

# 程序设计概述

先来玩个游戏?

# 8人过河问题

- ▶ 现有1父，1母，2儿子，2女儿，1警察，1小偷
- ▶ 只有一艘摆渡船，最多容纳2人。
- ▶ 限制条件：
  - ▶ 1. 父亲不能和女儿在一起，但母亲在除外。
  - ▶ 2. 母亲不能和儿子在一起，但父亲在除外。
  - ▶ 3. 小偷一定和警察在一起，但如果四旁无人，小偷可以一人独处。
  - ▶ 4. 小孩子不会撑船。
- ▶ 问：如何过河？

# 程序设计

## 抽象现实问题

- 设A为河一边的状态，用集合{A} 分别表示父亲，母亲；儿子1；儿子2；女儿1；女儿2；警察；小偷的状态。=1 表示人在，=0表示人不在。
  - 则初始A=(1,1,1,1,1,1,1,1)
- 设B为河另外一边的状态数组，其含义与A数组相同。
  - 则初始B=(0,0,0,0,0,0,0,0)。
- 完成过河任务后，A=(0,0,0,0,0,0,0,0); B=(1,1,1,1,1,1,1,1)

# 程序设计

## 设定约束条件

根据题意：河两边不能出现的状态有：

(1,0,x,x,1,x,x,x);(1,0,x,x,x,1,x,x); (0,1,1,x,x,x,x,x); (0,1,x,1,x,x,x,x); (1,x,x,x,x,x,0,1); (x,1,x,x,x,x,0,1);  
(x,x,1,x,x,x,0,1); (x,x,x,1,x,x,0,1); (x,x,x,x,1,x,0,1); (x,x,x,x,x,1,0,1)

船上的可选方案S为：

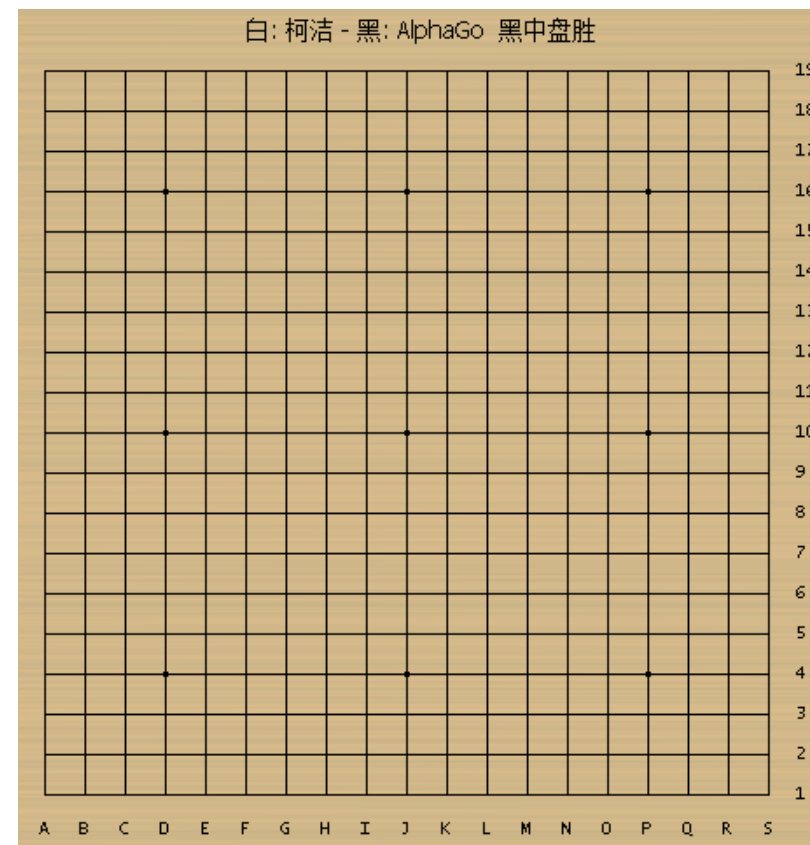
(1,0,0,0,0,0,0,0);(1,1,0,0,0,0,0,0);(1,0,1,0,0,0,0,0);(1,0,0,1,0,0,0,0);(1,0,0,0,0,0,1,0); (0,1,0,0,0,0,0,0);  
(0,1,0,0,1,0,0,0); (0,1,0,0,0,1,0,0); (0,1,0,0,0,0,1,0); (0,0,0,0,0,0,1,1); (0,0,0,0,0,0,1,0)

# 程序设计

- 选择语言完成方案撰写（作文+解题）
  - 从方案集S中依次选出可行方案，根据方案从A状态对B状态进行转换
  - 从方案集S中依次选出可行方案，根据方案从B状态对A状态进行转换
  - 不断尝试状态集和方案集的排列组合，直到追求状态实现
  - 给人类主子报喜，输出方案
- 计算机根据语言，运行求解
  - 想错/写错语言时的修改，称之为debug

# 程序设计

- ▶ 编程最重要的目的是解决问题，这意味着...
  - ▶ 发现和清晰地定义问题
  - ▶ 创造性地思考可执行的解决方案
  - ▶ 以及准确地表达解决方案的能力





# $3^{361}$ ，或者 $361!$

游戏	状态空间复杂度	游戏树复杂度
井字棋	$10^4$	$10^5$
国际跳棋	$10^{21}$	$10^{31}$
国际象棋	$10^{46}$	$10^{123}$
中国象棋	$10^{48}$	$10^{150}$
五子棋	$10^{105}$	$10^{70}$
围棋	$10^{172}$	$10^{360}$

1. 围棋有361格，每个格子有3种状态（黑，白，空），即 $3^{361}$ （因为 $\log_{10} 3 = 0.477$ ，约等于 $10^{0.477 \times 361} = 10^{172}$ ），但这样不知道如何达到某种状态的
2. 看过程：相当于361阶乘，或者361的全排列
  - $361!$ 约等于 $3 \times 10^{287}$ （或者 $2^{888}$ ），约等于  $10^{[(2+0)/2 \times 360]} = 10^{360}$

# 一旦可以清晰表达，我们可以将方案转变为程序

- 程序是指一组定义如何进行计算的指令集合。这里的“计算”可能是数学计算（如解方程）或符号运算（如搜索和替换）。通常包括：
  - 输入：从键盘、文件或其他设备中获取数据；
  - 输出：将数据显示到屏幕或者发送文件到其他设备；
  - 数学/逻辑：进行基本数学运算操作，如加法和减法等；
  - 判断和执行：检查条件，执行相应的代码；
  - 循环反复：重复检查并执行其中每个环节。
- 几乎所有的程序，无论多么复杂，也都是由类似上面的这些指令组成的，因此我们可以不断分解任务，逐一操作。

# 因此，本课目标

- ▶ 初步理解计算机程序设计和高级语言编程的一般方法和逻辑思考过程
- ▶ 初步理解如何将实际问题转化成计算机语言表达的模型
- ▶ 提高尝试和创新意识，为今后学习其他计算机技术和解决实际问题打好基础
- ▶ 掌握Python语言的基本语法，能使用Python语言编写完整的程序

# 对于管院同学

- ▶ 为什么商学院的学生应当学习更多信息技术（如编程）？
- ▶ 管理学院的我应该学习哪个语言？
- ▶ 我应该花多少时间学习编程？
- ▶ 学编程和学习其他课有什么不一样？

