Chapter2 变量

变量名=字母+数字+下划线，数字不能开头 变量名避免使用大写字母

```
name = “banksy black”
#大小写间转换
print(name.title())
print(name.upper())
print(name.lower())
#消除前后空格
print(name.lstrip())
print(name.rstrip())
print(name.strip())
```

age = 23 转字符串 age.str() 错 str(age) 对

** 表示乘方

先编写行之有效的代码，再改进

Chapter3 列表

```
cars = [‘bwm’, ‘audi’, ‘toyota’]
#倒着访问
print(cars[-2]) = ‘audi’
#列表末尾添加
cars.append(‘xxx’)
#列表指定位置插入
cars.insert(0, ‘yyy’)
#角标删除 无返回值
del cars[1]
#角标删除 有返回值
carname = cars.pop(1)
#值删除 只删除出现的第一个
cars.remove(‘audi’)
#永久性排序 不能直接输出 可先排序 后 print(cars)
cars.sort()/cars.sort(reverse=True)
cars.reverse()
#暂时性排序 可直接输出
print(sorted(cars))
print(sorted(cars, reverse=True))
#返回列表长度
len(cars)
```

Chapter 4

indent : 缩进
python 根据缩进判断语句归属
#产生数值列表
range(1, 5) = 1 2 3 4 左开右闭
range(2, 11, 2) = 2 4 6 8 10 指定步长
#现在不是列表不能直接输出 但可通过 for 遍历
list(range(1,5))

#列表解析
squares = [value**2 for value in range(1, 11)]
=[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

#一些函数
print(min(squares))
print(max(squares))
print(sum(squares))

#列表切片后还是列表，可通过其进行复制列表
a_foods = [‘apple’, ‘banana’, ‘cherry’]
print(a_foods[-2:]) 从倒数第 2 个开始，直到最后
print(a_foods[1:3]) 从第“1”个开始，有 3-1=2 个
b_foods = a_foods[:] 对
b_foods = a_foods 错

#列表 list 内容可变[ ]，元组 tuple 内容不可变( ),
#但可以给存储元组的变量重新赋值，元组长度也可重新改变

Chapter5 if-elif-else

#用 in/not in 关键字判断列表是否包含某些内容
if ‘audi’ in/not in cars:
print(“true”)

Chapter6 dictionary { }

alien_0 = {
‘color’: ‘green’,
‘points’: 5, 最后一个逗号可有可无
}
#添加 key-value
alien_0[‘position’] = ‘east’

#遍历
for key, value in alien_0.items():
for key in alien_0
  =for key in alien_0.keys()
for value in alien_0.values()
for value in set(alien_0.values()) set()去重

三种方式返回的都是 list，直接输出和普通 list 有些区别，但都可通过 for 遍历

#列表中嵌套字典
for alien in range(30): 循环 30 次
for alien in range(0, 30): 循环 30 次

```
alien_0 = {'color': 'green', 'points': 5, 'speed': 'slow'}
alien_1 = {'color': 'yellow', 'points': 10, 'speed': 'medium'}
alien_2 = {'color': 'red', 'points': 15, 'speed': 'fast'}
aliens = [alien_0, alien_1, alien_2]
```

#字典中嵌套列表 需要在字典中将一个键关联到多个值

```
pizza = {
    'crust': 'thick',
    'toppings': ['mushrooms', 'extra cheese'],
}
```

#字典中嵌套字典

```
users = {'aeinstein': {'first': 'albert',
                       'last': 'einstein',
                       'location': 'princeton'},
         'mcurie': {'first': 'marie',
                    'last': 'curie',
                    'location': 'paris'},
         }
```

