导入模块/脚本时，将运行其中的所有代码。

当文件作为脚本运行时，\_\_name\_\_=="\_\_main\_\_"。

Python变量没有默认值。

解释器打印的是repr表示的内容。

所有的\回车被忽略。

## 运算符

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 优先级 |  | 用法 | 计算结果 | 副作用 |  |
|  |  |  | 编译器自动连接相邻的字符串。 |  |  |
| 1 | ()括号 |  | 改变运算符的结合顺序。 |  |  |
| {}字典 |  |  |  |  |
| []列表 |  |  |  |  |
| ()元组 |  |  |  |  |
| 2 | ()函数调用 |  |  |  |  |
| [::]切片 |  |  |  |  |
| []下标 |  |  |  |  |
| .访问成员 |  |  |  |  |
| 3 | \*\*幂 |  |  |  | ！！！也可以用pow函数完成。 |
| 4 | ~按位求补 |  |  |  |  |
| -单目相反 |  |  |  |  |
| +单目相同 |  |  |  |  |
| 5 | %求余 |  |  |  | 1、a%b等价于a-((a//b)\*b)。  2、a%b符号和b相同。 |
| //除以并取整 |  |  |  | 1、若**表达式1**和**表达式2**的数据类型都是int，得到的结果的数据类型是int；否则得到的结果的数据类型是float。 |
| /除以 |  |  |  | 1、得到的结果的数据类型是float。 |
| @矩阵乘 |  |  |  |  |
| \*乘 |  |  |  |  |
| 6 | -减 |  |  |  |  |
| +加 |  |  |  |  |
| 7 | >>按位左移 |  |  |  |  |
| <<按位右移 |  |  |  |  |
| 8 | &按位与 |  |  |  |  |
| 9 | ^按位异或 |  |  |  |  |
| 10 | |按位或 |  |  |  |  |
| 11 | !=不等于 |  |  |  |  |
| ==等于 |  |  |  |  |
| >=大于或等于 |  |  |  |  |
| <=小于或等于 |  |  |  |  |
| >大于 |  |  |  |  |
| <小于 |  |  |  |  |
| is not不相同性测试 |  |  |  |  |
| is相同性测试 |  |  |  |  |
| not in非成员资格检查 |  |  |  |  |
| in成员资格检查 |  |  |  |  |
| 12 | not逻辑非 |  |  |  |  |
| 13 | and逻辑与 |  |  |  |  |
| 14 | or逻辑或 |  |  |  |  |
| 15 | if else添加表达式 |  |  |  |  |
| 16 | lambda lambda表达式 |  |  |  |  |

## 标识符、关键字

1、标识符由小写字母、大写字母、数字、下划线组成，第1个字符是小写字母、大写字母、或下划线。

2、标识符区分大小写。

## 语句和控制转移

type类：每个对象代表1个数据类型

|  |  |
| --- | --- |
| type(对象) |  |

判断是否是None

|  |
| --- |
| 表达式 is None |

|  |
| --- |
| import inspect  inspect.getmro(**有\_\_mro\_\_属性的对象**)  inspect.getclasstree(inspect.getmro(**有\_\_mro\_\_属性的对象**))  import pprint  pprint.pprint(inspect.getclasstree(inspect.getmro(**有\_\_mro\_\_属性的对象**))) |

容器（序列、映射、集合等）

Sequence和Iterable区别

<http://blog.axant.it/2010/08/25/python-sequence-vs-iterable-protocols/>

判断1个对象的类是否是抽象类Iterable的子类

|  |
| --- |
| import collections  isinstance(**表达式**,collections.Iterable) |

序列（list、tuple、range、str）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 长度 | | len(**序列**) | |  |
| 取元素 | 索引 | **序列**[**整数**] | | 得到1个元素（的地址）。  1、下标是整数，最小-n，最大n-1。下标超出范围，抛出IndexError异常。 |
| 分片 | **序列**[(**整数**)?:(**整数**)?(:**整数**)?] | | 得到1个序列（由元素的地址组成）（当然不包括结束位置的元素）。  1、开始位置、结束位置、步长是整数。  2、步长不等于0。  3、若省略开始位置或结束位置，   |  |  |  | | --- | --- | --- | |  | 步长小于0 | 步长大于0 | | 开始位置 | 序列结尾 | 序列开头 | | 结束位置 | 序列开头之前 | 序列结尾之后 |   4、步长默认是1。  5、步长不等于1时无法使用分片赋值。  ！！！开始位置和结束位置可以超出范围，超出范围时算作序列开头之前或序列结尾之后而不是取模。 |
| 改 | | **索引**=**表达式** | 分片赋值 | （修改的是序列里存的地址，元素没动。） |
|  | （修改的是序列里存的地址，元素没动。下同。） |
| 删 | | del **索引** | （删的是序列里存的地址，元素没动。） |
| del **分片** | （删的是序列里存的地址，元素没动。下同。） |
| 追加 | | **序列**+**序列** | | 得到1个新序列。  1、2个序列数据类型相同。 |
| 重复 | | **序列**\***整数**  **整数**\***序列** | | 得到1个新序列。  **整数**不大于0时得到[]。  不能是数值为整数的float。 |
| 成员资格判断 | | **表达式** in **序列**  **表达式** not in **序列** | | 1、比较的是值，不是地址。 |