# 2022年度泰晤士高等教育 中国学科评级结果(共82个学科)



来源:泰晤士高等教育  
制图:中国教育在线

2022-5-11



## 管理科学与工程(共54所高校)

评级

高校名称

A+

复旦大学 南京大学 上海交通大学 清华大学  
浙江大学

A

北京航空航天大学 华中科技大学 四川大学  
天津大学 同济大学 西安交通大学

A-

重庆大学 哈尔滨工业大学 中国人民大学  
南开大学



## 工商管理(共60所高校)

评级

高校名称

A+

复旦大学 南京大学 北京大学 上海交通大学  
清华大学 浙江大学

A

北京师范大学 哈尔滨工业大学 华中科技大学  
南开大学 中国人民大学 中山大学 同济大学  
武汉大学 西安交通大学

A-

中南大学 重庆大学 华东师范大学 四川大学  
天津大学



## 统计学(共51所高校)

评级

高校名称

A+

中南大学 复旦大学 北京大学 上海交通大学  
清华大学 中国科学技术大学

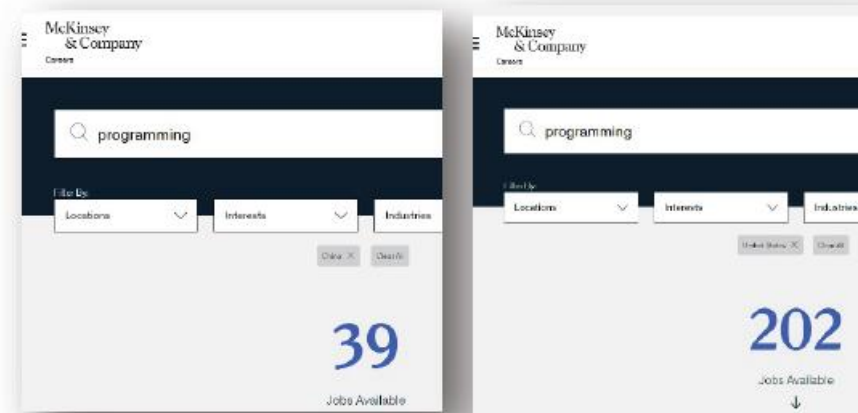
A

北京理工大学 北京师范大学 重庆大学  
华东师范大学 南开大学 深圳大学  
四川大学 武汉大学

A-

北京航空航天大学 华中科技大学 东南大学  
南方科技大学 中山大学 同济大学  
西安交通大学

No. 1 in China, by UTD TOP 100 Business School Research Rankings 2017-2021



2月14日在麦肯锡全球的招聘网站上以“编程”作为关键字搜索，中国区有39个结果，北美有202个结果。中国总共有170个岗位，北美总共有493个岗位。

June 7, 2021 in Issues in Education

## Python: The New MBA Must-Have

By C. Daniel Guetta, Mattan Griffel

SHARE: [f](#) [in](#) [t](#)

PRINT ARTICLE: 

<https://doi.org/10.1287/orms.2021.03.08>

# 商学院学生学习信息技术（编程）的意义

## 基础能力：拓展量化技能

金融、咨询领域的统计和分析能力；

营销策略的论证和实施；

经济学和商科的科研能力；

...

## 中层境界：转变思维方式

递归和循环思想；

数据结构的设计；

面向过程和面向对象的程序设计；

...

## 高层洞察：科创时代的技术发展和企业经营趋势

人工智能给企业带来了哪些机遇和挑战？

“低代码开发？那还学啥编程！”

“GPT-3等超级模型不断涌现，那我改进计算效率有啥用？”

...

# 商学院学生学习信息技术（编程）的意义

- ▶ 完整的数据团队
  - ▶ 数据管理（数据库）
  - ▶ + 数据处理（程序设计/业务逻辑）
  - ▶ + 数据分析（算法/统计）
  - ▶ + 数据产品（程序设计/开发）



# 管理学院的我应该学习哪个语言？

## ► 工程 vs 分析 （以Java和Python为例）

### ► Java的运行速度明显快于python

► 技术鄙视链：C(1)->C++(1.1-1.2)->Java(1.2-1.5)->Python(2-3)->R(3-4)

### ► Java可以在各种设备上运行，如手机、智能卡、汽车

### ► Python解释型语言：输入一行并按下回车键时，就会执行每一行。所以调试起来就不那么痛苦了（导致执行效率低），开发时间也比较短

# 我应该花多少时间学习编程？

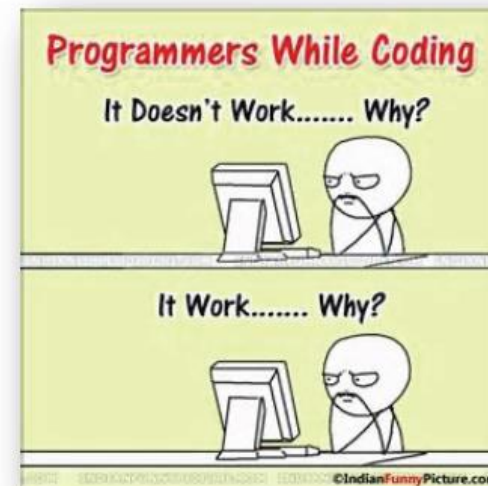
- ▶ 上课之外至少保证每周3-5小时
- ▶ 熟悉教材的代码、做作业
- ▶ 尝试一些拓展练习

# 学编程和学习其他课有什么不一样？

Enjoy your debugging;

“As soon as we started programming, we found to our surprise that it was not as easy to get programs right as we had thought. **Debugging** had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programs.”

- — Maurice Wilkes, 1979



# 可以期待

- ▶ 这是一门很有挑战性的课，也是一门让你可以为之自豪的课
- ▶ 它不保证成为大家找工作最有用的课，但希望成为能影响大家很多年的课
- ▶ 它不保证成为大家学得最好的课，但希望教给你们自学学不好的内容

# 课程计划

- 10次史带楼课堂授课：统一知识点要求
- 4次上机
- 1次期中考试
- 1次期末复习

## 10个知识点

1. 编程语言概述, Python安装和使用 (教材第1章)	安装上手	Lab 1
2. 运算符、表达式和内置函数、Python自带的基础数据结构 (教材第2、3章)		
3. 控制流: 条件、选择、循环 (教材第4章)	简单程序	Lab 2
4. 函数的定义和使用 (教材第5章)		
5. 字符串和简单文本处理 (教材第7、8章)	文件文本	期中考试
6. 文件和文件夹 (教材第9章)		
7. 面向对象入门 (教材第6章)	进阶使用	Lab 3
8. 网络编程入门 (tcp/ip, 收发email、简单爬虫, 可参考使用参考书第18章的部分内容)		
9. 数据分析I (计算工具包, 如numpy、scipy、pandas等)	面向管院	Lab 4
10. 数据分析2 (可视化包, 如matplotlib, seaborn等)		

未列入的重要知识点 (可以在lab讲一点): 数据库 (sql需要单学)、算法与数据结构 (可在现有知识中穿插介绍简单算法思想如排序等)

# 考核方式

- ▶ 期中考试 (10%，笔试或者上机考待定)
- ▶ 上机期末考试 (四教机房)
- ▶ 10次平时作业 (个人)，elearning提交(20%)
- ▶ 4次上机实验 (个人)，四教机房完成任务 (10%)
- ▶ 课堂内的出勤参与(10%)
- ▶ 作业提交时间都是周日晚上10点
- ▶ 上课前迟交扣分起评，上课后迟交该次无成绩。



# 还有一些课程有关的事实

- ▶ 许多老师（包括我在内）和优秀的程序员有很大距离...
  - ▶ 我只是研究者，会编程
- ▶ 得A并不意味着你是优秀的程序员和分析师...
  - ▶ 基础的Python功能满足考试，但是远未达到信管或管院的期待...
- ▶ 那干嘛还要学？
  - ▶ 今天的python，可能是明天的其他语言，学习“怎么学”
  - ▶ 做一些Baby projects让你真的掌握理解一个语言的一些特点
  - ▶ 知道自己...也许其实擅长或者其实并不擅长编程
  - ▶ 了解信息技术和数据分析工作的一部分（比如作为未来学习算法、数据库等的基础）

# 教材和参考书

- 董付国 Python程序设计基础 (第2版)
- 江红 余青松 Python程序设计与算法基础教程 (第2版)



Python程序设计基础 (第2版)



Python程序设计与算法基础教程 (第2版)



