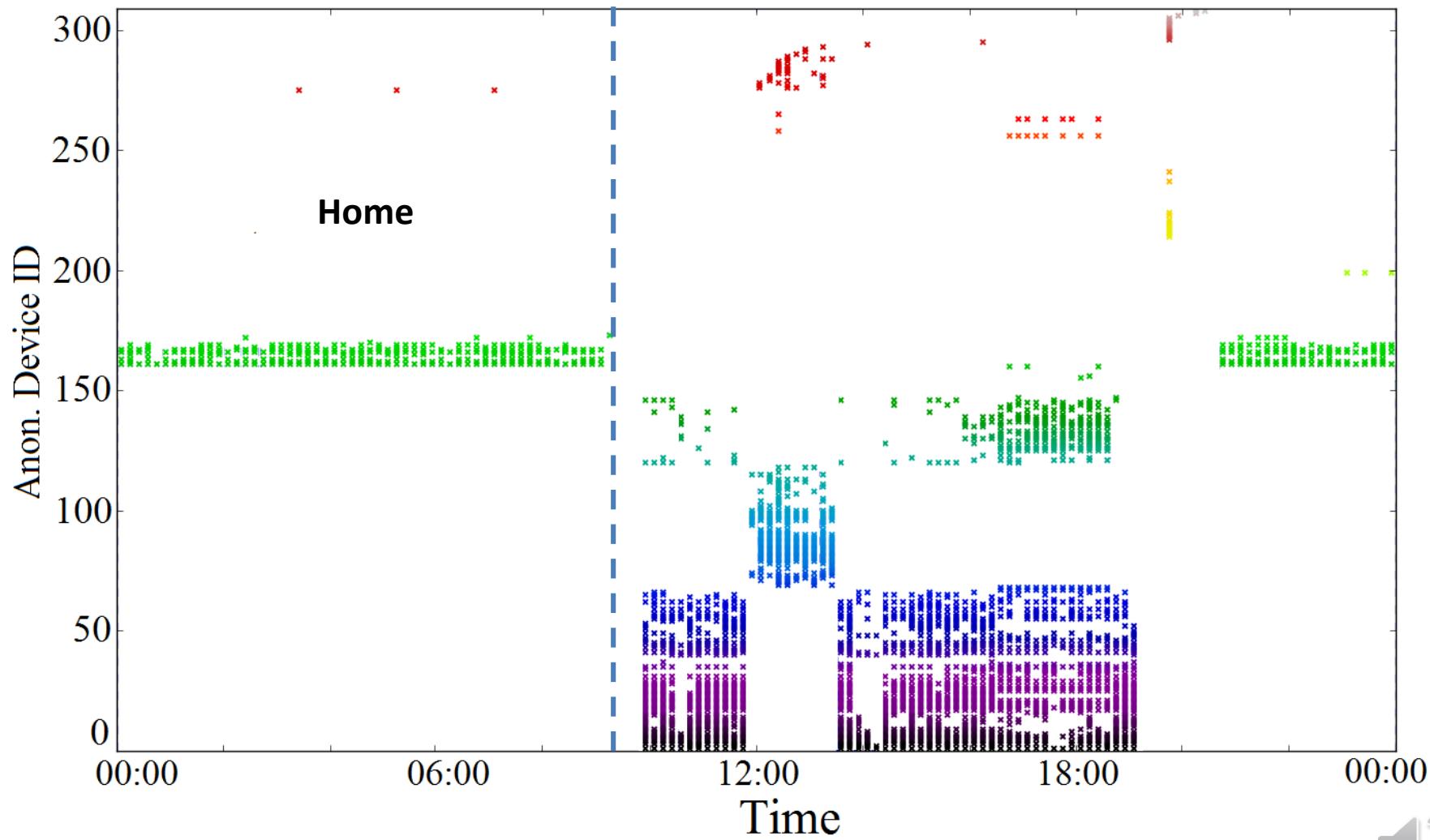


Mining mobile user behaviors

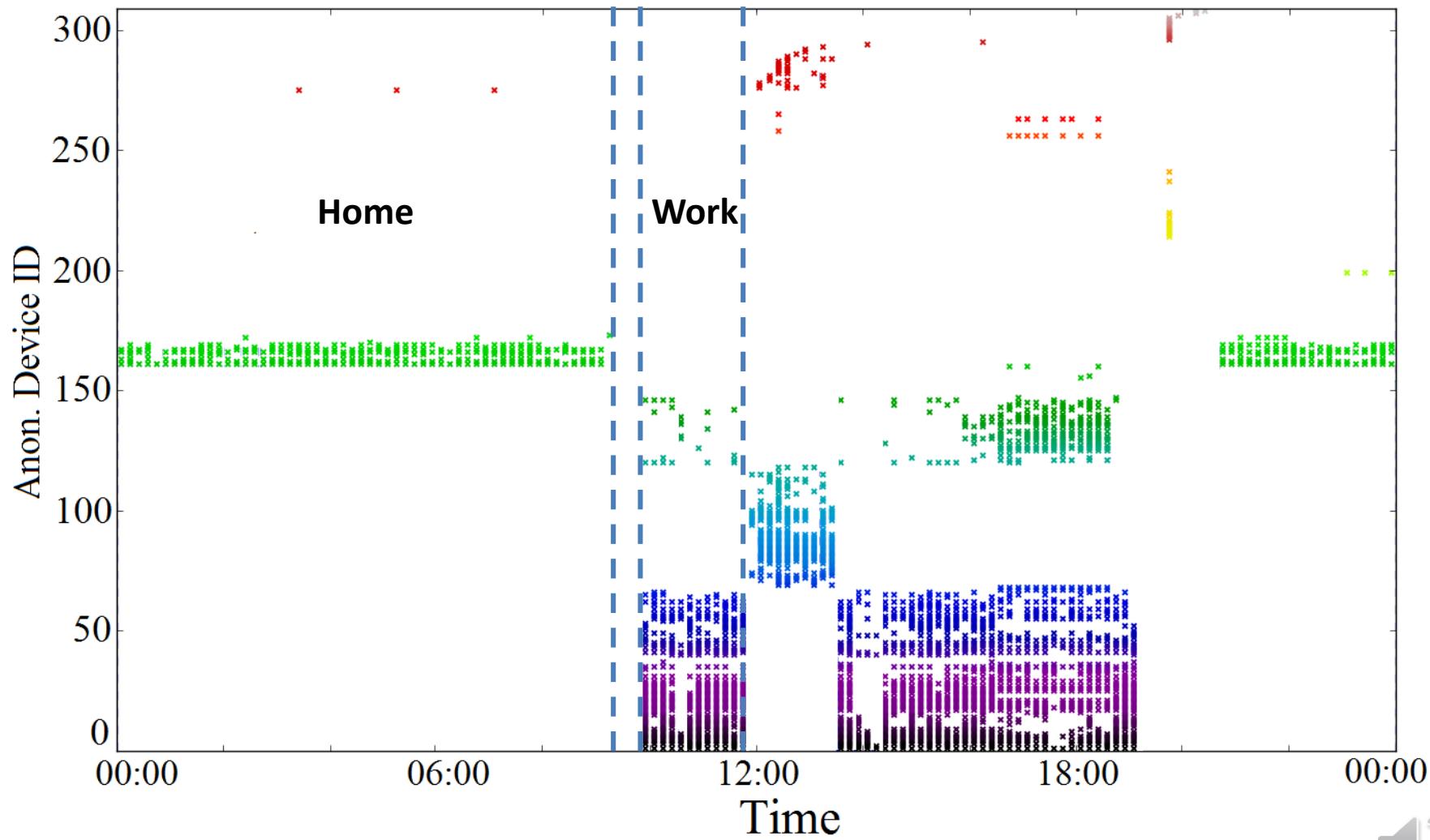
- WiFi device ID discovered by a phone, in a day