Chapter7 input & while

variable = input('提示信息')
#如果想接受 int 型输入信息 可用 int()
int() 和 str() 类似 实现类型转化

#如果有多个可以退出循环的点，可以使用标记 flag
while flag: 退出时 flag = false
while true: 退出时 break

for 循环利于遍历，不利于修改；while 同 list,tuple 结合可收集、存储并组织大量输入

Chapter8　function

#位置实参
　　def book(bookname)
　　book('Gone with the wind')
#关键字实参　多个参数时，实参顺序不影响
　　def book(bookname)
　　book(bookname='Gone with the wind')
#参数默认值
　　def book(bookname='Gone with the wind')

注意，非默认值关联到第一个参数，如果有多个参数，第一个参数不应有默认值
　　def book(bookname = "book1", bookname2):
　　　　print("my favourite book is "+bookname)
　　　　print("my favourite book is "+bookname2)
　　book("book2")　　错

　　def book(bookname, bookname2= "book2"):
　　　　print("my favourite book is "+bookname)
　　　　print("my favourite book is "+bookname2)
　　book("book1")　　对

#如果把列表传进函数，对列表的修改是永久性的　其他 int 类型/string 类型不会
#为了防止上述事情发生可以　传参传个切片副本　list1[:]，缺点是花费时间和内存

#传递任意数量的实参，多参情况下，这种实参往后稍一稍
　　def makepizza(*toppings)　　创建一个 toppings 空元组
　　makepizza('mushroom')
　　makepizza('mushroom', 'cheese')
#不知传入参数信息
　　def buildprofile(first, last, **user_info)　　创建一个 user_info 空字典
　　buildprofile('a', 'b', location='tianjin', sex='male')

#模块也是.py 文件
　　1. 先 import module_name　再调用 module_name.function_name()
　　2. 先 from module_name import function_name1, function_name2
　　　　（from module_name import *）
　　　　再直接调用　function_name1
#也可以通过 as 给函数/模块指定别名　import 结尾+as+别名

#函数名/模块名只用小写字母和下划线

Chapter9　class
#注意类的定义　class Restaurant():　　def __init__(self, restaurant_name, cuisine_type):

```
#所有类中方法的第一个参数都是 self

#类的继承 class ChineseRestaurant(Restaurant):
    def __init__(self, restaurant_name, cuisine_type):
        super().__init__(restaurant_name, cuisine_type)
#从模块中导入类与之前导入函数类似，不推荐导入模块中所有类

#类名采用驼峰命名法，类名的每个单词的首字母都大写，而不是用下划线
#实例名和模块名采用小写，并在单词之间加上下划线
#类中的方法之间空一行 模块中的类之间空两行
#先 import 标准库的模块，空一行之后在引入自己编写的模块

Chapter10  file

#open 是打开文件，with 是 python 会在合适的时候自动将其关闭
with open('pi_digits.txt') as file_object:
    contents = file_object.read()  整个文件一起读取
    print(contents)

#     for line in file_object:   逐行读取
#         print(line.rstrip())   删除空白行

#   file_object.readlines()  逐行读取  返回一个列表，再对列表进行处理即可

#写入空文件文件，如果不是空文件会先清空再写入
with open(filename, 'w') as file_object:
    file_object.write("i love programming.")

#追加内容到文件
with open(filename, 'a') as file_object:
    file_object.write("i'm lying.")

#try-except-else  块  try 中无错才会执行 else，跟 java 的 try-catch-finally 不同
#优点，避免让用户看到 traceback，防止恶意攻击；让程序往下执行，不因此终止
try:
    print(5/0)
except ZeroDivisionError:   Error 要么不写，要么匹配
    print("you idiot") / pass 关键字
else:
    ***

#将字符串分割为列表  wordlist = str.split()
#数字符串中子字符串出现次数  count = str.count()
```

```
#使用 json 存储数据
import json
numbers = [2, 3, 5, 7]
filename = 'numbers.json'
with open(filename, 'w') as file_object:
    json.dump(numbers, file_object)

import json
filename = 'numbers.json'
with open(filename) as file_object:
    numbers = json.load(file_object)
print(numbers)
```