# 课堂内计算机的使用规则

- ▶ 课程分为教室和上机
  - ▶ 教室上课务必自带计算机，上机课可以使用自己的电脑
- ▶ 教室通常分为讲解+操作
  - ▶ 讲解期间建议有限使用自己的电脑，认真听讲；
  - ▶ 操作期间请保证电脑电量充足

# 答疑方式

- ▶ 课程会建群，供学生之间交流和班级发通知等，并不是正式答疑渠道，但可以尝试解决简单的、存在共性的小问题；不要在群里追着助教或者私信助教立马要答案。助教们也要上课和科研，看到后会尽快回复；
- ▶ 涉及多行代码的通常无法在微信内答疑，请发email向助教答疑，写清楚问题，你自己尝试的失败或不解经历，附上所有代码，写清自己姓名学号，email主题包含“答疑”。助教回答不了会找我商量
- ▶ 最好的答疑时间是课堂和操作时间，直接问；不能当堂回答的保证事后答复
- ▶ 公平起见，正在进行中的作业题目不答疑
- ▶ 老师没用过mac，对mac上出现的系统问题无法回答，抱歉

# Python是这样一种语言

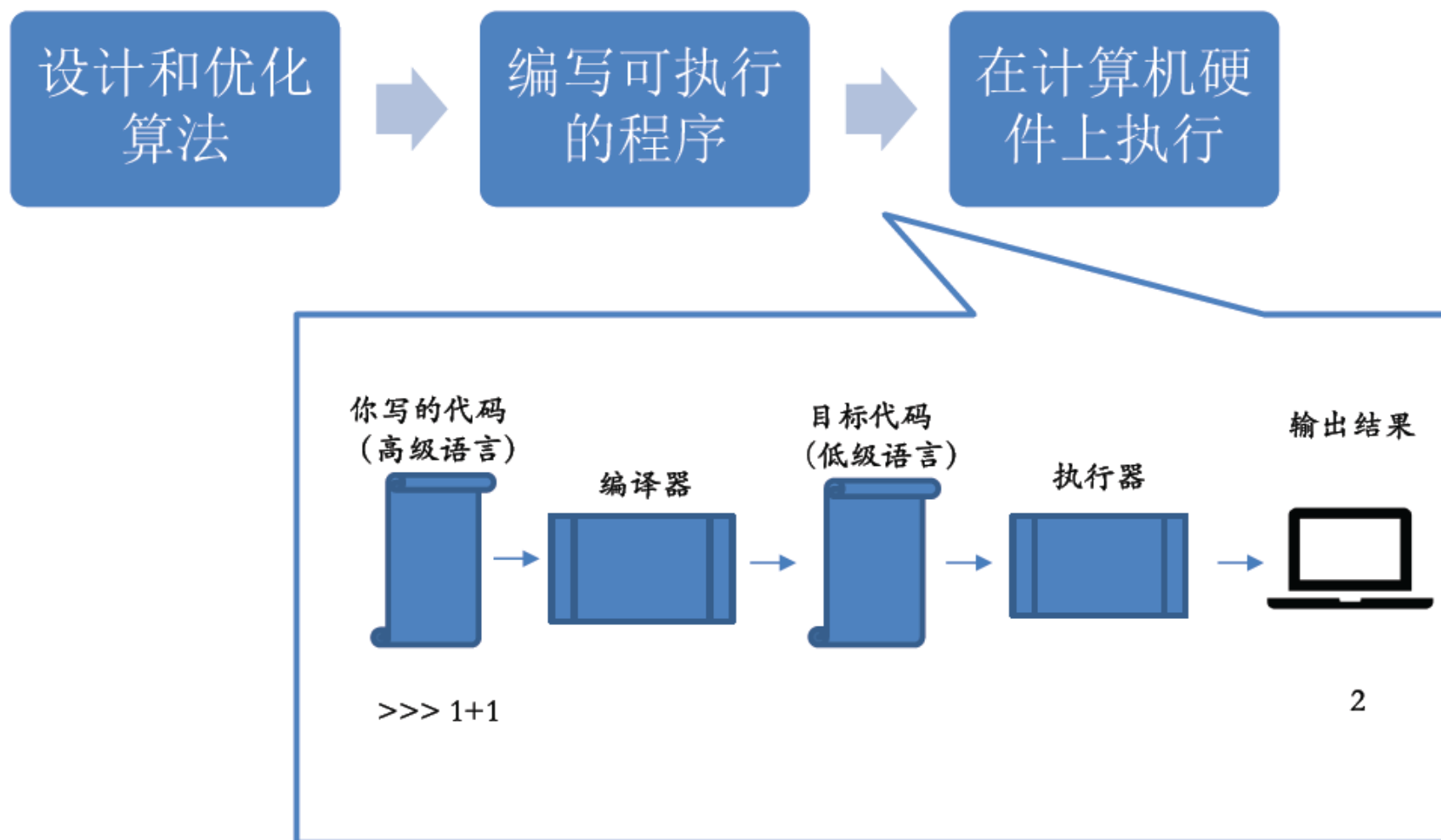
- Python = 球蟒



- Python是一门解释型高级编程语言

## Python版本之争

- ▶ Python目前存在2.x和3.x两个系列的版本，互相之间不兼容，相应的函数库不一样
- ▶ Python 2.x系列于2020年全面放弃维护和更新
- ▶ Python 3
  - ▶ 最新版本 3.10.x



设计和优化  
算法



编写可执行的  
程序



在计算机硬  
件上执行

低级语言



汇编语言

计算机只能运  
行低级语言编  
写的程序

高级语言



Python, Java,  
C...

容易编写、可移  
植、程序短、可  
阅读...

## arm64机器语言与汇编语言

```
BF C3 1E B8 stur wzr, [x29, #-0x14]
BF 83 1E B8 stur wzr, [x29, #-0x18]
A8 83 5E B8 ldur w8, [x29, #-0x18]
1F 91 01 71 cmp w8, #0x64
2A 01 00 54 b.ge 0x102b926a0
A8 83 5E B8 ldur w8, [x29, #-0x18]
A9 C3 5E B8 ldur w9, [x29, #-0x14]
28 01 08 0B add w8, w9, w8
A8 C3 1E B8 stur w8, [x29, #-0x14]
A8 83 5E B8 ldur w8, [x29, #-0x18]
08 05 00 11 add w8, w8, #0x1
A8 83 1E B8 stur w8, [x29, #-0x18]
F6 FF FF 17 b 0x102b92674
A8 C3 5E B8 ldur w8, [x29, #-0x14]
E9 03 08 AA mov x9, x8
EA 03 00 91 mov x10, sp
49 01 00 F9 str x9, [x10]
00 00 00 B0 adrp x0, 1
00 F8 3D 91 add x0, x0, #0xf7e
D5 00 00 94 bl 0x102b92a0c
```

## IA-64机器语言与汇编语言

```
C7 45 EC 00 00 00 00 movl $0x0, -0x14(%rbp)
C7 45 E8 00 00 00 00 movl $0x0, -0x18(%rbp)
83 7D E8 64 cmpl $0x64, -0x18(%rbp)
0F 8D 17 00 00 00 jge 0x101a48b02
8B 45 E8 movl -0x18(%rbp), %eax
03 45 EC addl -0x14(%rbp), %eax
89 45 EC movl %eax, -0x14(%rbp)
8B 45 E8 movl -0x18(%rbp), %eax
83 C0 01 addl $0x1, %eax
89 45 E8 movl %eax, -0x18(%rbp)
E9 DF FF FF FF jmp 0x101a48ae1
48 8D 3D 27 22 00 00 leaq 0x2227(%rip), %rdi
8B 75 EC movl -0x14(%rbp), %esi
B0 00 movb $0x0, %al
E8 D1 0C 00 00 callq 0x101a497e4
```

