# Python01

Python 与 JavaScript 相同, 都是一款编程语言. 但是运行的环境不同:

* js: 运行在浏览器上.
* python: 与C,Java语言相同, 需要通过**编译**操作之后, 运行在 电脑上.

下载编译环境软件: https://dl.softmgr.qq.com/original/Development/python-3.8.5-amd64.exe

## 编译环境

官方下载地址: https://www.python.org/downloads/release/python-392/

python的编译环境大体上分两个版本:

* 2.7: 早期的python版本, 其中很多语法在 3.\* 版本中被移除. 导致新旧版本是两个体系;
  + 有很多项目是使用2.7版本制作的, 无法迁移到3.\*
* 3.\*: 目前的主流版本
  + >=3.9: 目前最新的, 已经不支持 **win7** 系统. 只能安装到win10
  + <3.9: 可以兼容 **win7** 和 **win10**

### 安装



image-20210322092319096

测试, 重新打开cmd命令行工具:

* windows: python -V 或 py -V
* mac: python03 -V

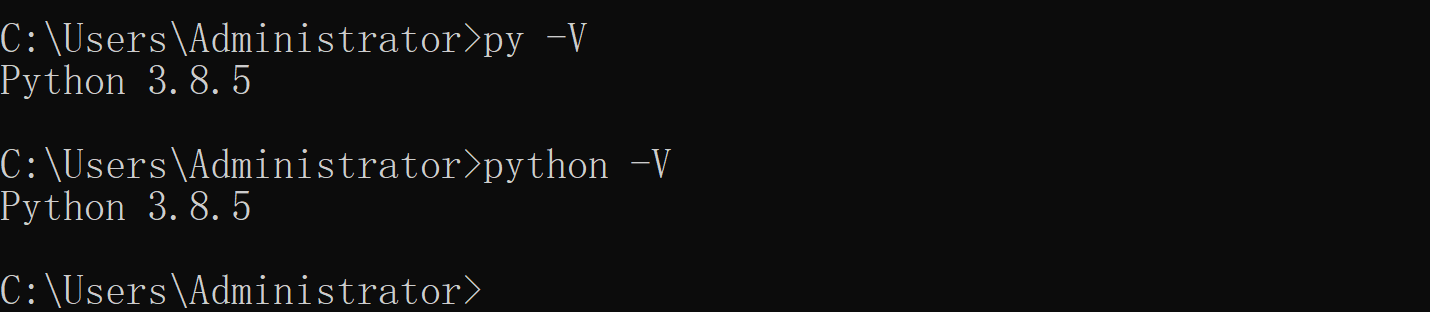


image-20210322092931807

### 相关插件

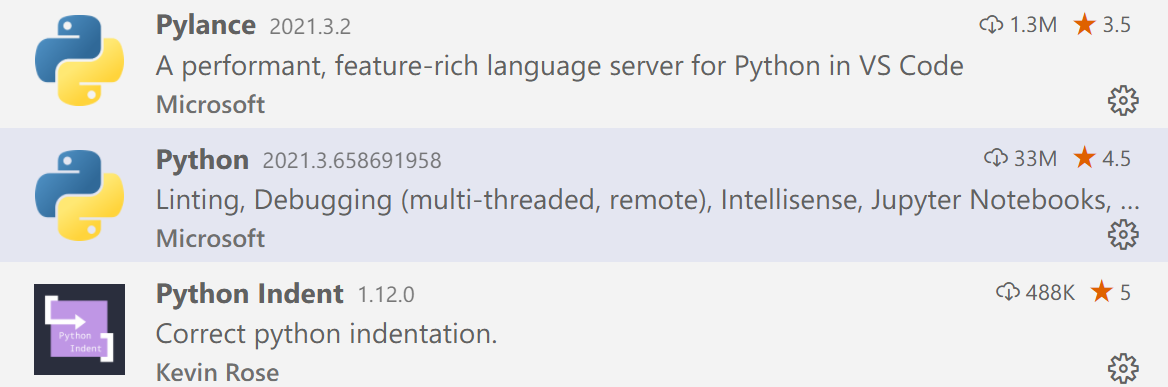


image-20210322093515445

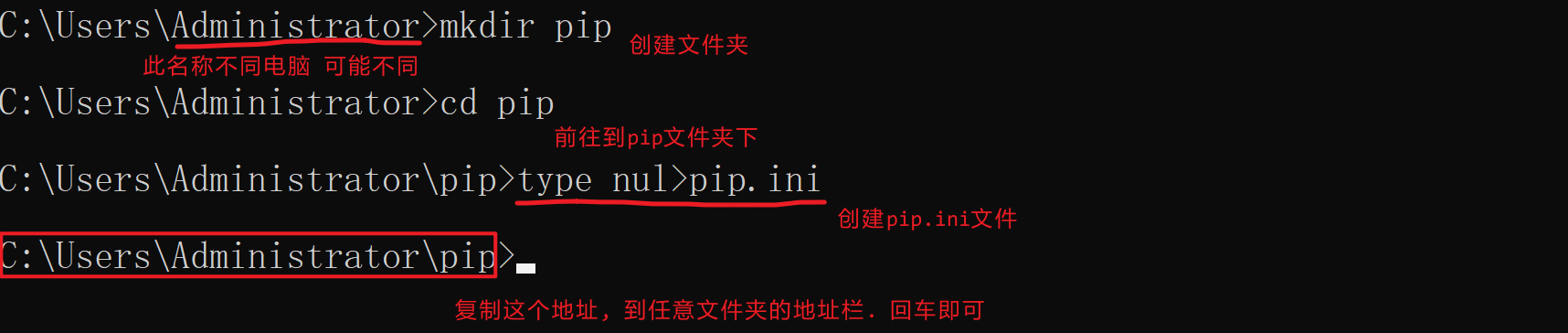
### 中国镜像

与 node 的 npm相同, python 有 pip 工具, 用于下载第三方模块: 这些工具默认都要从国外的源下载内容.

教程: https://www.runoob.com/w3cnote/pip-cn-mirror.html

打开命令行工具 cmd: mkdir pip 此命令是创建文件夹pip

格式说明: C:\Users\\*\*\* 其中\*\*\* 不同电脑不一样! 不一样是正常的.

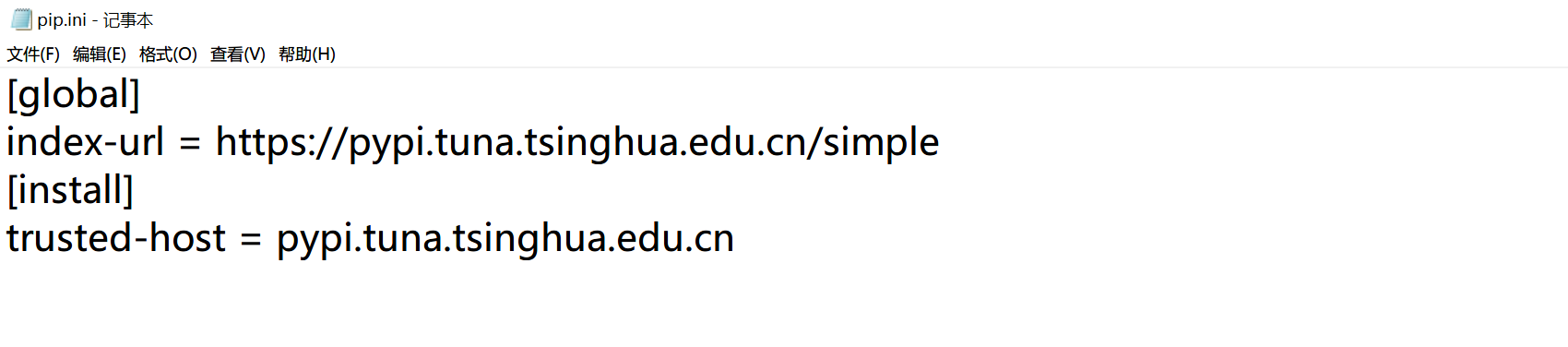


pip的中国镜像地址比较多: **需要测试** 把下列网址放到浏览器访问, 能访问的就可以用.

