基础知识：

# 对变量赋值时可声明数据类型，也可以直接赋值；变量名不可以以数字开头

|  |  |  |
| --- | --- | --- |
| 内置变量： | | |
| \_\_name\_\_ | "\_\_main\_\_" | 主代码 |
| "<mod>" | 作为模块被导入 |
| \_\_file\_\_ | 当前代码 | 文件 |
| \_\_cached\_\_ | 缓存文件 |
| \_\_package\_\_ | 当前包名 | |
| \_\_loader\_\_ | 模块导入方式 | |

|  |  |  |
| --- | --- | --- |
| 基本函数： | | |
| print(\*value, end="\n", sep=" ") | 输出 | end：结束符 |
| sep：分隔符 |
| input(str) | 输出str，读取输入的字符串 | |
| help(arg) | 查看帮助 | |
| callable(obj) | 判断是否可被调用 | |
| exit(code) | 终止进程 | |
| compile(source, filename, mode) | 编译实例 | |

|  |  |  |
| --- | --- | --- |
| 变量操作： | | |
| eval(str) | 字符串 → 变量 | |
| id(var) | 返回变量标识 | |
| type(obj) | 返回变量类型 | |
| globols() | 变量列表 | 全局 |
| locals() | 作用域 |
| dir(mod | obj) | 作用域 / 属性 |

|  |  |  |
| --- | --- | --- |
| 数值操作： | | |
| int(str, base) | 将对应进制的字符串转化成int | |
| bin(x) | 返回字符串 | 2进制，前缀0b |
| oct(x) | 8进制，前缀0o |
| hex(x) | 16进制，前缀0x |
| abs(x) | 返回数字绝对值 | |
| divmod(x, y) | 返回商、余数 (效率低下) | |
| pow(x, exp, mod) | 带模快速幂 | |
| round(x, decimals) | 改变浮点数精度 | |

|  |  |
| --- | --- |
| 基本语句： | |
| del <object> | 删除变量 |
| pass | 不做任何动作 |
| @callable | 将下一行的函数作为参数 |

|  |  |  |
| --- | --- | --- |
| ASCII码： | | 0x30 ~ 0x39：0 ~ 9 |
| ord(char) | 字符 → ASCII码 | 0x41 ~ 0x5A：A ~ Z |
| chr(ascii) | ASCII码 → 字符 | 0x61 ~ 0x7A：a ~ z |

|  |  |
| --- | --- |
| 分支循环： | |
| x if <bool> else y | 三目运算符 |
| while / if / elif / else | 分支常用语句 |
| for <var> in <iter>: | 变量循环获得序列里的值 |
| 循环过程中不能改 dict, list |
| 不覆盖使用变量 |
| break | 跳出循环体 |
| continue | 返回循环语句 |

|  |  |  |  |
| --- | --- | --- | --- |
| 转义序列： | | | |
| 振铃字符\a | 换行字符\n | 垂直制表符\v | 退格字符\b |
| 水平制表符 (以4个字符为一列)\t | | | 回车字符\r |

|  |  |
| --- | --- |
| 运算优先级： | |
| 幂运算 | \*\* |
| 位翻转 | ~ |
| 乘除 | \*，/，//，% |
| 加减 | +，- |
| 位移动 | <<，>> |
| 位运算 (有序) | &，^，| |
| 比较操作符 | >，>=，<，<=，==，!= |
| 逻辑运算符 | not，and，or |

# -：改符号位，并取补码

|  |
| --- |
| GIL全局解释器锁： |
| 每个线程执行时都需要先获取GIL (并发) |
| 在系统响应之前，GIL会暂时释放 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PyCharm： | | | | | | | | | |
| Ctrl + F | 查找 | Tab | 加缩进 | Ctrl + J | 代码提示 | Ctrl + Alt + L | | 格式化文件 | |
| Ctrl + R | 替换 | Shift + Tab | 减缩进 | Ctrl + / | 注释 | Ctrl + Alt + O | | 导入优化 | |
|  |  |  |  | F11 | 书签 | Ctrl + Shift + F10 | | 运行 | |
| IDLE： | | | | | | | | | |
| Ctrl + F | 查找 | Ctrl + ] | 加缩进 | Ctrl + N | 文件编辑 | Alt + M | 代码模块 | F1 | 帮助文档 |
| Ctrl + H | 替换 | Ctrl + [ | 减缩进 |  |  |  |  | F5 | 运行 |
| 代码提示 | ~/Lib/idlelib/autocomplete.py：import <目标模块>  ~/Lib/idlelib/config-extensions.def：popupwait= 0 | | | | | | | | |