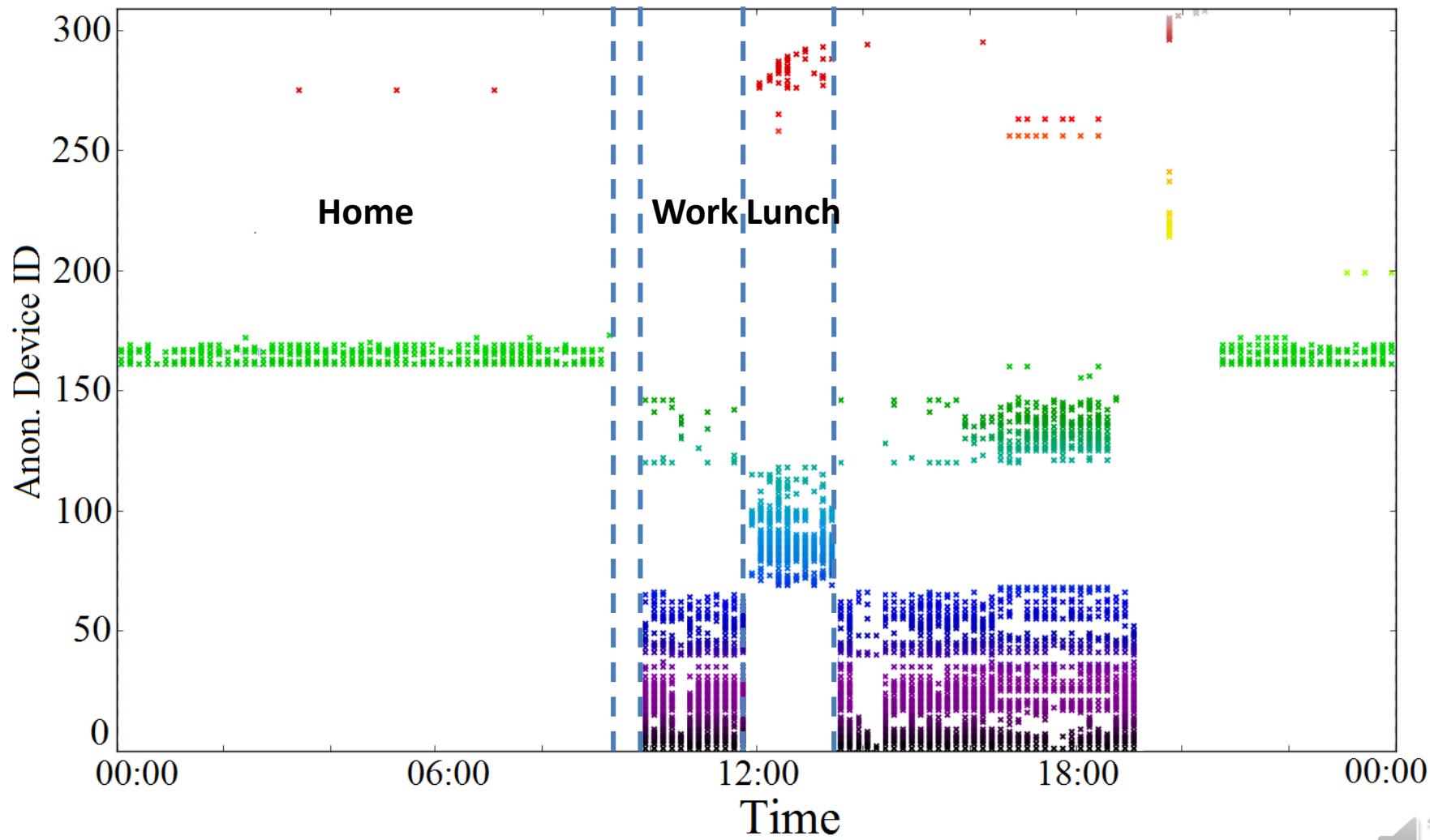
- WiFi device ID discovered by a phone, in a day



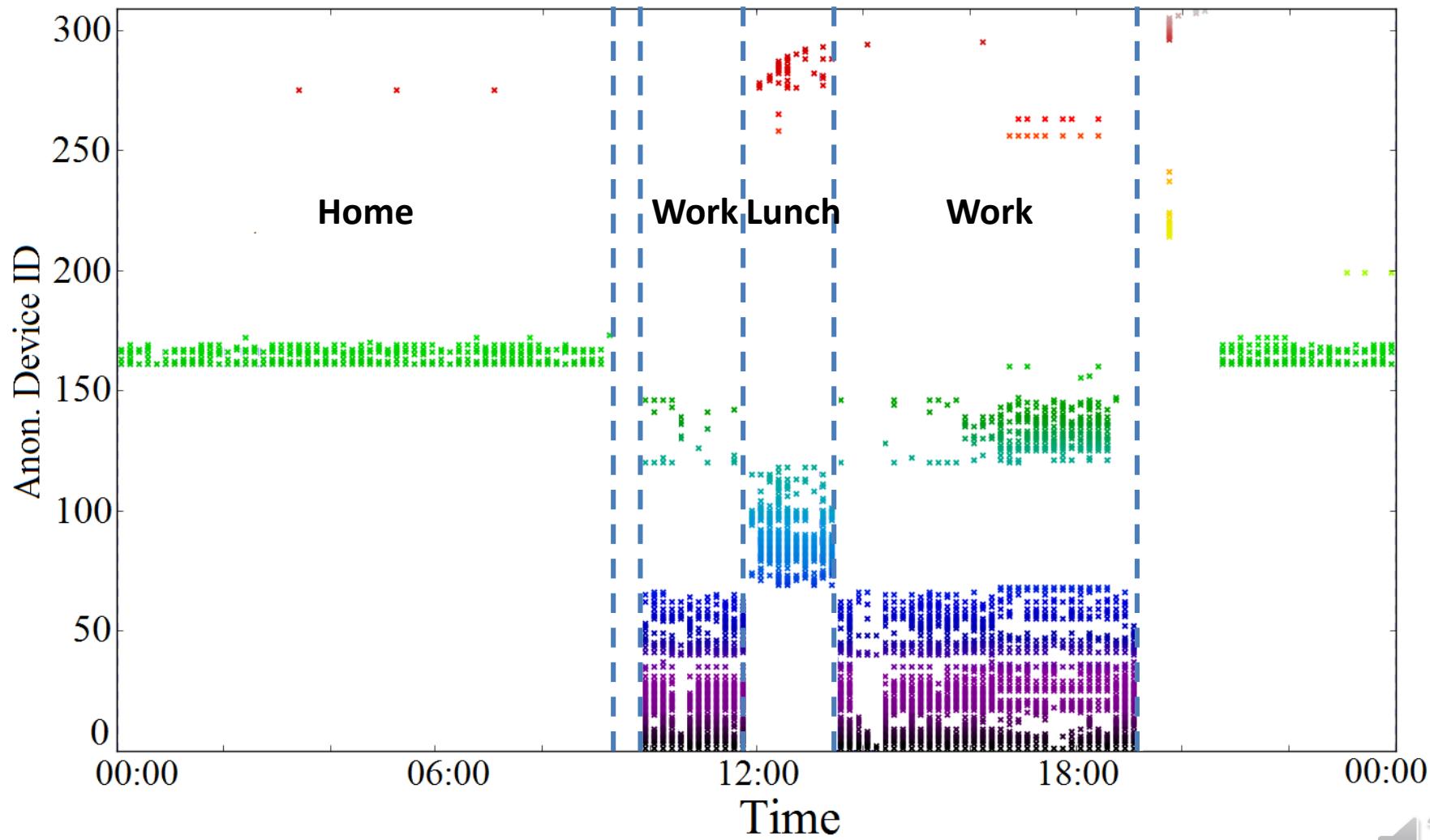
- WiFi device ID discovered by a phone, in a day