* 中国科学技术大学 : https://pypi.mirrors.ustc.edu.cn/simple
* 豆瓣：http://pypi.douban.com/simple/
* 清华大学: https://pypi.tuna.tsinghua.edu.cn/simple

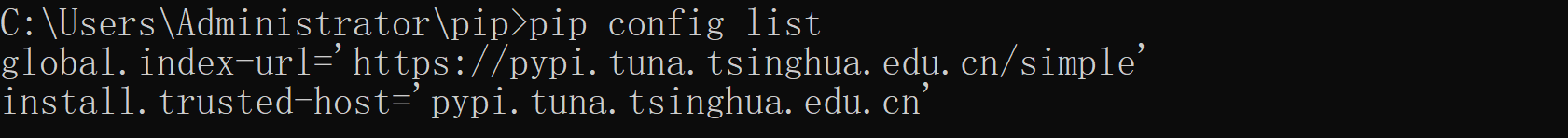
修改 **pip.ini** 文件的内容并保存: 下方为**清华**的**示例**:

[global]  
index-url = https://pypi.tuna.tsinghua.edu.cn/simple  
[install]  
trusted-host = pypi.tuna.tsinghua.edu.cn



测试命令: pip config list

mac: pip3 config list



## Python

为什么要学习python?

* python是目前编程语言排行榜 **TIOBE** 前三
* python可以做很多事情:
  + 爬虫: 从网络上爬取各种资源.
  + 人工智能
  + **网站服务器**: 与node.js相似, 但是 python可以完成 **前后端分离** 和 **前后端不分离** 两种网站的制作
  + 少儿教育

**最终目标**: 体验 **前后端分离** 和 **前后端不分离** 两种不同网站的制作.

## 首次

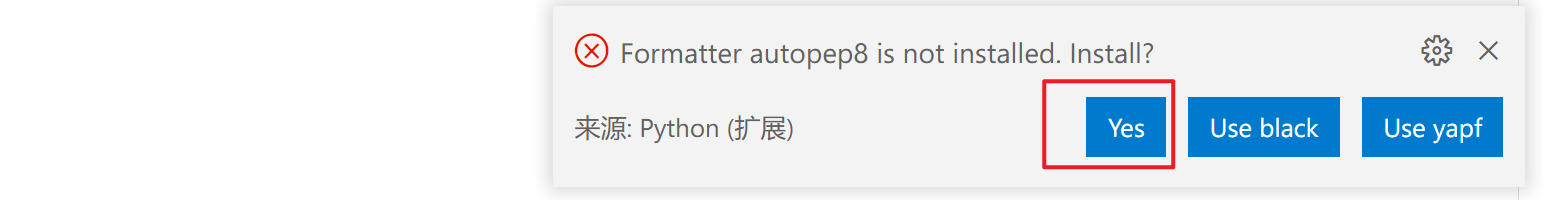
python的制作方式 和 node.js 完全一样:

文件后缀名: .py

首次创建文件, 保存之后 vscode 右下角应该会弹出一些提示, 要求安装支持的扩展:

选 ok yes 等; 不安装不影响代码书写

这就是为什么设置pip中国镜像的原因: 就是这里要安装额外的模块



# python中注释是 #  
# 快捷键 ctrl+/  
  
# python作者的初衷: 创建一门简单的编程语言  
# python语法借鉴了 java 和 C  
# JavaScript语法借鉴了 java 和 C  
# python 和 JavaScript 语法非常相似  
  
# 打印:  
# js: console.log('Hello World!')  
# java: System.out.println("Hello World!")  
  
# python: 写法同C语言  
print("Hello World!")  
  
'''  
多行注释: 同js的 /\* \*/  
  
python的文件.py 的执行 和 node.js 相同:  
\* node xxx.js  
\* py xxx.py   
 \* mac电脑: python03 xxx.py  
'''

