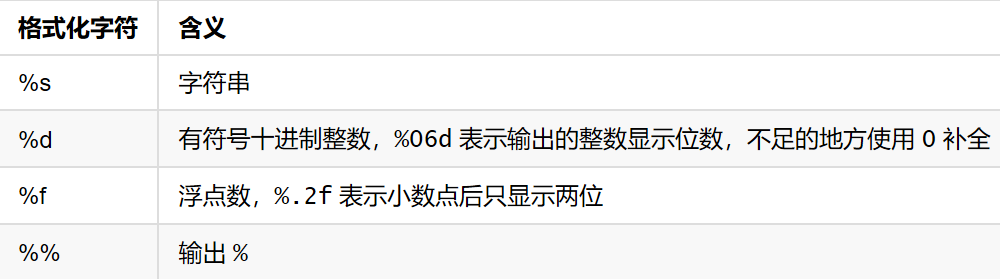
输入

name =input(“提示”)

字符格式



print(“%02d + %.2f = %.2f”%(a,b,c))

if语句使用

if 条件：

执行

elif 条件：

执行

else ：

执行

随机函数

import random

random.randint(a,b)---[a,b]

特殊运算符

//取整

%取模

\*\*乘方

逻辑运算符

and与

or或

not非

成员运算符

3 in (1, 2, 3) 返回 True

循环

while 条件：

执行

计数+1

break跳出循环

continue忽略一次循环

转义字符



函数方法定义

def 函数名(num)：

方法

（return 返回值）

传递参数

函数名(10)

模块函数

文件1 One中

def sum(num1,num2):

return num1+num2

文件2 Two中

import One

One.sum(3,5)

list列表

name\_list = [“a”,”b”,”c”]

name\_list.index(“a”,0,2) 查找范围左闭右开



循环遍历

for name in name\_list:

print(name)

元组

info\_tuple = ()

只包含一个元素时加“，” info\_tuple = (50, )

功能函数



遍历

for item in info:

print(item)

元组列表转换

list(元组)

tuple(列表)

字典 key：value

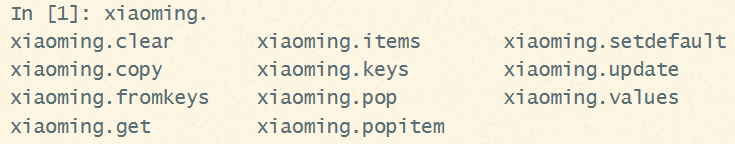
xiaoming = {"name": "小明",

"age": 18,

"gender": True,

"height": 1.75}

功能函数



遍历

for k in xiaoming:

print("%s: %s" % (k, xiaoming[k]))

字符串

string = "Hello Python"

for c in string:

print(c)

功能函数

判断



查找替换



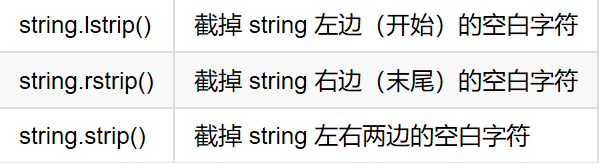
大小写转换



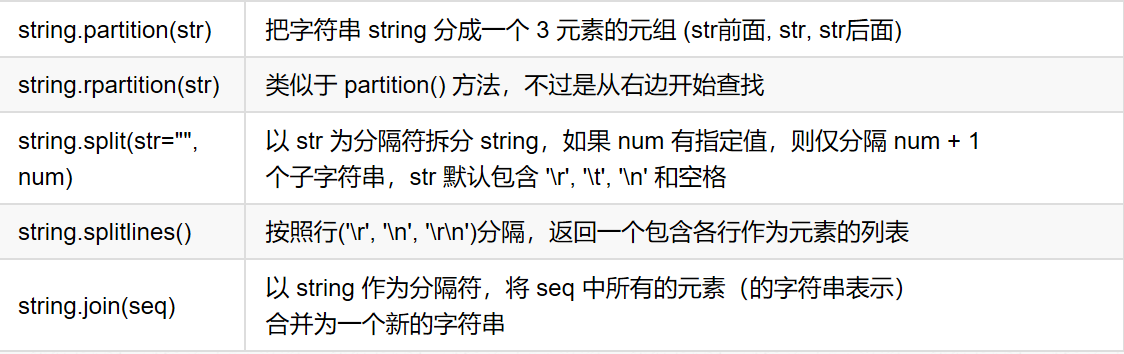
文本对齐



去除空白字符



拆分和连接



字符串切片

字符串[开始索引:结束索引:步长] 范围[开始索引, 结束索引)