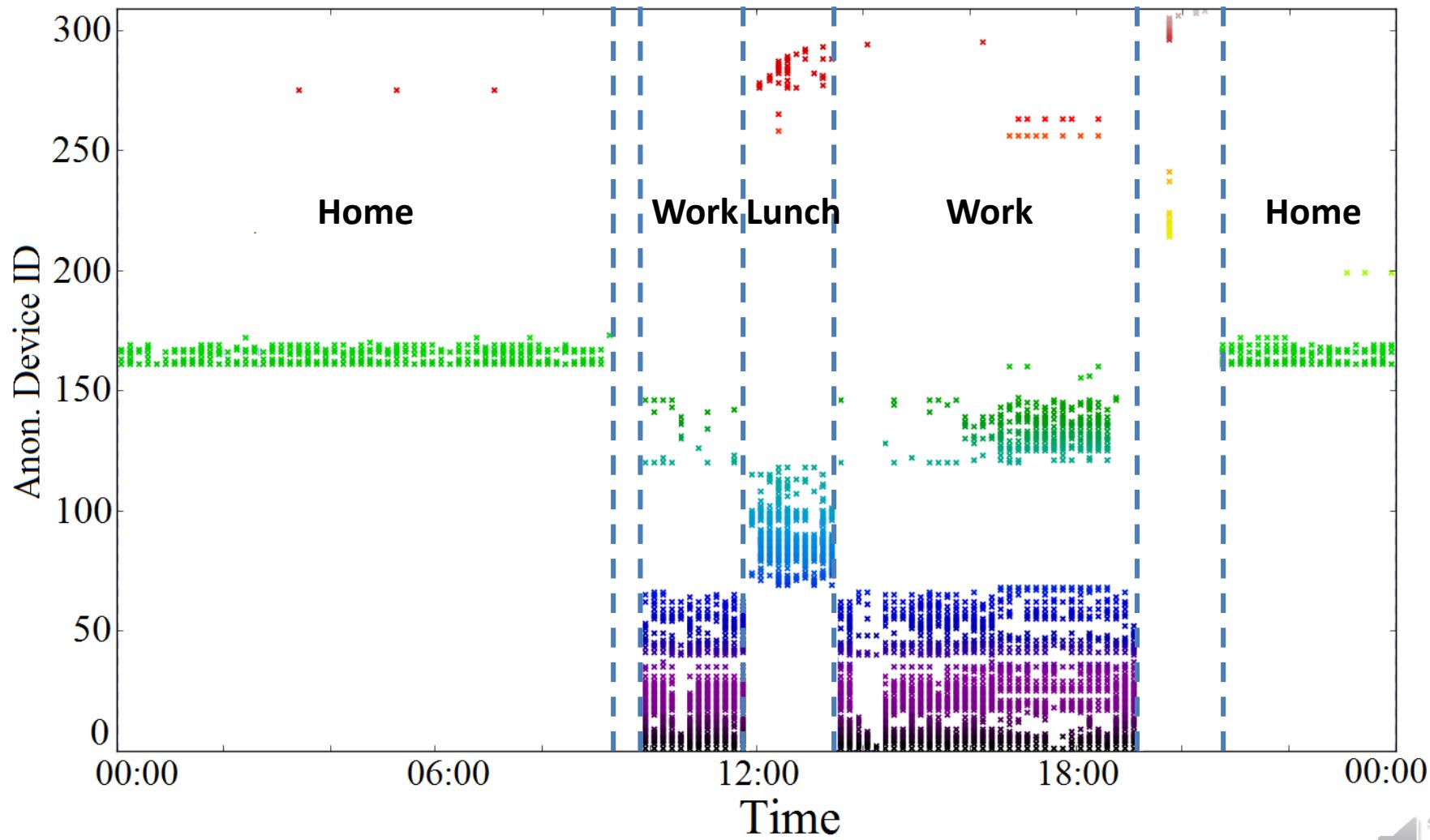
- WiFi device ID discovered by a phone, in a day