异常处理：

|  |  |
| --- | --- |
| 异常语句： | |
| raise ERROR(str) | 引发指定解释的异常 |
| assert <bool>, <str> | 其后条件为False则报错 |

|  |  |
| --- | --- |
| Error基类： | |
| BaseException | 所有异常 |
| Exception | 常规错误 |
| StandardError | 内建标准异常 |
| ArithmeticError | 数值计算错误 |
| EnvironmentError | 操作系统错误的基类 |
| LookupError | 无效数据查询的基类 |

|  |  |
| --- | --- |
| Error常用： | |
| SystemExit | 解释器请求退出 |
| KeyboardInterrupt | 用户中断执行 (通常是输入^C) |
| StopIteration | 迭代器没有更多的值 |
| GeneratorExit | 生成器发生异常来通知退出 |
| AssertionError | 断言语句失败 |
| AttributeError | 实例没有这个属性 |
| IndexError | 序列中没有没有此索引 |
| KeyError | 映射中没有这个键 |
| SystemError | 一般的解释器系统错误 |
| TypeError | 对类型无效的操作 |
| ValueError | 传入无效的参数 |
| NameError | 变量名未定义 |
| SyntaxError | 语法错误 |

|  |  |
| --- | --- |
| try: | 检测范围 |
| except <Errors> as e: | 异常后的处理代码 |
| finally: | 必定执行的代码 |

迭代iter：

|  |  |
| --- | --- |
| 迭代器： | |
| \*<iter> | 迭代器裂解 |
| next(iter, default) | 弹出下一个值 |
| range(start, stop, step) | 返回线性序列 |
| enumerate(iter) | 返回携带偏移量的元组 |
| zip(\*iter) | 纵向拼接后转置 |

|  |  |  |
| --- | --- | --- |
| 切片： | | |
| <seq>[start: stop: step] | | 切片，[::-1] 逆转序列 |
| slice(\*args) | | 返回切片表达式 |
| 实例方法 | indices(len) | 裁剪切片防越界 |

|  |  |  |
| --- | --- | --- |
| 序列： | | |
| len(iter) | 返回序列长度 | |
| sum(iter, extra=0) | 返回累加值 | |
| max(iter, key) | 返回最大值 | key：键函数 |
| min(iter, key) | 返回最小值 |
| sorted(iter, key) | 非原地升序排序 |
| all(iter) / any(iter) | 返回元素bool运算 | |
| reversed(iter) | 颠倒元素顺序，返回生成器实例 | |

列表list (动态)：

|  |  |
| --- | --- |
| append(obj) | 添加元素 |
| extend(iter) | 扩充 |
| remove(obj) | **删除相等元素 (1)** |
| insert(i, obj) | **插入元素** |
| pop(i=-1) | 弹出对应元素 |
| count(obj) | 返回元素出现次数 |
| index(obj, start, end) | 返回元素的位置 |
| reverse() | 元素翻转 |
| sort(key, reverse) | 原地升序排序 |
| clear() | 清空列表 |

元组tuple (静态)：

|  |  |
| --- | --- |
| count(obj) | 返回元素出现次数 |
| index(obj, start, end) | 返回元素的位置 |