# input(): 用于在命令行工具中, 收集用户录入信息的函数  
  
# 细节1: python中需要写 ; 结尾  
# 细节2: python中的变量不需要声明, 类似于早期的js  
name = input('主人, 您的名字是?')  
  
print('欢迎', name, '的到来, 我是小助手 命运多舛')  
  
'''  
JS中变量声明关键词:  
\* 早期: 只有var 此时期可以省略  
  
\* ES6: let 和 const 带来了作用域特性; let 代替了 var; const属于新增的常量写法  
'''

# 变量类型  
'''  
在 JS 中:   
\* 基础变量类型: number(数字:整型+浮点) string字符串 boolean布尔(true/false)  
\* 空类型: null(赋值过,清空了) 和 undefined(从未赋值)  
\* 复合类型: array数组 object对象  
'''  
  
# python中的基础数据类型:  
  
# 整型: python把 js 的 number 类型细分为 整型int 和 浮点型float  
a = 5  
# type() 相当于js的 typeof()  
print(type(a)) # <class 'int'> int -> integer  
  
# 浮点: 带小数点和科学计数法  
a = 5.5  
print(type(a)) # <class 'float'>  
  
a = 5e2  
print(type(a)) # <class 'float'>  
  
# 布尔类型: 值为 大驼峰写法  
a = True  
print(type(a)) # <class 'bool'>  
  
a = False  
print(type(a)) # <class 'bool'>  
  
# 空  
a = None  
print(type(a)) # <class 'NoneType'>

# 字符串  
  
# 单行字符串: 使用 单引号 和 双引号 都可以  
a = 'Hello World!'  
a = "Hello World!"  
  
print(type(a)) # <class 'str'>  
  
# 多行字符串: 作者复用了 多行注释的写法 ''' '''  
a = '''  
床前明月光, 疑是地上霜.  
举头望明月, 低头写BUG.  
-- 命运轨迹  
'''  
print(a)  
  
  
# 转义符 \: 去掉字符串中 特殊字符的特殊含义  
word = "He said:\"I'm fine!\""  
# \" 可以把" 的定界符特征去掉, 变为普通的字符". 就不会冲突了  
print(word)  
  
word = "使用\\\" 可以显示普通的\""  
  
print(word) # 使用\" 可以显示普通的"

# 模板字符串: 变量放到字符串中进行显示  
  
name = '东东'  
age = 36  
  
# 输出 xxx今年xx岁  
# js中: `${name}今年${age}岁`  
  
# 变量拼接到字符串中的写法共有两种方案:  
# 方案1: 例如JS, 使用特定的包围来表示变量: 例如js的 ${}, php的 {}  
# 方案2: 例如node.js的数据库语句: 采用?占位符 select \* from xxx where name=? and age=?  
  
# 此处, python采用占位符方案实现  
word = '%s今年%s岁' % (name, age)  
print(word)  
  
  
name = 'iPhone12'  
price = '9999'  
year = '2021'  
  
# 目标显示: xxxx年的xxx, 售价xxx元  
word = '%s年的%s, 售价%s元' % (year, name, price)  
  
print(word)  
  
'''  
详解占位符:  
%s string 转化字符串显示, 原样显示  
%d digit 只显示整数位, 没有四舍五入  
%f float 浮点型, 6位小数  
%.nf 保留n位小数, 带四舍五入  
'''  
  
a = 12.6456  
  
print('%s %d %f %.2f' % (a, a, a, a))