- WiFi device ID discovered by a phone, in a day

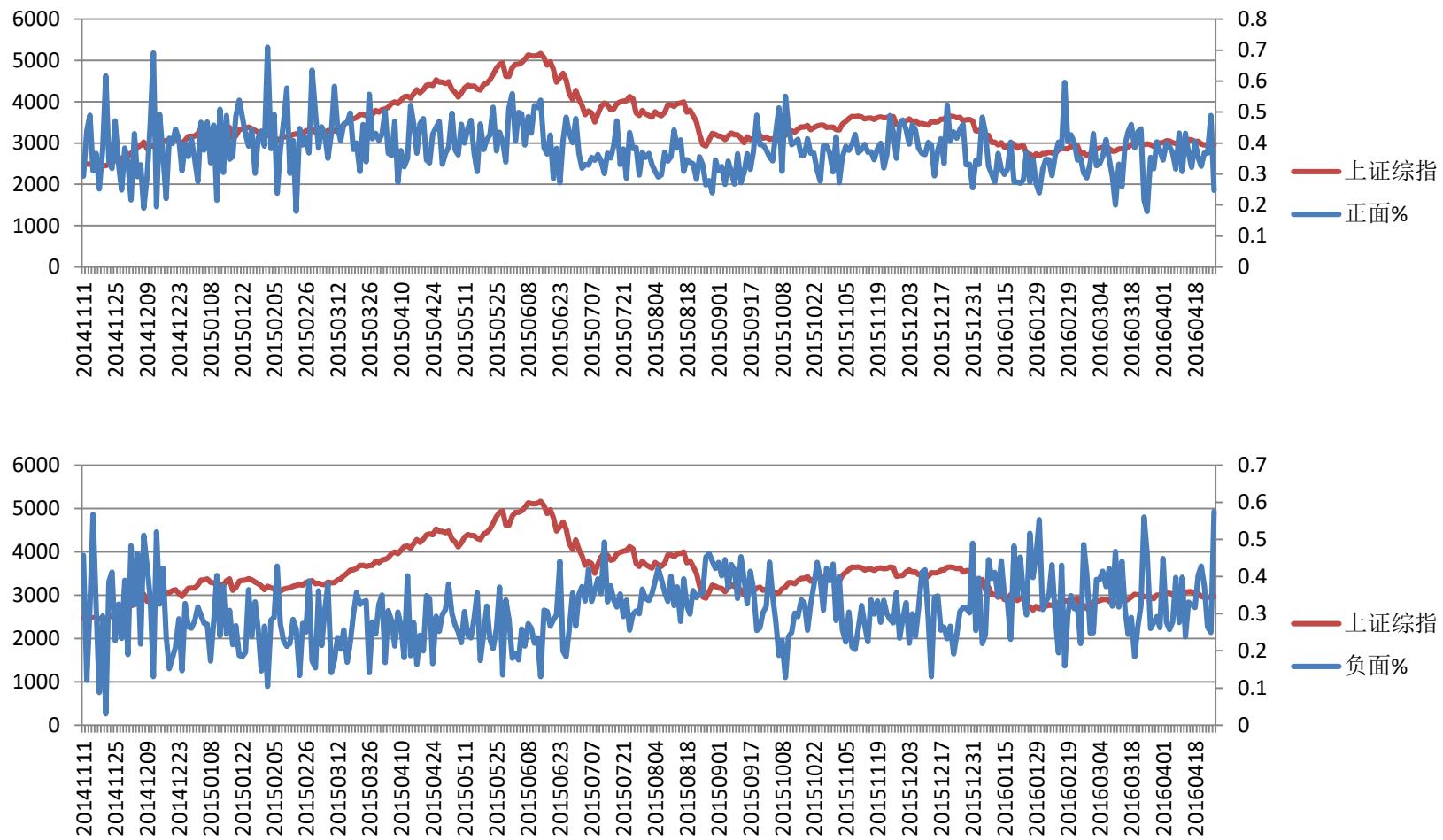


- WiFi device ID discovered by a phone, in a day



Social media and stock market

Positive/negative social media sentiment and the stock market



How to get Python

- **Option 1**, install Anaconda
 - Includes core Python environment, IDE (Integrated Development Environment). All in one.
 - Data science and visualization packages
 - Easy to manage (add, remove, update) packages
 - **Python 3.7** version for this course. **Anaconda3 version 2018.12**
 - <https://www.anaconda.com/products/individual>

How to get Python

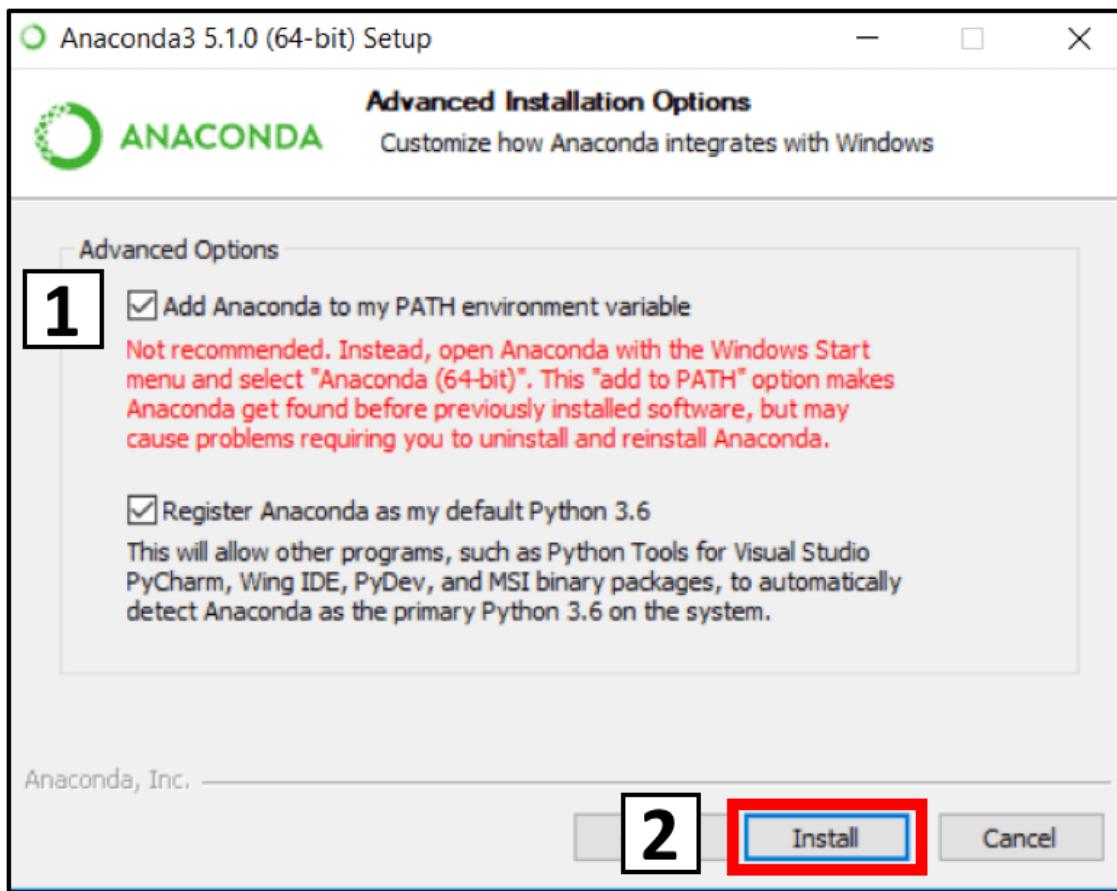
- For Windows
 - Check your Windows bit, 64bit or 32bit, most often 64bit.