字符串str (静态)：

|  |  |  |  |
| --- | --- | --- | --- |
| 字符前缀： | | | |
| u | Unicode | r | 不识别转义 |
| b | bytes | f | 格式化 |

|  |  |  |
| --- | --- | --- |
| 大小写： | | |
| title() | 单词首字母大写 | |
| casefold() | 所有字母 | 改为小写 |
| upper() | 改为大写 |
| swapcase() | 翻转字母大小写 | |

|  |  |
| --- | --- |
| 端操作： |  |
| ljust / rjust / center(width, char=" ") | 使用char填充至指定宽度 |
| lstrip / rstrip / strip(char=" ") | 删去两端所有子字符串 |
| startswith / endswith(char, start, end) | 两端是否为子字符串 |

|  |  |  |
| --- | --- | --- |
| 查找： | | |
| count(sub, start, end) | 返回子字符串出现次数 | |
| find / rfind(sub, start, end) | 查找子字符串的位置 | 无则返回-1 |
| index / rindex(sub, start, end) | 无则报错 |

|  |  |
| --- | --- |
| 切分： | |
| partition / rpartition(sub) | 以子字符串为界切分成三元组 |
| split(sep=" ", time) | 切分字符串 |
| join(iter) | 作为分隔符插入序列 |

|  |  |  |
| --- | --- | --- |
| 修改： | | |
| <str>.replace(old, new, time) | 将旧字符串替换为新字符串 | |
| <str>.expandtabs(tabsize) | 修改水平制表字符数 (默认为4) | |
| <bytes>.decode(encoding) | 按给定的解码方式 | 解码 |
| <str>.encode(encoding) | 编码 |

|  |  |  |
| --- | --- | --- |
| 编码： | | |
| "unicode\_escape" | "utf-8" | "gbk" |

|  |  |
| --- | --- |
| 判断： | |
| isalpha() | 是否只包含字母 |
| isalnum() | 是否只包含字母、数字 |

|  |  |  |
| --- | --- | --- |
| format(\*obj) | 格式化字符串 | |
| %c | 字符、ASCII码 | |
| %s | 字符串 | |
| %M.Nd | 整数 | M限定数字最小总宽度  N制定小数点后精度  "-M"左对齐 |
| %M.Ne | e记法定点数 (1.5e10，1.5E10) |
| %M.Nf | 定点数 (小数点后精度默认6位) |
| %M.Ng | 根据值使用%e或%f |
| %o | 无符号8进制数 | |
| %x | 无符号16进制数 | |

|  |  |  |  |
| --- | --- | --- | --- |
| f"{<object>:mode}" | | f字符串 | |
| mode | <5：左对齐 | >5：右对齐 | ^5：居中对齐 |
| 千位分隔符：, \_ | | |

# 使用"{{"可打印"{"，"}}"可打印"}"

|  |  |
| --- | --- |
| 字符样式： | |
| \033[{font};{color};{bk\_color}m | 字符前缀 |
| range(108) | 样式编码 |

常量string：

|  |  |
| --- | --- |
| ascii\_uppercase | 大写字母字符流 |
| ascii\_lowercase | 小写字母字符流 |
| whitespace | 空白字符字符流 |
| punctuation | 标点符号字符流 |

正则re：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 正则表达式： | | | | | |
| . | 换行符之外的任意字符 | | | | |
| \d | 数字 (\D表非) | | | | |
| \s | 空白符 (\S表非) | | | | |
| \w | 字母、数字、下划线、汉字 (\W表非) | | | | |
| ^ | 置于开头，只匹配前缀 | | | | |
| $ | 置于结尾，只匹配后缀 | | | | |
| | | 或 | | | | |
| ( ) | 捕获组 (findall有效 / <Match>.group(1)读取) | | | | |
| [ ] | 字符类 | - | 在中间表范围 (\u4e00-\u9fa5表中文) | | |
| ^ | 在首位表非 | | |
| { } | 前一字符重复次数 | | | | |
| \* | 等价 {0,} | + | 等价 {1,} | ? | 等价 {0,1} |

|  |  |  |  |
| --- | --- | --- | --- |
| 编译标志RegexFlag： | | | |
| I | 忽略字母大小写 | M | "^"、"$"跨行匹配 |
| S | "."可匹配换行符 | **X** | **忽略**表达式**中的**空格和注释 |