头尾可以省略，最后一位 -1

截取从 2 ~ 5 位置 的字符串

print(num\_str[2:6])

从开始位置，每隔一个字符截取字符串

print(num\_str[::2])

逆序print(num\_str[::-1])

公共方法



格式转换

|  |  |  |  |
| --- | --- | --- | --- |
|  | 转换成列表 | 转换成元组 | 转换成集合 |
| 列表list | — | **tuple**(列表对象） | **set**(列表对象） |
| 元组tuple | **list**(元组对象） | — | **set**(元组对象） |
| 集合set | **list**(集合对象） | **tuple**(集合对象） | — |

shift+F6 统一修改变量名

修改全局变量

在函数中用global声明

支持多值参数的函数（\*元组，\*\*字典）

def demo(num, \*args, \*\*kwargs):

demo(1, 2, 3, 4, 5, name="小明", age=18, gender=True)

0~9 = 48~57

A~Z = 65~90

a~z = 97~122

asc码与字符转换

chr(1) ord(A)

三元

if a>b:  
 return a 🡺 a if a>b else b

else

return b

a if a>b else b

对象（类）

dir(类名) 查看该类所有方法

类的创建,封装(变量或方法前加\_\_)

class Person:

def \_\_init\_\_(self):

self.\_\_name = None

self.\_\_age = None

def get\_name(self):

return self.\_\_name

def set\_name(self, name):

self.\_\_name = name

def get\_age(self):

return self.\_\_age

def set\_age(self, age):

self.\_\_age = age

暴力访问：

对象名.\_类名\_\_私有成员

class 类名(object)

pass

查看继承关系 类名.\_\_mro\_\_

子类中调用父类中被重写覆盖的方法

super().方法名()

多继承

父类名.方法名(self)

默认继承第一个类

class A(B,C)

def a(self)

B.a(self) /C.a(self)

isinstance(a,list)判断a的类型

推导式可以快速生成数据存储结构中的数据

范例1：创建包含1到100所有整数的列表

list1 = [data **for** data **in** range(1,101)]

范例2：创建包含1到10的平方和的列表

list2 = [data\*\*2 for data in range(1,11)]

范例3：创建字典，key从1到5，value是key的平方

dict1 = [(data,data\*\*2) for data in range(1,6)]

可变参数

\*args 元组

\*\*kwargs 字典

\*/\*\* 拆包、组包

匿名函数

函数名 = lambda 形参 ： 返回值

结果 = (lambda 形参 ： 返回值)(实参)

单例模式

1.修改\_\_new\_\_

\_\_instance=None

def \_\_new\_\_(cls, \*args, \*\*kwargs):

if cls.\_\_instance is None:

cls.\_\_instance=object.\_\_new\_\_(cls)

return cls.\_\_instance

2.装饰器实现

def singe(cls):

\_instance = []

def wrap(\*args,\*\*kwargs):

if len(\_instance)==0:

\_instance.append(cls(\*args,\*\*kwargs))

return \_instance[0]

return wrap

文件操作

一

1.打开文件： file = open("文件名","读写模式")

2.操作文件： \*

3.关闭文件: file.close()

二

1.打开文件: with open("文件名","读写模式") as file:

2.操作文件: \*

3.关闭文件(自动关闭文件)

模式：

r w a …+ 读 写 追加 读+写

rb wb ab …+ 二进制读写

只读没有文件报错

read(1) readline()一行

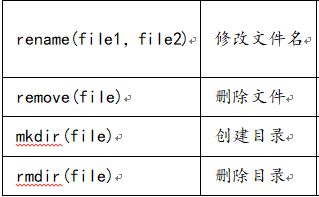
readlines()一行列表

file1 = open("3.txt","w+",encoding="utf-8") 改中文

os操作

import os

os.rename(“old”,”new”)



模块导入

导入格式：

from 模块名称 import 资源名称

from 模块名称 import 资源名称1，资源名称2

from 模块名称 import \*

控制导入资源

\_\_all\_\_ = ["资源名称1","资源名称2"]

屏蔽原始模块中的运行代码

if \_\_name\_\_ == "\_\_main\_\_":