- Download and install 64bit or 32bit Anaconda accordingly
 - https://repo.anaconda.com/archive/Anaconda3-2018.12-Windows-x86_64.exe 64bit
 - <https://repo.anaconda.com/archive/Anaconda3-2018.12-Windows-x86.exe> 32bit

How to get Python

- Recommend you check the first box



How to get Python

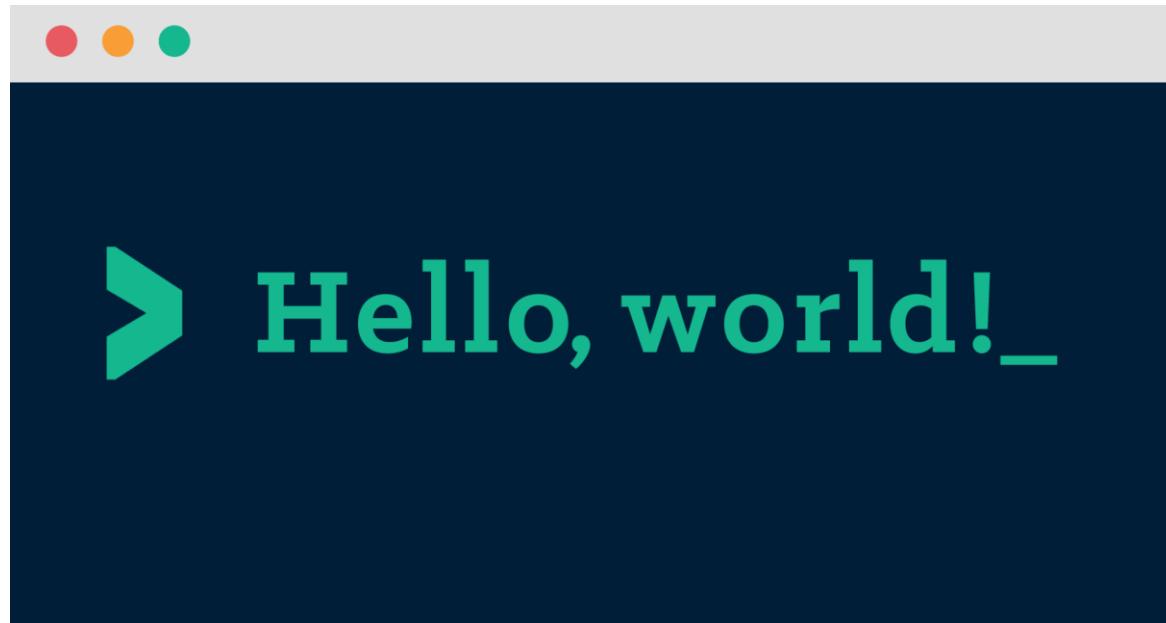
- For Mac OS
 - https://repo.anaconda.com/archive/Anaconda3-2018.12-MacOSX-x86_64.pkg
- For Linux
 - <https://www.anaconda.com/distribution/#linux>
 - Install according to your operation system bit and CPU (x86 or Power8/9)
- After installation
 - Try the **Spyder** IDE, included in Anaconda
 - Or in Anaconda Prompt/command line/terminal, enter Python (depending on whether you have added Anaconda to your system path during installation)

How to get Python

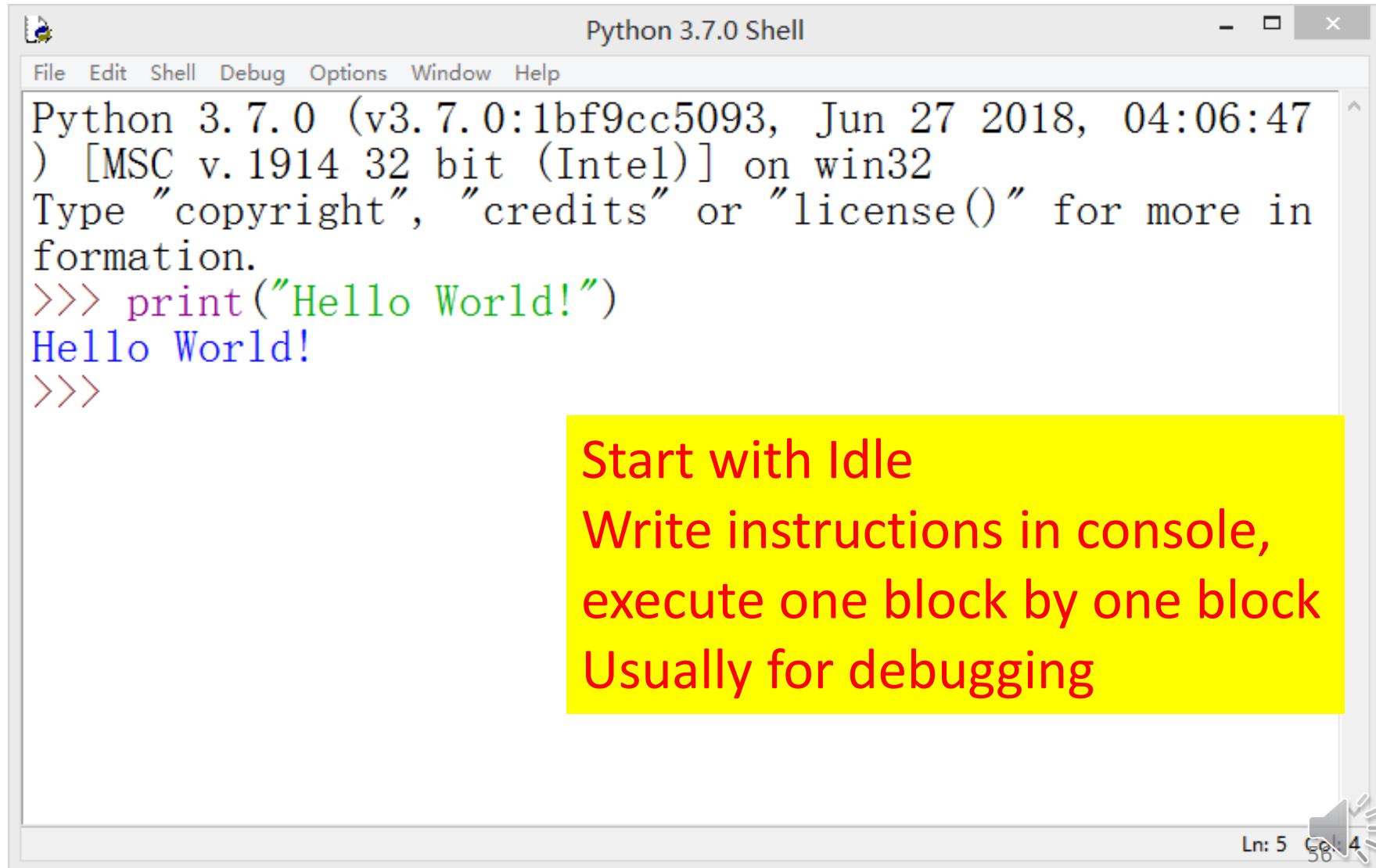
- **Option 2, install Python from python.org**
 - Download python 3.7.1 at
www.python.org/downloads
 - 32bit/64bit, Windows, Linux/UNIX, Mac OSX depends on your computer
 - Run python or idle in Windows
 - Type in python3 or python on Linux or Mac OSX
 - Two programming modes in Python
 - Interactive
 - Batch

How do I know I installed everything correctly?