|  |  |
| --- | --- |
| 匹配函数： | |
| compile(pattern, flags) | 编译的正则表达式 |
| findall(pattern, string, flags) | 匹配的字符串列表 |
| sub(pattern, repl, string, count, flags) | 替换子字符串 |
| split(pattern, string, maxsplit, flags) | 分割字符串 |

|  |  |  |
| --- | --- | --- |
| 匹配实例： | | |
| search(pattern, string, flags) | | 返回匹配结果 |
| match(pattern, string, flags) | | 返回前缀的匹配结果 |
| finditer(pattern, string, flags) | | 返回所有的匹配结果 |
| 实例方法 | group(i=0) / groups() | 匹配内容 |
| start(i=0) | 起始位置 |
| end(i=0) | 结束位置 |
| span(i=0) | 匹配范围 |

字典dict (动态)：

|  |  |  |
| --- | --- | --- |
| \*\*<dict> | 返回关键字参数 | |
| fromkeys(keys, fill) | 返回统一值的字典 | |
| keys() / values() / items() | 返回键值对的迭代器 | |
| get(key, value=None) | 查找值 | 不存在时返回默认值 |
| setdefault(key, value=None) | 不存在时添加进字典 |
| pop(key) | 弹出 | 键值对的值 |
| popitem() | 末尾键值对 |
| copy() | 返回拷贝 | |
| update(dict, \*\*kwargs) | 更新字典 | |

集合set (动态)：

# 哈希表存储结构，利于查找

|  |  |  |
| --- | --- | --- |
| 元素操作： | | |
| add(x) | 添加元素 | |
| remove(x) | 移除元素 (无则报错) | |
| discard(x) | 移除元素 (不报错) | |
| issubset(set) | 判断 | 子集 |
| issuperset(set) | 超集 |

|  |  |
| --- | --- |
| 运算操作： | |
| intersection(\*set) | 交集 |
| x & y |
| union(\*set) | 并集 |
| x | y |
| difference(\*set) | 差集 |
| x - y |
| symmetric\_difference(\*set) | 对称差集 |
| (x | y) - (x & y) |
| symmetric\_difference\_update(\*set) | 更新为对称差集 |

文本TextIO：

# with open() as <TextIO>: 打开的文本会自动关闭

|  |  |  |  |
| --- | --- | --- | --- |
| open(file, mode, encoding) | | | |
| 实例属性 | mode | 默认模式 | "r" 只读模式 |
| "t" 文本模式 |
| 写入模式  (若文件存在) | "w" 覆盖 |
| "x" 报错 |
| "a" 追加写入 |
| 特殊模式 | "b" 二进制模式 |
| "+" 可读写模式 |
| name | 文件名称 | |
| closed | 是否关闭 | |
| 实例方法 | read(limit=-1) | 读取指定个数的字符 | |
| readlines() | 读取行列表 | |
| write(str) | 写入内容 | |
| writelines(lines) | 写入行列表 | |
| close() | 关闭并保存 | |

函数function：

# <func>.\_\_doc\_\_：返回函数文档

|  |  |  |  |
| --- | --- | --- | --- |
| 定义函数 | def | 常规函数 | |
| lambda | 匿名函数 | |
| 返回值 | return | 执行时结束函数 | |
| yield | 将函数转变为生成器 (迭代器) | |
| <gen>.send(value) | 指定yield语句的产出值 |
| 变量声明 | global | 全局变量 | |
| nonlocal | 上一级函数的局部变量 | |
| 特殊参数 | <key>: <type> = ? | 设置参数类型、默认值 | |
| \*args | 收集剩余参数 | 返回元组 |
| \*\*kwargs | 收集剩余关键字参数 | 返回字典 |

|  |  |
| --- | --- |
| 映射： | |
| filter(func\_or\_none, iterable) | 保留返回值是True的元素 |
| map(func, \*iterables) | 返回映射结果 |