# 下标取值  
  
word = 'ABCDEFGHIJKLMN'  
# 0123456789  
  
print(word[0]) # 从左向右 0  
print(word[-1]) # 从右向左 -1 开始  
print(word[2:]) # 从序号2开始到结尾  
print(word[2:5]) # 从序号2 ~ 4 : 含头不含尾, [2:5] 不含5  
  
# 通用的求长度函数:len()  
print(len(word))  
  
# 字符串相关方法  
word = 'nice TO meet YOU'  
  
print(word.upper())  
print(word.title())  
print(word.capitalize())  
print(word.lower())

# 复合数据类型: 数组  
  
# python把数组 细分成 4 种  
'''  
\* list: 就是js的数组  
\* tuple : 元组  
\* set : 集合  
\* dict : 字典  
'''  
  
# list 列表: 就是js的array  
  
skills = ['jQuery', 'ajax', 'axios', 'xhr', 'vue', 'vuex']  
  
print(skills)  
print(type(skills)) # <class 'list'>  
  
# 长度: 通用函数 len()  
print(len(skills))  
  
# 增  
skills.append('REACT') # 同js的push()  
print(skills)  
  
# 插入到指定位置  
skills.insert(1, 'ANGULAR') # 同js的 skills.splice(1,0,'ANGULAR')  
print(skills)  
  
# 删  
skills.pop() # 删除结尾元素; 对应js的pop()  
print(skills)  
  
skills.pop(1) # 删除指定序号元素; 对应js的 splice(i,1)  
print(skills)  
  
# 改  
skills[0] = '命运'  
print(skills)  
  
# 查  
print(skills[0])  
print(skills[-1])  
print(skills[2:])  
print(skills[2:5]) # 含头不含尾

'''
  
tuple: 元组类型
  
特点: 把数组的 增删改 操作去掉, 只剩下查功能 -- 只能看不能摸
  
总结: 安全
  
用途: 作为函数的返回值存在
  
'''
  
   
# 从 [] -> ()
  
a = ('vue', 'ajax', 'axios')
  
   
print(a)
  
print(type(a)) # <class 'tuple'>
  
   
# 元组不支持修改
  
# a[0] = 'mike'
  
   
print(a[0])
  
print(a[-1])
  
   
# 注意事项: 由于元组使用()进行声明
  
# 元组中至少一个 逗号,
  
b = ('mike',)
  
print(b)
  
print(type(b))

'''
  
set 集合
  
   
特征: 内部元素不重复, 但是去重不保证顺序
  
'''
  
   
a = {'mike', 'lucy', 'lucy', 'mike', 'lily', 'lily'}
  
   
print(a)
  
print(type(a))
  
   
# 用途: 通过强制转换类型的方式, 把 数组类型去重
  
   
arr = [1, 2, 3, 4, 4, 3, 2, 1]
  
   
arr\_unique = set(arr)
  
print(arr\_unique)

'''
  
dict 字典类型, 全称: dictionary
  
   
特征: 具有最高的查询速度
  
   
解释 JS 的对象类型
  
'''
  
boss = {
  
 "键": "值", # 键值对 结构
  
 'name': '文华',
  
 'age': 40,
  
 'gender': 1,
  
 'skills': ('vue', 'jQuery', 'Flutter', 'React')
  
}
  
   
print(boss)
  
print(type(boss))
  
   
# 增
  
# js中 boss.address = 'xxx'
  
boss['address'] = '北京市丰台区'
  
print(boss)
  
   
# 删
  
boss.pop('address')
  
print(boss)
  
   
# 改
  
boss['age'] = 18
  
print(boss)
  
   
# 查
  
print(boss['age'])

# 运算符
  
'''
  
数学运算符
  
比较运算符
  
逻辑运算符
  
三目运算符
  
'''
  
   
print(5 + 3)
  
print(5 - 3)
  
print(5 \* 3)
  
print(5 / 3)
  
print(5 % 3) # 取余
  
print(5 \*\* 3) # 次幂
  
   
# +的表现: 可以拼接两个字符串
  
print('5' + '3')
  