- A tradition among programmers
 - always test installation by writing the same program



Step 1: Interactive programming



The screenshot shows the Python 3.7.0 Shell window. The title bar reads "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47)
) [MSC v. 1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

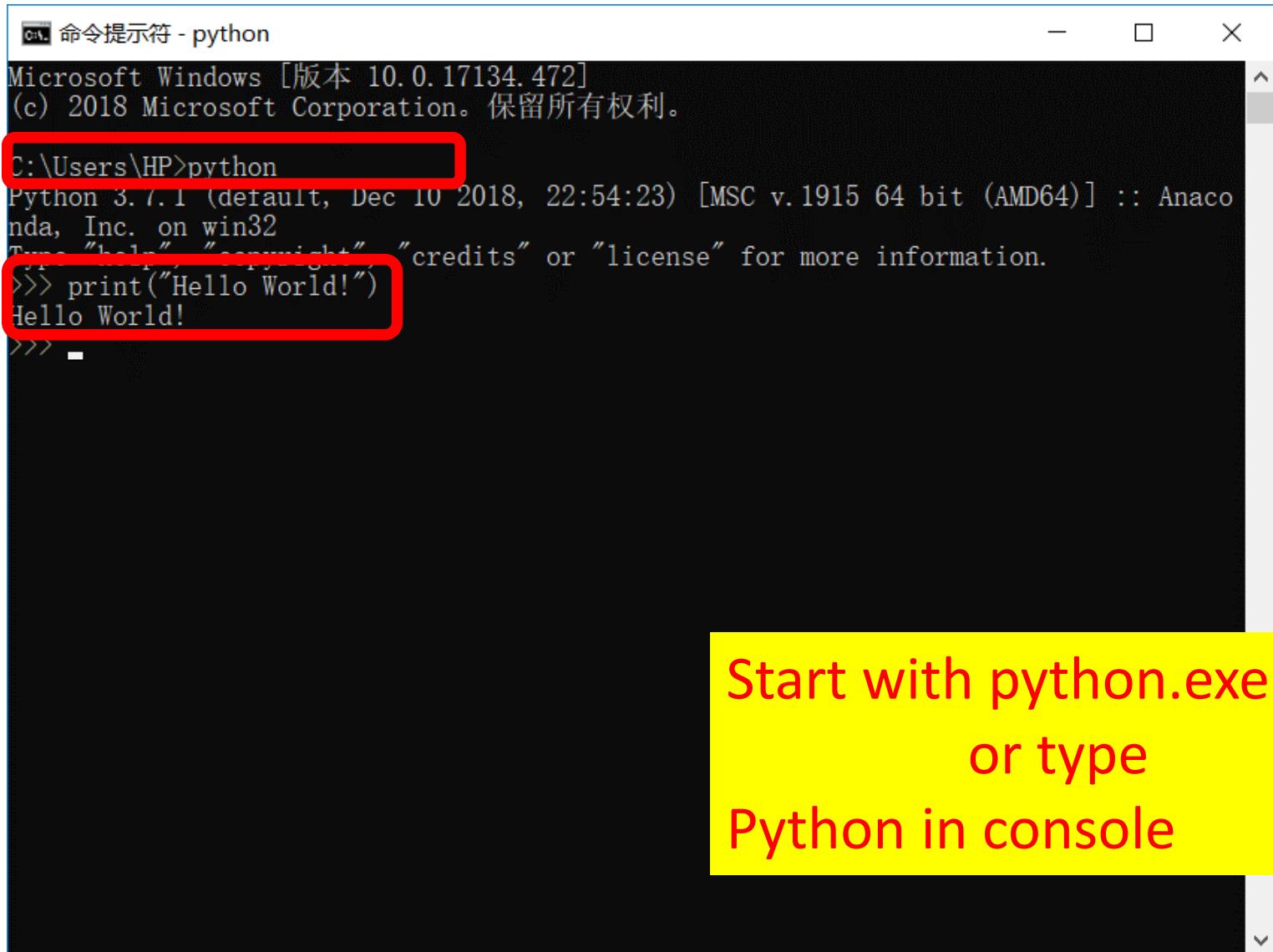
>>> print("Hello World!")
Hello World!
>>>
```

A yellow callout box in the bottom right corner contains the following text in red:

Start with Idle
Write instructions in console,
execute one block by one block
Usually for debugging

Ln: 5 Col: 4

Step 1: Interactive programming



```
命令提示符 - python
Microsoft Windows [版本 10.0.17134.472]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\HP>python
Python 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World!")
Hello World!
>>>
```

Start with `python.exe`
or type
`Python` in console

Step 2: Batch programming

Comment starts with #