## C语言

```
int sum = 0;
for (int i = 0; i < 100; i++)
{
    sum += i;
}
printf("sum:%d", sum);
```

**PyScript** C:\ProjectCode\PyScript
 

- Domain2ip2locality
- DomainSearch
- JBOSS-JMX-EJB-InvokerServ
- VT\_domain\_ip
- backdoor
  - connect.py
  - reverseTcp.py
- chinaz-extractIPandlocality
- enginesearch

```

shellcode += "\x75\x05\xbb\x4

md = Cs(CS_ARCH_X86, CS_MODE_32)
for i in md.disasm(shellcode, 0):
    print "0x%x:\t%s\t%s" % (i.address, i.mnemonic, i.opstr)

```

Debug:

Debugger

```

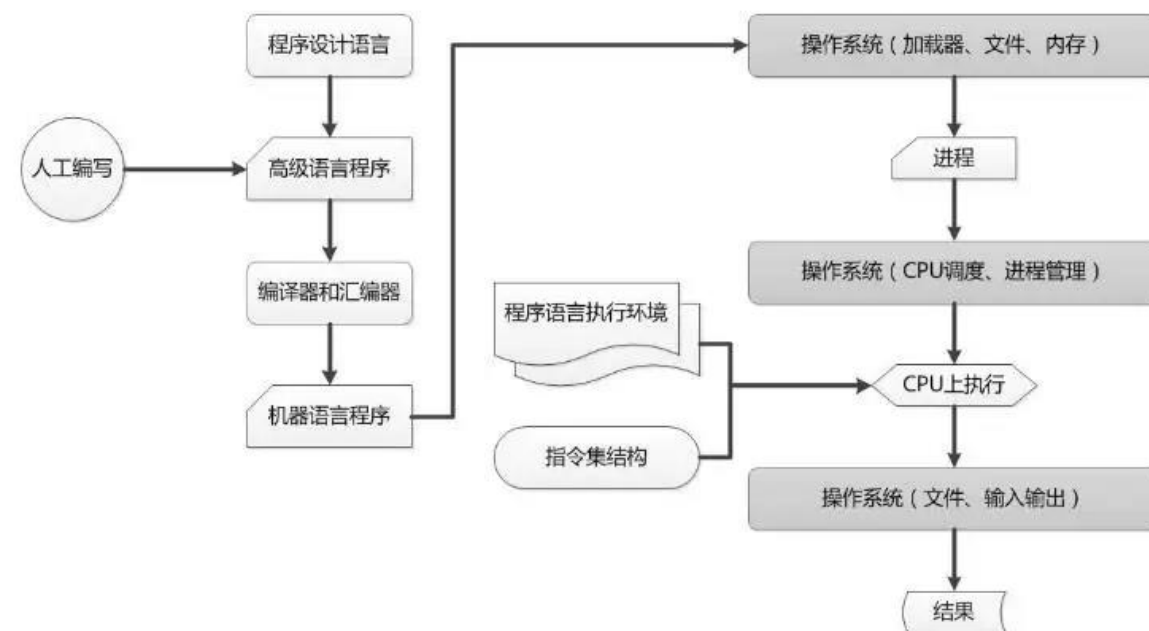
0x2c:  push    edx
0x2d:  push    edi
0x2e:  mov     edx, dword ptr [edx + 0x10]
0x31:  mov     ecx, dword ptr [edx + 0x3c]
0x34:  mov     ecx, dword ptr [ecx + edx + 0x78]
0x38:  jecxz   0x82
0x3a:  add     ecx, edx
0x3c:  push    ecx
0x3d:  mov     ebx, dword ptr [ecx + 0x20]
0x40:  add     ebx, edx
0x42:  mov     ecx, dword ptr [ecx + 0x18]
0x45:  jecxz   0x81
0x47:  dec     ecx
0x48:  mov     esi, dword ptr [ebx + ecx*4]
0x4b:  add     esi, edx
0x4d:  xor     edi, edi
0x4f:  lodsb   al, byte ptr [esi]

```



# 程序设计与运行

- 程序设计语言进行编程
- 编译器和汇编器将高级语言编译成机器语言  
便于机器识别
- 机器语言程序加载到内存才能形成一个运动  
中的程序（即进程）
- 进程在CPU（计算机芯片）上执行，由操作  
系统完成





# Python开发环境的安装与使用

- 默认编程环境： **IDLE**
- 其他常用开发环境：
  - Anaconda3 (内含Jupyter和Spyder) : <https://www.anaconda.com/download>
  - Visual Studio Code
  - PyCharm
  - PyScripter
  - Eclipse+PyDev
  - wingIDE
  - PythonWin
  - .....

B站视频:

[https://www.bilibili.com/video/BV1YS4y1z7eu?p=7&vd\\_source=beb8714af7cf5231115f279477c872bf](https://www.bilibili.com/video/BV1YS4y1z7eu?p=7&vd_source=beb8714af7cf5231115f279477c872bf)

# Python 安装及环境搭建

## Python 安装

### 下载python

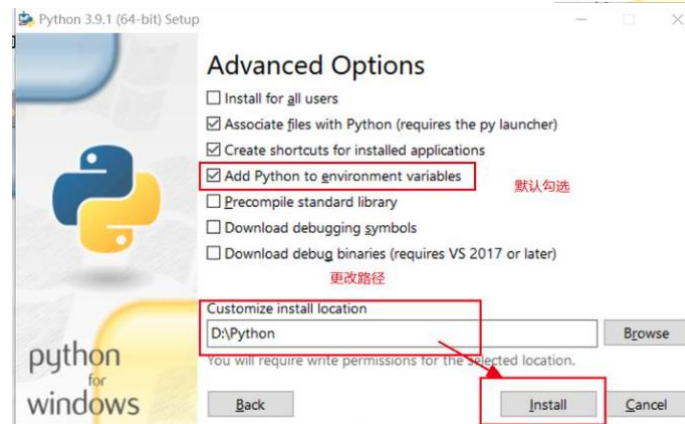
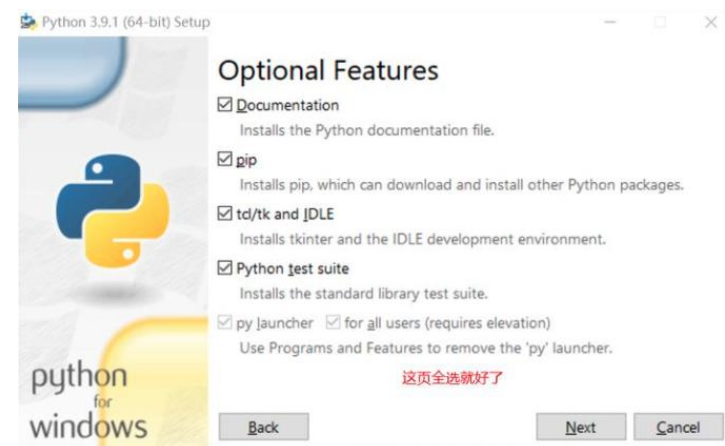
- 进入Python官网，点击Download

<https://www.python.org/>

- 选择版本 (3.7-3.10) 及平台  
(Windows/mac) 32位或64位

### Windows安装，选择目录

- 勾选pip/IDLE, add python to environment variables
- 将目录及scripts 目录加入path (后续)