list

|  |  |  |  |
| --- | --- | --- | --- |
| 构造函数 | list()  list(**Iterable**) | |  |
| 插入 | **列表**.insert(**元素**) | 分片赋值 | 修改列表，得到None。 |
|  |  |
| 追加 | **列表**.append(**元素**) | 修改列表，得到None。 |
| **列表**.extend(**Iterable**) | 修改列表，得到None。 |
| 删元素 | **列表**.remove(**元素**) | 修改列表，得到None（只删第1个）。  1、若列表中没有该元素，抛出ValueError异常。 |
|  |  |
| 取元素+删元素 | **列表**.pop((**整数**)?) | | 得到1个元素（的地址）。  1、下标是整数，最小-n，最大n-1。下标超出范围，抛出IndexError异常。  2、省略下标，取和删最后1个元素。 |
| 倒置 | **列表**.reverse() | | 修改列表，得到None。 |
| 元素出现次数 | **列表**.count(**元素**) | | 1、比较的是值，不是地址。  ！！！只算列表直接包含的元素。 |
| 元素位置 | **列表**.index(**元素**) | | 1、若列表中没有该元素，抛出ValueError异常。 |
| （元素可比较） | **列表**.sort() | | 修改列表，得到None。  ！！！是稳定排序。 |
| 随机排序 | from random import shuffle  shuffle(**列表**) | | 修改列表，得到None。 |

|  |  |  |
| --- | --- | --- |
| 反向迭代器 | reversed(**列表**) |  |
|  |  |  |
|  |  |  |

tuple

|  |  |  |
| --- | --- | --- |
| 构造函数 | tuple()  tuple(**Iterable**) |  |

list相当于数组数据类型，元素数据类型一般相同，长度一般会改变；tuple相当于const结构体数据类型，元素数据类型一般不一定相同，创建后长度不改变。

set

|  |  |  |
| --- | --- | --- |
| 构造函数 | set()  set(**Iterable**) | 1、自动去除重复元素，下同。 |
| 长度 | len(**集合**) |  |
| 成员资格判断 | **表达式**in **集合**  **表达式**not in **集合** | 1、比较的是值，不是地址。 |
| 包含关系 | **集合**.issubset(**Iterable**)  **集合**.issuperset(**Iterable**)  **集合**<**集合**  **集合**<=**集合**  **集合**>**集合**  **集合**>=**集合** |  |
|  | **集合**==**集合**  **集合**!=**集合** |  |
| 并 | **集合**.add(**元素**) | 修改集合，得到None。  1、若列表中有该元素，不造成任何影响。 |
| **集合**.union(**Iterable**)  **集合**|**集合** | 得到1个新集合。 |
| 差 | **集合**.remove(**元素**) | 修改集合，得到None。  1、若列表中没有该元素，抛出KeyError异常。 |
| **集合**.difference(**Iterable**)  **集合**-**集合** | 得到1个新集合。 |
| 交 | **集合**.intersection(**Iterable**)  **集合**&**集合** | 得到1个新集合。 |
| 对称差 | **集合**.symmetric\_difference(**Iterable**)  **集合**^**集合** | 得到1个新集合。 |

|  |
| --- |
| sorted(**Iterable**)  得到1个新列表。 |
| reversed(**Sequence**)  得到反向迭代器。 |

~~类型对象~~类

input repr

import time

|  |
| --- |
| time()  得到当前时间，单位是s。 |
|  |

|  |  |
| --- | --- |
| help(\*args,\*\*kwds)  显示帮助。 | |
| input(prompt=None)  输入并返回字符串。 | |
| ascii(obj)  类似于repr，但非ASCII字符变成\\u**xxxx**。 | |
| repr | eval |
|  | |

|  |  |
| --- | --- |
| min(iterable,\*[,default=obj,key=func])  min(arg1,arg2,\*args,\*[,key=func]) | max(iterable,\*[,default=obj,key=func])  max(arg1,arg2,\*args,\*[,key=func]) |
| sum(iterable,start=0) |  |