```
#hello.py  
  
print("Hello World!")  
print("Hello World!!")
```

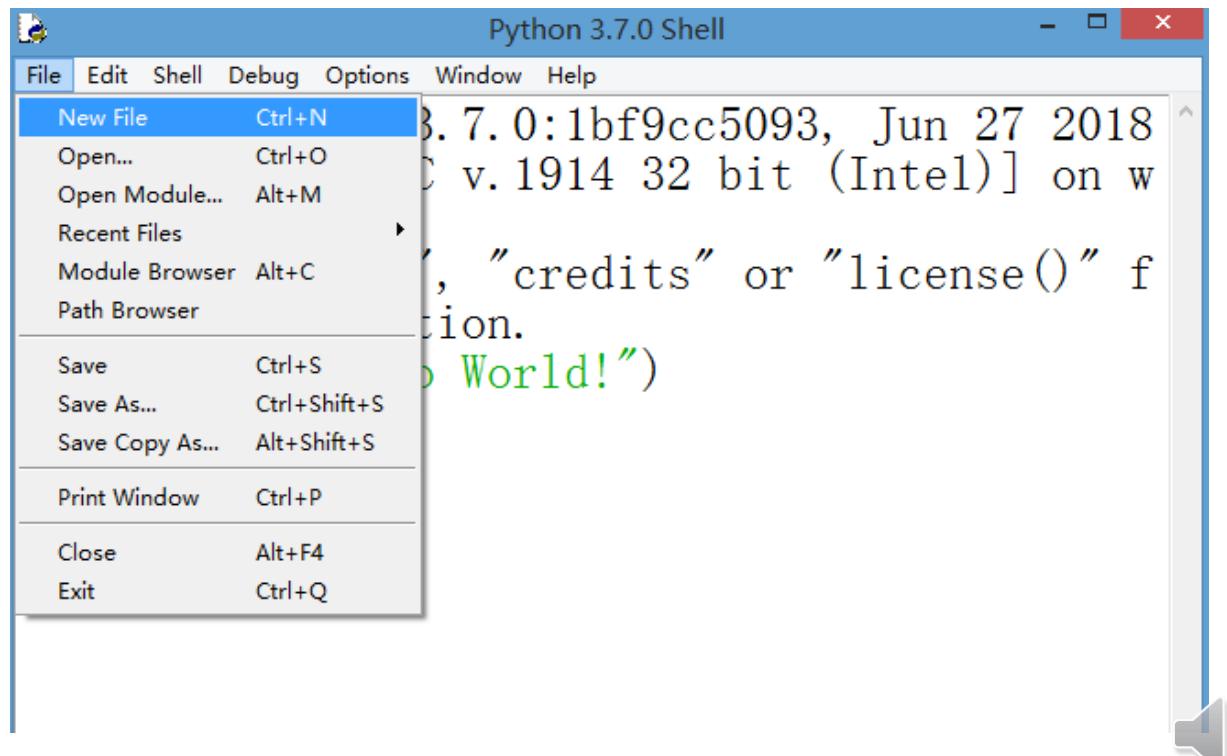
**Write instructions in a file hello.py,
execute the file by
python hello.py**



Step 2: Batch programming

In IDLE:

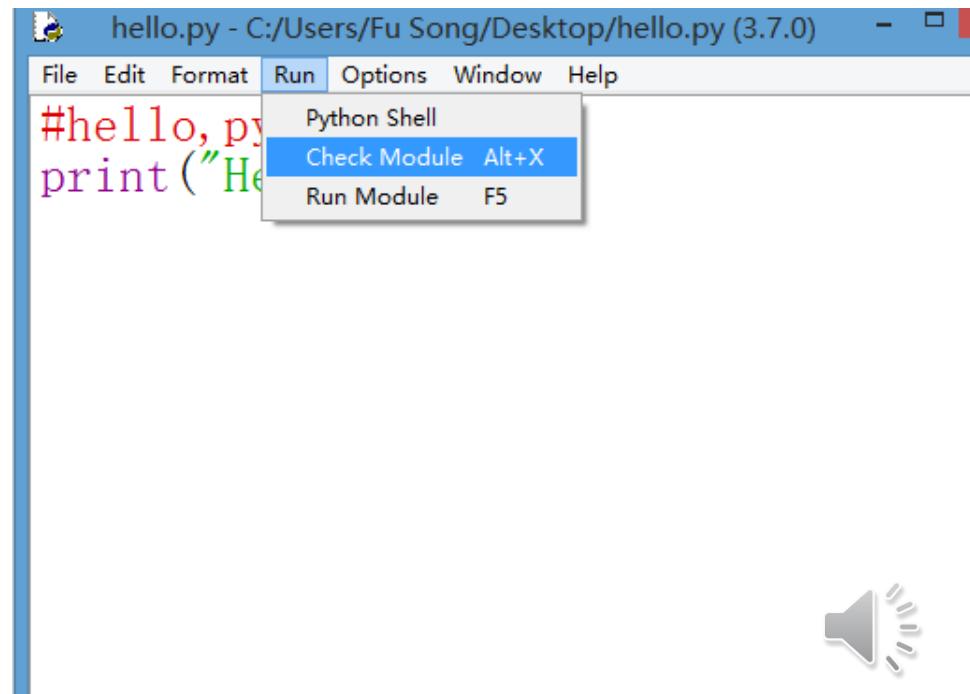
1. “File”==>“New File” create a file,
2. write your python program
3. save file as hello.py



Step 2: Batch programming

In IDLE:

1. “File”=>“New File” create a file,
2. write your python program
3. save file as filename.py
4. “Run”=>“Check Module”
check syntax
5. “Run”=>“Run Module”
**run the program and
print the result in IDLE**



In Spyder (Anaconda)

The screenshot shows the Spyder IDE interface with two main sections highlighted by red arrows:

- Batch programming**: Points to the code editor where a "print" statement is highlighted. A red box surrounds the entire code area, and another red arrow points to a yellow box labeled "Batch programming".
- Interactive programming**: Points to the IPython console window where a "print" command is run and its output is shown. A red box surrounds the console output, and another red arrow points to a yellow box labeled "Interactive programming".

Code Editor (hello.py)

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Feb 12 21:27:03 2019
4
5 @author: HP
6 """
7
8 print("Hello World!")
```

IPython Console

```
In [1]: print('Hello World!')
Hello World!
```

Permissions: RW | End-of-lines: CRLF | Encoding: UTF-8 | Line: 8 | Column: 22 | Memory: 50%

Print in Python

- print a string: string needs to be surrounded by double or single quotes
 - `print("text")` or `print('text')`
- `print()` prints a blank line
- <https://docs.python.org/3/library/functions.html#printf>

Print in Python

```
#hello.py  
print('Hello World! ')  
print("Hello World!")
```

Output

```
Hello World!  
Hello World!  
>>>
```

What if we want the output to be:

```
'Hello World!'  
"Hello World!"  
>>>
```

Print in Python

- We will use escape character (\)

```
#hello.py
print('\'Hello World!\'')

print("\\"Hello World!\\\"")

print('\"Hello World!\"')

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

```
'Hello World!'
"Hello World!"
"Hello World!"
'Hello World!'
>>>
```

Print in Python

Python escape characters:

For this	Use this	Setting x to:	Printing x will yield:
'	\'	'Don\'t do that'	Don't do that
"	\"	"She said \"hi\""	She said "hi"
\	\\	"Backslash: \\\"	Backslash: \
[newline]	\n	"1\n2"	1 2
[carriage return]	\r	"1\r2"	2 overwrites the 1
[horizontal tab]	\t	"1\t2"	1 2
[backspace]	\b	"12\b3"	13
[16 bit unicode]	\uxxxx	"Katakana a: \u30A1"	Katakana a: ア
[32 bit unicode]	\Uxxxxxxxxx	"Katakana a: \u000030A1"	Katakana a: ア

Print in Python

```
#hello.py
print('I\'ll say hello!')
print("I'll say hello!")
print("I'll say\n hello!")
```

```
I'll say hello!
I'll say hello!
I'll say
hello!
>>>
```

Comments

- **Syntax:**

- **# comment text (one line)**

- **''' block**

- comment ''' (span over lines, use in pairs)**

- **test.py**