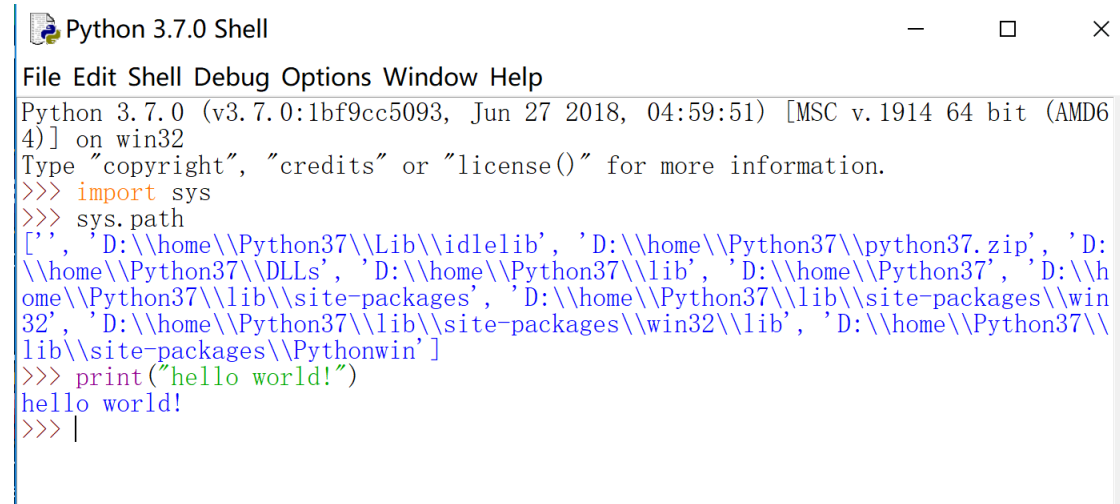
# 安装参考视频

- ▶ [https://www.bilibili.com/video/BV1YS4y1z7eu?is\\_story\\_h5=false&p=1&share\\_from=ugc&share\\_medium=iphone&share\\_plat=ios&share\\_session\\_id=772C0D71-4574-4A4D-8B48-5C457B128A7C&share\\_source=WEIXIN&share\\_tag=s\\_i&timestamp=1661405579&unique\\_k=mMaxBGB](https://www.bilibili.com/video/BV1YS4y1z7eu?is_story_h5=false&p=1&share_from=ugc&share_medium=iphone&share_plat=ios&share_session_id=772C0D71-4574-4A4D-8B48-5C457B128A7C&share_source=WEIXIN&share_tag=s_i&timestamp=1661405579&unique_k=mMaxBGB)

# 开发环境 (IDLE)

## 1. IDLE

- 在windows菜单中选择IDLE，进入互动界面
- 称之为开发环境



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import sys
>>> sys.path
['', 'D:\\home\\Python37\\Lib\\idlelib', 'D:\\home\\Python37\\python37.zip', 'D:\\home\\Python37\\DLLs', 'D:\\home\\Python37\\lib', 'D:\\home\\Python37', 'D:\\home\\Python37\\lib\\site-packages', 'D:\\home\\Python37\\lib\\site-packages\\win32', 'D:\\home\\Python37\\lib\\site-packages\\win32\\lib', 'D:\\home\\Python37\\lib\\site-packages\\Pythonwin']
>>> print("hello world!")
hello world!
>>> |
```

# 运行环境

- Python 运行模式
- 1 命令行
  - Python 程序.py
- 或者：
  - Python
  - >>>quit() /exit()

```
管理员: 命令提示符

C:\>python
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> sys.path
['', 'D:\\home\\Python37\\python37.zip', 'D:\\home\\Python37\\DLLs', 'D:\\home\\Python37\\lib', 'D:\\home\\Python37', 'D:\\home\\Python37\\lib\\site-packages', 'D:\\home\\Python37\\lib\\site-packages\\win32', 'D:\\home\\Python37\\lib\\site-packages\\win32\\lib', 'D:\\home\\Python37\\lib\\site-packages\\Pythonwin']
>>> print("hello world!")
hello world!
>>> exit()

C:\>_
```

# 让我们实验以下步骤

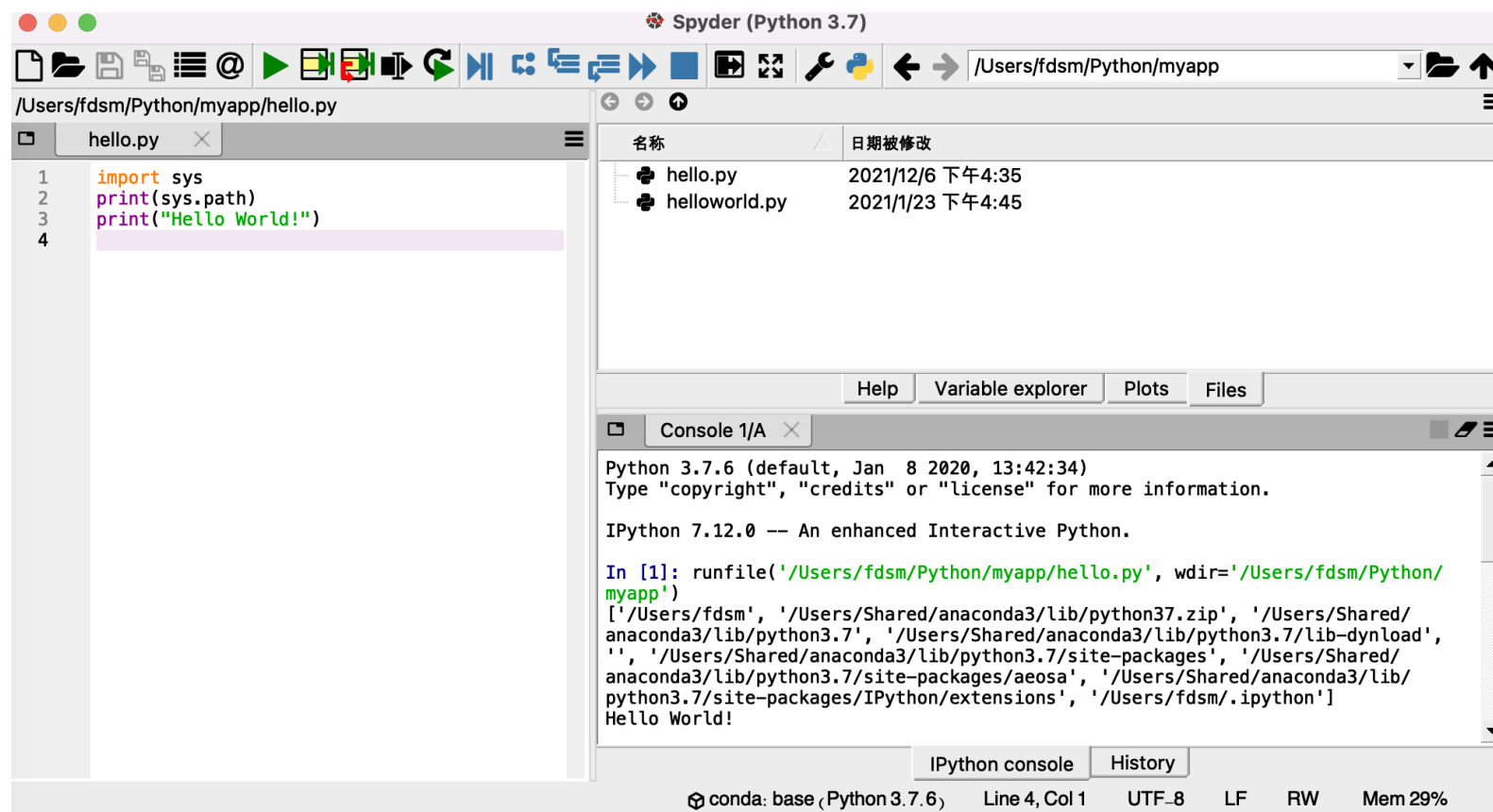
- ▶ “另存为” helloworld.py
  - ▶ 注意观察打开的目录窗口：原来程序被idle放到了这
- ▶ 用“打开”看到刚才存储的文件，将其拷贝到你认为方便的硬盘位置，比如C盘或D盘根目录下
  - ▶ Ctrl+c (选中文件拷贝) ， Ctrl+v (粘贴到打开的位置)
  - ▶ 原来程序和word文档没差别，都可以进行文件操作
- ▶ 用记事本打开该文件，修改内容，只保留指令
  - ▶ 原来程序可以像文本一样修改

- ▶ Windows打开“运行”、输入cmd进入命令行界面，然后直接调用python
  - ▶ Python helloworld.py （报错，找不到文件）
  - ▶ Python d:\helloworld.py
    - ▶ 程序真正是这样被操作运行的
    - ▶ print才输出，之前sys.path内容不见了！
  - ▶ 执行
    - ▶ cd\
    - ▶ d:
    - ▶ Python helloworld.py
    - ▶ 原来同位置就可以不用输入路径了，再次确认之前为什么出错

# 更多的(集成)开发环境举例

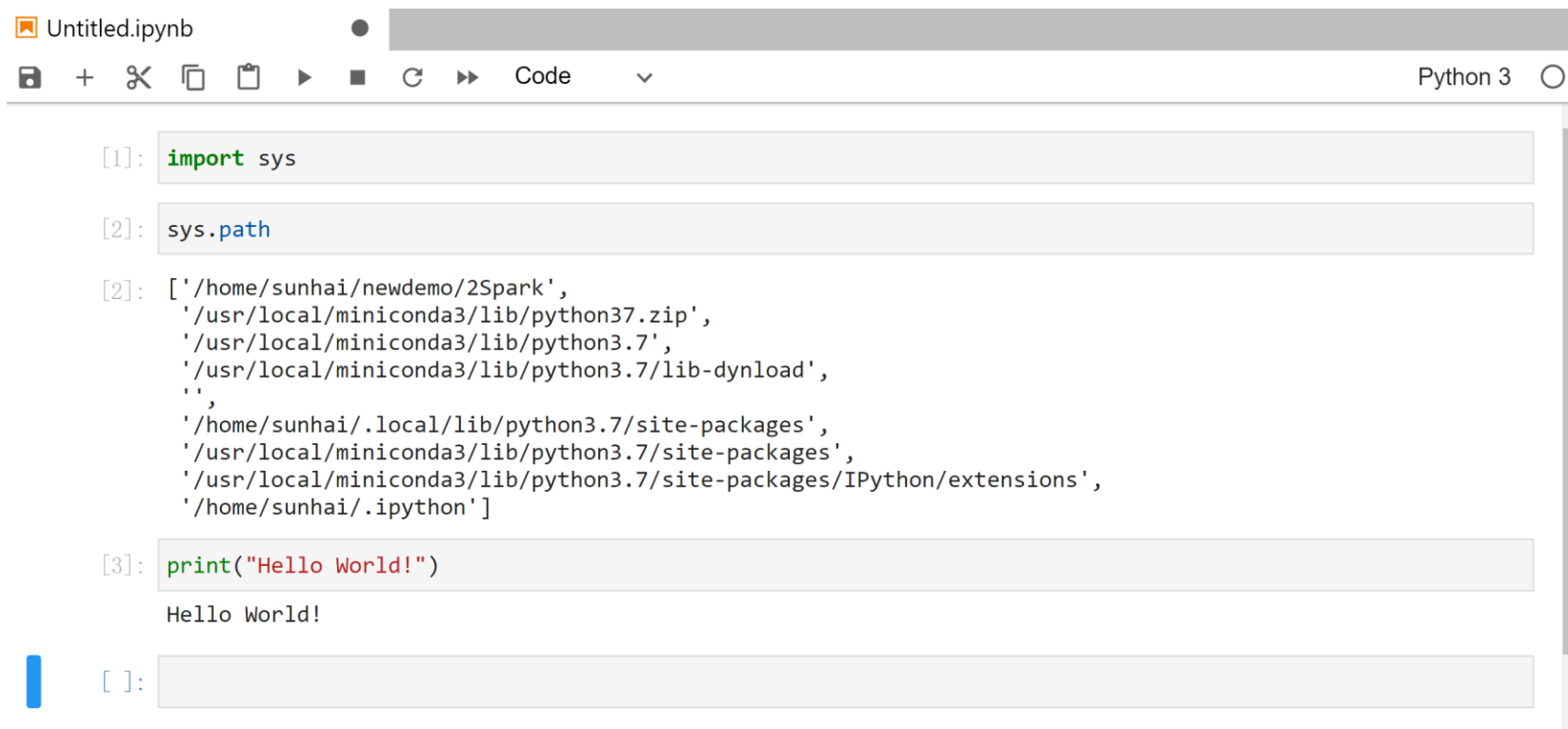
比如下载和安装Anaconda3, 内含  
Spyder 和Jupyter两种开发环境

小知识: 集成开发环境 (IDE, Integrated Development Environment) 是用于提供程序开发环境的应用程序, 集成了代码编写、编译、调试等一体化的开发环境。所有具备这一特性的软件都可以叫集成开发环境。如微软的 Visual Studio系列等。





# Jupyter Notebook