   
# 开发语言根据 强类型 和 弱类型 分成两个流派:
  
# 弱类型: php js 让系统根据语境 自动转化数据的类型进行适配; ---灵活但是有错误的可能.
  
# 强类型: java,c,python 严格要求表达式中的数据类型必须匹配语境; 不匹配则报错, 让用户来抉择.
  
   
# print(5 + '3')
  
   
print(5 + int('3'))
  
print(str(5) + '3')

# 比较运算符
  
   
print(5 > 3)
  
print(5 >= 3)
  
print(5 < 3)
  
print(5 <= 3)
  
print(5 == 3)
  
print(5 != 3)
  
   
'''
  
JS中: === 和 !==
  
弱类型语言会自动转换类型: 所以 == 和 === 出现
  
== 自动转类型比较
  
=== 不转类型比较
  
'''
  
print(5 == '5') # False
  
# python的== 相当于 JS的 ===, 因为py是强类型语言

# 逻辑运算符
  
'''
  
 大多数语言 python(语义化)
  
逻辑与 && and
  
逻辑或 || or
  
逻辑非 ! not
  
'''
  
   
# 逻辑与: 都真则真,有假为假
  
print(True and True)
  
print(True and False)
  
   
# 逻辑或: 都假则假,有真则真
  
print(True or False)
  
print(False or False)
  
   
# 逻辑非: 非真为假,非假为真
  
print(not False)
  
print(not True)

# 三目运算符
  
   
# 条件 ? 真值 : 假值
  
   
# python坚持 语义化特点: 外国人习惯倒装语法
  
# 真值 if 条件真 else 假值
  
   
married = False
  
   
print('已婚' if married else '未婚')

'''
  
流程控制语句:
  
\* 条件分支: if
  
\* 循环: for
  
'''
  
   
# 作者: 简化理念--懒
  
'''
  
if (条件){ 代码... }
  
'''
  
   
married = True
  
# 省略了 () 和 {}
  
# 必须遵循的规范: 由于没有了{}限定代码范围; 要求严格的缩进, 来代表包含关系
  
if married:
  
 print('已婚')
  
   
# 双分支语句: if 和 else 是同级别, 所以必须在一个缩进
  
if married:
  
 print("已婚")
  
else:
  
 print('未婚')
  
   
   
score = input('请输入文华的得分(0-100):')
  
# input()的返回值 肯定是字符串类型
  
score = int(score)
  
   
if score >= 90:
  
 print('优秀')
  
elif(score >= 60 and score < 90):
  
 # else if -> elif
  
 print('良好')
  
else:
  
 print('不及格')

# for循环
  
# python没有传统的for循环: for(let i=0; i<xx; i++){}
  
   
# python只提供 遍历数组写法
  
   
skills = ['vuex', 'vue', 'axios', 'ajax', 'json']
  
   
for item in skills:
  
 print(item)
  
   
# key?
  
# range(): 快速生成含有指定数量个数字的数组. 序号从0开始
  
for i in range(10):
  
 print(i)
  
   
print(list(range(10)))
  
   
skills = ['vuex', 'vue', 'axios', 'ajax', 'json']
  
# 0 1 2 3 4
  
# 序号的取法
  
for index in range(len(skills)):
  
 # arr[下标]: 利用下标读取对应的值
  
 print('index:', index, ' item:', skills[index])

# range 函数
  
   
a = range(10) # 10个数字, 从0开始
  
print(list(a))
  
   
a = range(10, 20) # 含头不含尾 10~19 没20
  
print(list(a))
  
   
a = range(10, 20, 2) # 参数3: 步长
  
print(list(a))

# 练习1: 计算 1~100之间所有数字的总和 5050
  
sum = 0
  
   
for i in range(1, 101):
  
 sum += i
  
   
print(sum)
  