```
'''  
Fall 2020, SIST  
This program prints important messages.  
'''  
print("Hello, world!")  
Print("")                      # blank line  
print("I'll say hello!")
```

Output:

Hello, world!

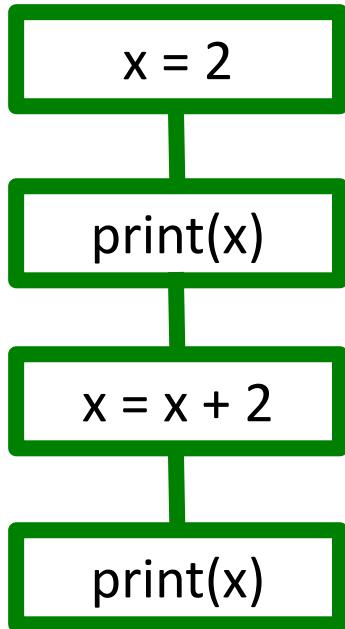
I'll say hello!



Program steps or flow

- Like a recipe or installation instructions, a program is a **sequence** of steps to be done in order.
- Some steps are **conditional** - they may be skipped.
- Sometimes a step or group of steps are to be **repeated**.

Sequential steps



Program:

`x = 2`

`print(x)`

`x = x + 2`

`print(x)`

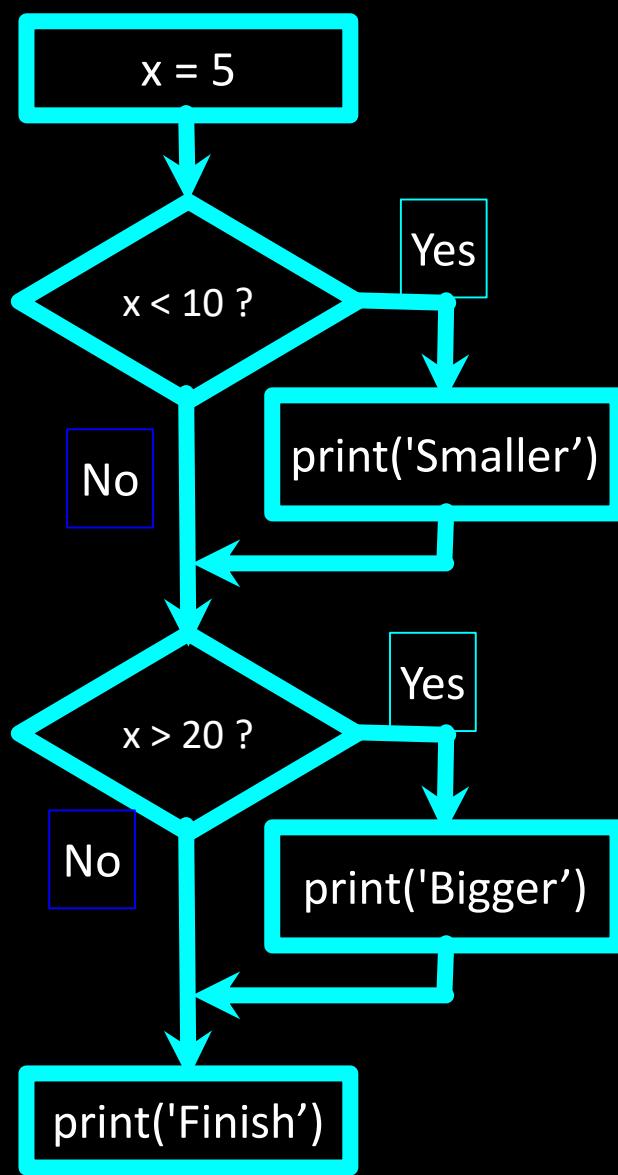
Output:

`2`

`4`

When a program is running, it flows from one step to the next. As programmers, we set up “paths” for the program to follow.

Conditional steps



Program:

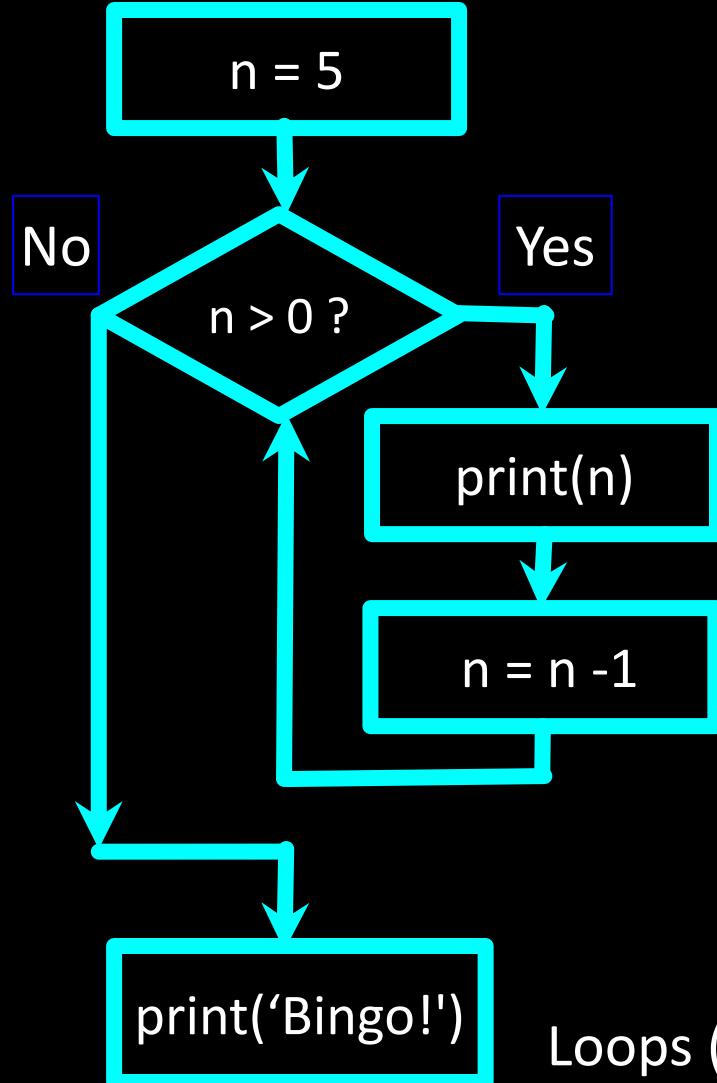
```
x = 5  
if x < 10:  
    print('Smaller')  
if x > 20:  
    print('Bigger')  
print('Finish')
```

Output:

Smaller
Finish



Repeated steps



Program:

```
n = 5
while n > 0 :
    print(n)
    n = n - 1
print('Bingo!')
```

Output:

5
4
3
2
1
Bingo!

Loops (repeated steps) have **iteration variables** that change each time through a loop. Often these **iteration variables** go through a sequence of numbers.



Readings (recommended)

- The Python Tutorial
 - Chapter 3: An Informal Introduction to Python

Assignments and Tutorial 1

- Will release logistics assignment (1% of the SI100B course) and assignment 0 (zero point) today, due in a week
- logistics assignment on academic integrity, due: Sept. 16 23:59 (next Wed.) on Gradescope
- Assignment 0 helping you to get familiar with programming assignment submission. Due: Sept. 16 23:59 (next Wed.) on PIEGEN;

TA Tutorial

- **Time and Location for First Tutorial: 8PM Friday 9/11, TC201**
- Help with setting up the Python environment
- Help with writing and running first Python programs
- Basics on command line and IDE usage
- Help with HW submission (assignment 0)

Recap

- How a programs runs on a basic computer
- What is Python
- How to get Python
- Run a Python program
- Basics about print and comments
- Basic control flow