```
Untitled.ipynb Python 3

[1]: import sys

[2]: sys.path

[2]: ['/home/sunhai/newdemo/2Spark',
      '/usr/local/miniconda3/lib/python37.zip',
      '/usr/local/miniconda3/lib/python3.7',
      '/usr/local/miniconda3/lib/python3.7/lib-dynload',
      '',
      '/home/sunhai/.local/lib/python3.7/site-packages',
      '/usr/local/miniconda3/lib/python3.7/site-packages',
      '/usr/local/miniconda3/lib/python3.7/site-packages/IPython/extensions',
      '/home/sunhai/.ipython']

[3]: print("Hello World!")
Hello World!

[ ]:
```

# 四类常见IDE

- ▶ 本地的Python通用类环境
  - ▶ 基础Python的IDLE: 复旦大学上机和测试环境
  - ▶ 全功能的PyCharm: 有高校免费的完整版
- ▶ 面向数据分析的Python环境Anaconda
  - ▶ Jupyter Notebook
  - ▶ 类似Rstudio和Matlab的Spyder
- ▶ 浏览器内直接使用的在线版本
  - ▶ Datalore、Kaggle、Google Colab、Github还有很多
- ▶ 基于本地Python编译器+命令行的通用代码编辑器
  - ▶ VS Code、Sublime或者...记事本
- ▶ 建议学会**IDLE**、后续熟练**PyCharm**或**Jupyter**的一个, 以后有需要的话再说其他的

# 开启仪式

- >>> print("hello world")
  
- ▶ 1. 安装配置好Python环境, 在IDLE中打印 “Hello! World” (作业提交一个运行画面截图) ;
- ▶ 2. 在IDLE中将打印 “Hello! World”的语句存为hello.py文件, 随后尝试用IDLE将其打开并运行, 再次在终端得到 “Hello! World” (作业提交一个运行画面截图) ;
- ▶ 3. 退出你的编译环境, 通过记事本等文本编辑器单独新建一个文件, 存为"hello1.py"文件,随后尝试用IDLE将其打开并运行, 再次在终端得到 “Hello! World” (作业提交一个运行画面截图) ;思考: 两个py文件有什么不同?
- ▶ 4. 打开命令行工具(windows里为运行->cmd), 运行python 路径名\hello.py (作业提交一个运行画面截图)
- ▶ 5. 在你的IDLE中动手操作并复现教材第一章1.6节的三段代码。(注意: import的库需要你提前安装。(作业提交三个运行画面截图)
  
- ▶ 小提示1: 在IDLE当中如果你输入一半, 例如 “print” 的 “pri” 时, 可以按tab键, IDLE会给你一些代码的提示, 或者帮助你补全代码。
- ▶ 小提示2: 在IDLE当中, 可以用组合键Alt+P来自动输入前一句命令。这样不用在代码重复时不停地拷贝粘贴了。
  
- ▶ **纸上得来终觉浅, 绝知此事要躬行。第一周, 一定要自己动手哦!**

# Python编程规范

为了方便理解和交流而做的规范，类似语文要求，不是必须，但是约定俗成；  
等后续学到逻辑结构时再尝试，目前的语句都需要顶格书写

## (1) 缩进

- ✓ python程序是依靠代码块的缩进来体现代码之间的逻辑关系的，**缩进结束就表示一个代码块结束了。**
- ✓ 同一个级别的代码块的缩进量必须相同。
- ✓ 一般而言，以**4个空格**为基本缩进单位。

```
with open(fn) as fp:
    for line in csv.reader(fp):
        if line:
            print(*line)
```

# Python编程规范

(2) 每个import语句只导入一个模块，写在代码最前面

```
import csv  
import random  
import datetime  
import pandas as pd  
import matplotlib.pyplot as plt
```

# Python编程规范

(3) 最好在每个类、函数定义和一段完整的功能代码之后增加一个空行，在运算符两侧各增加一个空格，逗号后面增加一个空格。