   
# 练习2: 计算1~100 之间所有 偶数之和
  
sum = 0
  
   
for i in range(1, 101):
  
 if i % 2 == 0:
  
 sum += i
  
   
print(sum)
  
   
# 练习3: 计算 1~100 之间, 同时被 3 和 2 整除的数字 之和
  
sum = 0
  
   
for i in range(1, 101):
  
 if i % 2 == 0 and i % 3 == 0:
  
 sum += i
  
   
print(sum)

# 练习1: 打印 1990 ~ 2025 年之间所有的闰年组成的数组
  
# tips: 闰年 被400整除 或 不能被100整除但被4整除
  
years = []
  
   
for year in range(1990, 2026):
  
 if year % 400 == 0 or year % 100 != 0 and year % 4 == 0:
  
 years.append(year)
  
   
print(years)
  
   
# 练习2: 鸡兔同笼问题
  
# 笼子有若干鸡和兔子, 35个头 94只脚, 请问 鸡和兔子 各多少只?
  
# tips: 计算机速度快, 可以快速尝试所有组合 找到匹配的
  
for tu in range(36):
  
 # 0~35
  
 ji = 35 - tu
  
 if ji\*2 + tu\*4 == 94:
  
 print('鸡:%s, 兔:%s' % (ji, tu))
  
   
   
# 练习3: 百钱买百鸡
  
# 文华有100元, 媳妇让他买100只鸡 必须正好花光.
  
# 已知: 公鸡5元 母鸡3元 小鸡1元/3只. 帮文华找到所有购买方案
  
for gj in range(21):
  
 for mj in range(34):
  
 xj = 100 - gj - mj
  
   
 if gj\*5 + mj\*3 + xj/3 == 100:
  
 print('公鸡%s, 母鸡%s, 小鸡%s' % (gj, mj, xj))

# 冒泡排序
  
   
arr = [12, 78, 445, 556, 23, 56, 89, 67]
  
   
# 利用冒泡排序 把数组 从小到大排列
  
   
# 两两比较, 比较的次数是 总数量 - 1
  
count = len(arr)-1
  
   
for j in range(count):
  
 for i in range(count):
  
 # 两两比较 大的 向后移动
  
 if arr[i] > arr[i+1]:
  
 # 互换位置 a,b = b,a; python具备的写法
  
 arr[i], arr[i+1] = arr[i+1], arr[i]
  
   
print(arr)

'''
  
while: 负责不固定次数的循环
  
   
while 条件:
  
 xxxx
  
'''
  
   
# 制作 猜数字游戏:
  
import random
  
# 0-1000之间 随机获取数字
  
target = random.randint(0, 1000)
  
   
count = 0
  
   
while True:
  
 count += 1
  
   
 num = input('请猜测一个数字(0-1000):')
  
 num = int(num)
  
   
 if num == target:
  
 print('恭喜您, 猜对了, 共猜了%s次' % count)
  
 break
  
 if num < target:
  
 print('猜小了')
  
 if num > target:
  
 print('猜大了')

'''
  
continue
  
break
  
'''
  
   
for i in range(20):
  
   
 if i == 8:
  
 # 跳出当前次循环, 开始下一次
  
 continue
  
   
 if i == 11:
  
 # 终止整个循环
  
 break
  
   
 print(i)

## 作业

### 简易计算器

命令行中的效果如下:

$ 请输入一个数字: 3
  
$ 请输入运算符(+ - \* /): +
  
$ 请输入一个数字: 5
  
$ 5 + 3 = 8

### 求2~100之间所有的质数之和

质数: 只能被 自身 和 1 整除的数字 称为质数

## 预告

封装的技巧:

* 函数
* 面向对象
* 模块
  + 自定义模块
  + 系统模块
  + 第三方模块
  + 确保安装了mysql数据库, 下载 xampp 安装即可
      
    https://www.apachefriends.org/index.html