```
sockDianming.close()
sockDianming = None

# 开始点名
def buttonStartDianmingClick():
    # 如果还没有注册，拒绝运行
    if int_zhuce.get() == 0:
        tkinter.messagebox.showerror('很抱歉', '请联系作者进行软件注册!')
        return
    if int_zuoye.get() == 1:
        tkinter.messagebox.showerror('很抱歉', '现在正在收作业')
        return
    if int_canDianming.get() == 1:
        tkinter.messagebox.showerror('很抱歉', '现在正在点名')
        return
    tkinter.messagebox.showinfo('恭喜', '设置成功，现在开始点名')
    #开始点名
    int_canDianming.set(1)
    global tDianming_id
    t = threading.Thread(target=thread_Dianming)
    t.start()
    tDianming_id = t.ident
    buttonStartDianming = tkinter.Button(root, text='开始点名', command=buttonStartD
    buttonStartDianming.place(x=20, y=60, height=30, width=100)

def buttonStopDianmingClick():
    # 如果还没有注册，拒绝运行
    if int_zhuce.get() == 0:
        tkinter.messagebox.showerror('很抱歉', '请联系作者进行软件注册!')
```

Diagram illustrating Python coding style annotations:

- Red arrow pointing to the empty line after `sockDianming = None`: 空行
- Red arrow pointing to the space around the assignment operator `=` in `sockDianming = None`: 有空格
- Green arrow pointing to the space after the comma in `buttonStartDianming = tkinter.Button(...)`: 有空格
- Green arrow pointing to the space after the comma in `buttonStartDianming.place(...)`: 有空格
- Green arrow pointing to the space after the comma in `buttonStartDianming = tkinter.Button(...)`: 没有空格
- Red arrow pointing to the empty line after `buttonStartDianming.place(...)`: 空行

# Python编程规范

(4) 尽量不要写过长的语句。如果语句过长，可以考虑拆分成多个短一些的语句，以保证代码具有较好的可读性。如果语句确实太长而超过屏幕宽度，最好使用续行符 (line continuation character) “\”，或者使用圆括号将多行代码括起来表示是一条语句。

```
x = 1 + 2 + 3\  
    + 4 + 5\  
    + 6  
  
y = (1 + 2 + 3  
     + 4 + 5  
     + 6)
```

# Python编程规范

## (5) 注释

- ✓ 以符号#开始，表示本行#之后的内容为注释。

```
# 读取并显示上面代码生成的csv文件内容
with open(fn) as fp:
    for line in csv.reader(fp):
        if line:
            print(*line)
```



# 安装扩展库的几种方法

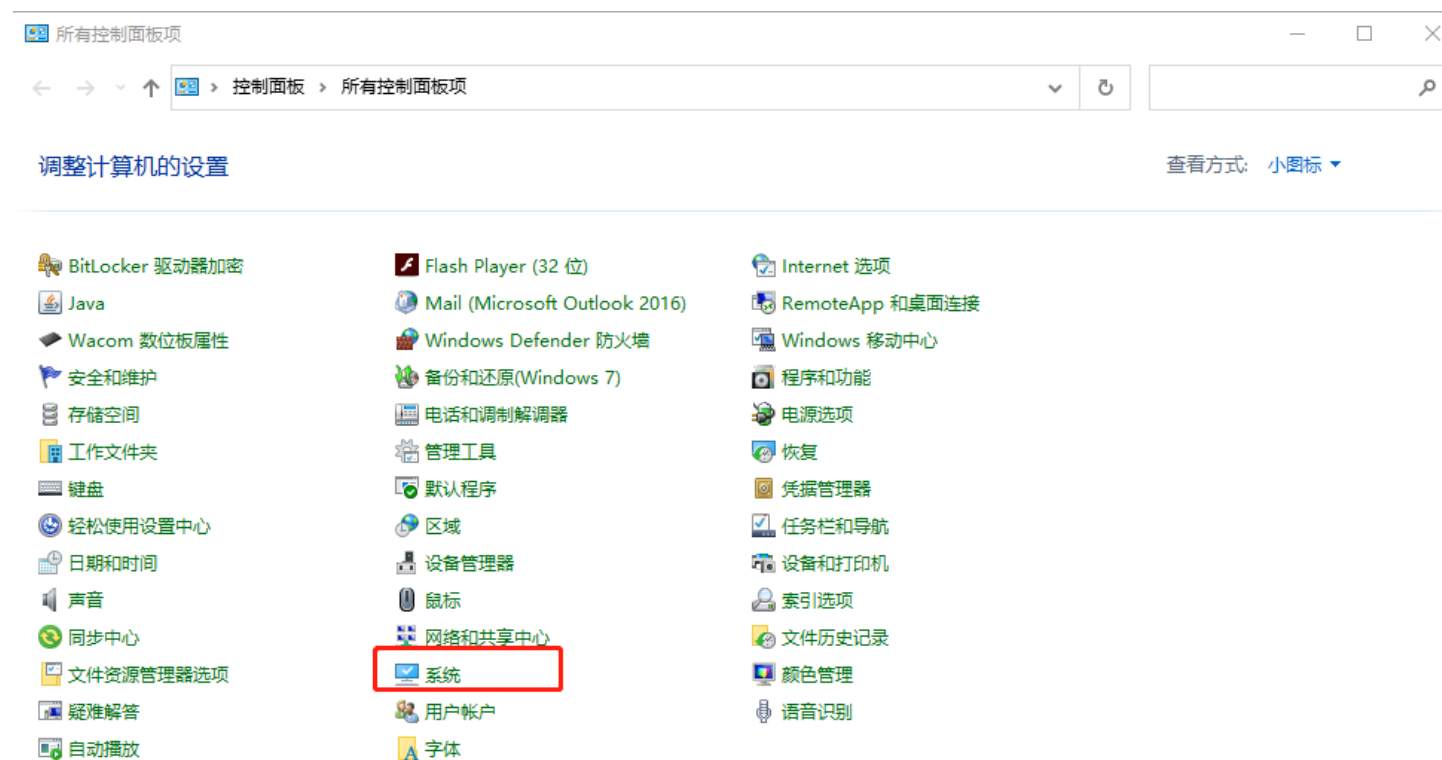
前提：安装了pip这个程序

- ▶ pip在线安装（命令提示符环境）
- ▶ pip离线安装：<https://www.lfd.uci.edu/~gohlke/pythonlibs/>
- ▶ exe安装，不是每个扩展库都支持
- ▶ 如果机器上安装了多个Python开发环境，那么在一个环境下安装的扩展库无法在另一个环境下使用，需要分别安装。
- ▶ conda在线安装(Anaconda)，支持多环境设置
  - ▶ conda env list 查看当前存在哪些虚拟环境

# 安装pip

- ▶ 首先在python文件下的Scripts看是否有pip这个文件
  - ▶ 比如我的在: C:\Users\dell\AppData\Local\Programs\Python\Python310\Scripts\
  - ▶ 如果有, 到该目录下 命令行执行pip list, 无报错, 则已安装成功
- ▶ 如果没有, 自行下载<https://pypi.org/project/pip/#files>
  - ▶ 注意: 是压缩文件, 系统里需要先安装winrar这类解压缩程序
  - ▶ 进入解压缩后的文件夹, 在命令行下执行
    - ▶ python setup.py install
- ▶ 小诀窍
  - ▶ 命令行可以用鼠标选定后ctrl+c / v
  - ▶ 路径信息在文件资源管理器上方直接输入, 就可以转到相应目录

- 如果是路径报错，去控制面板->系统->高级系统设置->环境设置，在path中加入（两个均可，通常加系统的path更通用）
  - C:\Users\dell\AppData\Local\Programs\Python\Python310\Scripts\



相关设置

[BitLocker 设置](#)

[设备管理器](#)

[远程桌面](#)

[系统保护](#)

[高级系统设置](#)

[重命名这台电脑](#)

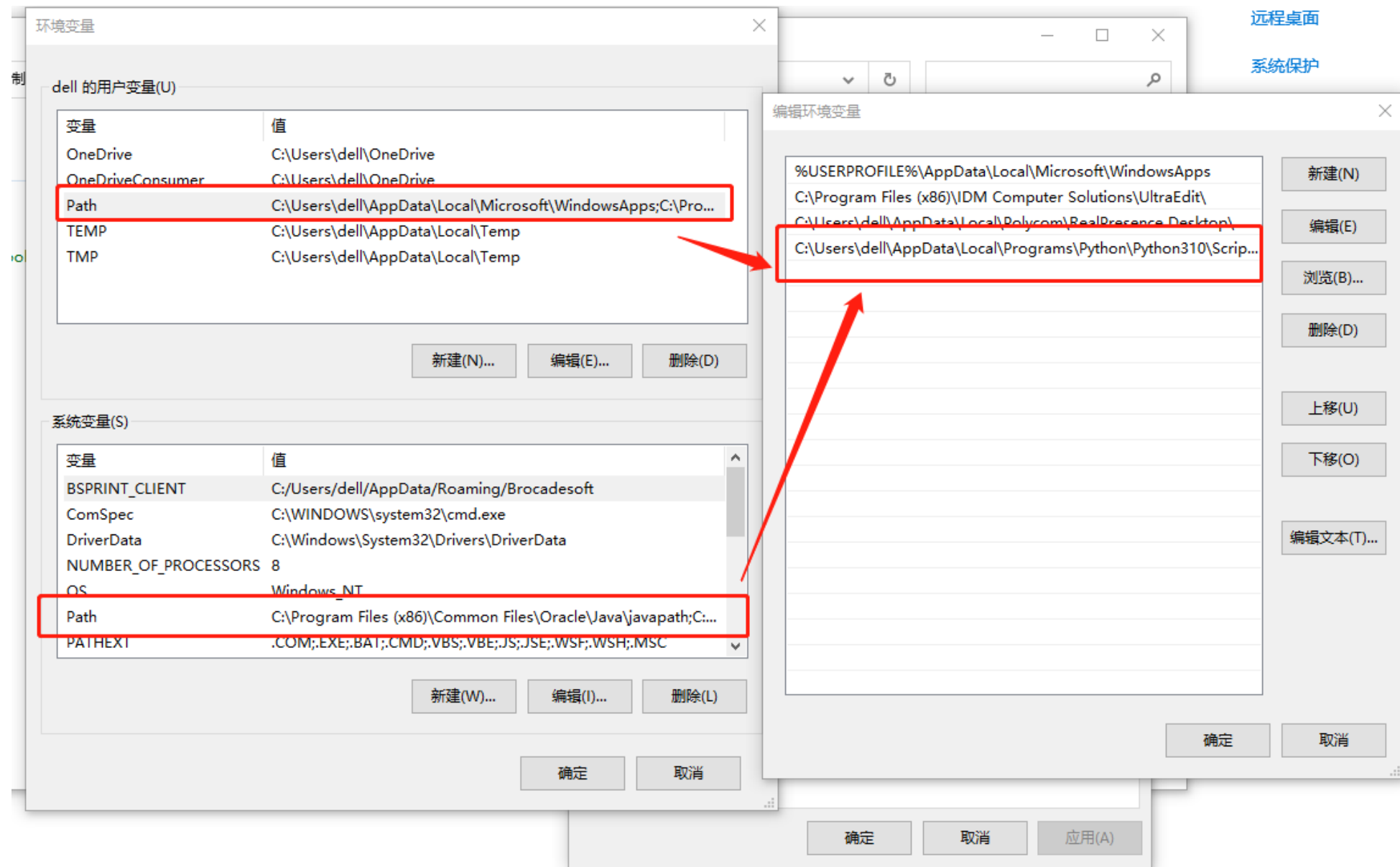
 [获取帮助](#)

 [提供反馈](#)



远程桌面

系统保护



# 安装扩展库的几种方法

pip命令示例	说明
<code>pip download SomePackage[==version]</code>	下载扩展库的指定版本，不安装
<code>pip freeze [&gt; requirements.txt]</code>	以requirements的格式列出已安装模块
<code>pip list</code>	列出当前已安装的所有模块
<code>pip install SomePackage[==version]</code>	在线安装SomePackage模块的指定版本
<code>pip install SomePackage.whl</code>	通过whl文件离线安装扩展库
<code>pip install package1 package2 ...</code>	依次（在线）安装package1、package2等扩展模块
<code>pip install -r requirements.txt</code>	安装requirements.txt文件中指定的扩展库
<code>pip install --upgrade SomePackage</code>	升级SomePackage模块
<code>pip uninstall SomePackage[==version]</code>	卸载SomePackage模块的指定版本

把SomePackage替换  
为实际要安装或卸载  
的扩展库名

<http://www.lfd.uci.edu/~gohlke/pythonlibs/>  
下载时选择合适版本，并且不要修改文件名

也可以通过 <https://pypi.org/> 查询

`pip install numpy`

# 标准库与扩展库对象的导入与使用

- Python默认安装仅包含基本或核心模块，启动时也仅加载了基本模块，在需要时再显式地导入和加载标准库和第三方扩展库（需正确安装），这样可以减小程序运行的压力，并且具有很强的可扩展性。
- 从“木桶原理”的角度来看，这样的设计与安全配置时遵循的“最小权限”原则的思想是一致的，也有助于提高系统安全性。

# import 模块名 [as 别名]

```
>>> import math                                #导入标准库math
>>> math.sin(0.5)                              #求0.5（单位是弧度）的正弦
0.479425538604203

>>> import random                             #导入标准库random
>>> n = random.random()                       #获得[0,1) 内的随机小数
>>> n = random.randint(1,100)                 #获得[1,100]区间上的随机整数
>>> n = random.randrange(1, 100)              #返回[1, 100)区间中的随机整数
>>> import os.path as path                     #导入标准库os.path, 并设置别名为path
>>> path.isfile(r'C:\windows\notepad.exe')
True

>>> import numpy as np                         #导入扩展库numpy, 并设置别名为np
>>> a = np.array((1,2,3,4))                   #通过模块的别名来访问其中的对象
>>> a
array([1, 2, 3, 4])
>>> print(a)
[1 2 3 4]
```

在这里，试试刚才的两个小技巧

小提示1：在IDLE当中如果你输入一半，例如“print”的“pri”时，可以按tab键，IDLE会给你一些代码的提示，或者帮助你补全代码。

小提示2：在IDLE当中，可以用组合键Alt+P来自动输入前一句命令。这样不用在代码重复时不停地拷贝粘贴了。



# from 模块名 import 对象名[ as 别名]

```
>>> from math import sin          #只导入模块中的指定对象，访问速度略快
>>> sin(3)
0.1411200080598672
>>> from math import sin as f    #给导入的对象起个别名
>>> f(3)
0.1411200080598672
>>> from os.path import isfile
>>> isfile(r'C:\windows\notepad.exe')
True
```

# from 模块名 import \*

>>> from math import *	#导入标准库math中所有对象
>>> sin(3)	#求正弦值
0.1411200080598672	
>>> gcd(36, 18)	#最大公约数
18	
>>> pi	#常数 $\pi$
3.141592653589793	
>>> e	#常数e
2.718281828459045	
>>> log2(8)	#计算以2为底的对数值
3.0	
>>> log10(100)	#计算以10为底的对数值
2.0	
>>> radians(180)	#把角度转换为弧度
3.141592653589793	

# from... import 与 import区别

- import X 导入一个X模块，相当于导入的是一个文件夹
- from X import Y: 导入了X模块中的一个函数Y; 注：相当于导入的是一个文件夹中的文件
  - from X import \*** 相当于 **import X**
- 区别
  - import math
    - math.exp(3)
  - from math import \*
    - exp(3)
    - 缺点：容易与其它同名函数重名，模块名相当于namespace