对象object：

# 类属性：可在实例化之前被设置

# 私有变量：self.\_\_var

|  |  |  |
| --- | --- | --- |
| vars(obj) | 实例属性字典 | |
| super(cls, obj=None) | 父类 | |
| issubclass(cls, cls|iter) | 判断是否 | 子类 |
| isinstance(obj, cls|iter) | 实例 |
| hasattr(obj, name) | 判断是否含有属性 | |
| setattr(obj, name, value) | 设置属性 | |
| getattr(obj, name, default) | 获取属性 | 不存在时返回默认值 (未提供则报错) |
| delattr(obj, name) | 删除属性 | 不存在时报错 |

|  |  |
| --- | --- |
| 惰性类变量： | |
| property(fget, fset, fdel, doc) | 惰性类变量 |
| @property | fget装饰器 |
| @<attr>.setter | fset装饰器 |
| @<attr>.deleter | fdel装饰器 |

|  |  |
| --- | --- |
| 装饰器： | |
| @staticmethod | 静态方法，无需self参数 |
| @classmethod | 类方法，需cls参数 |

|  |  |  |
| --- | --- | --- |
| 实例属性 | \_\_dict\_\_ | 实例属性字典 |
| \_\_class\_\_ | 所属类别 |
| \_\_doc\_\_ | 说明文档 |
| \_\_module\_\_ | 所属模块 |
| 基本方法 | \_\_new\_\_(cls, \*args, \*\*kwargs) | 先于\_\_init\_\_被调用，返回实例 |
| object.\_\_new\_\_(cls)：执行cls的 \_\_init\_\_ |
| \_\_init\_\_(self, \*args, \*\*kwargs) | **构造函数** (签名与 \_\_new\_\_ 相同) |
| \_\_del\_\_(self) | **析构函数** |
| \_\_call\_\_(self, \*args, \*\*kwargs) | **仿函数** |
| 类型转换 | \_\_int\_\_ / \_\_float\_\_ / \_\_complex\_\_(self) | int() / float() / complex() |
| \_\_bool\_\_ / \_\_str\_\_(self) | bool() / str() |
| \_\_round\_\_(self, n) | round() |
| \_\_iter\_\_ / \_\_next\_\_(self) | iter() / next() |
| 特殊操作 | \_\_len\_\_(self) | len() |
| \_\_dir\_\_(self) | dir() |
| \_\_enter\_\_(self) | with |
| \_\_await\_\_(self) | await, 未完成时yield, 完成时return |
| \_\_repr\_\_(self) | print() |
| 键值操作 | \_\_getitem\_\_(self, item) /  \_\_class\_getitem\_\_(cls, item) | <object>[item] |
| \_\_setitem\_\_(self, key, value) | <object>[key] = value |
| \_\_contains\_\_(self, obj) | in |
| 属性操作 | \_\_getattr\_\_(self, item) | 访问不存在的属性的行为 |
| \_\_getattribute\_\_(self, item) | 属性被访问的行为 |
| \_\_setattr\_\_(self, key, value) | 属性被设置的行为 |
| \_\_delattr\_\_(self, item) | 属性被删除的行为 |
| \_\_get\_\_(self, instance, owner) | 描述符被访问的行为 |
| \_\_set\_\_(self, instance, value) | 描述符被设置的行为 |
| \_\_delete\_\_(self, instance) | 描述符被删除的行为 |
| 比较**方法** | \_\_lt\_\_ / \_\_le\_\_(self, other) | <, <= |
| \_\_gt\_\_ / \_\_ge\_\_(self, other) | >, >= |
| \_\_eq\_\_ / \_\_ne\_\_(self, other) | ==, != |
| **算术方法** | \_\_neg\_\_(self) | 负号 |
| \_\_add\_\_ / \_\_iadd\_\_(self, other) | + |
| \_\_sub\_\_ / \_\_isub\_\_(self, other) | - |
| \_\_mul\_\_ / \_\_imul\_\_(self, other) | \* |
| \_\_truediv\_\_(self, other) | / |
| \_\_floordiv\_\_ / \_\_ifloordiv\_\_(self, other) | // |
| \_\_mod\_\_ / \_\_imod\_\_(self, other) | % |
| \_\_matmul\_\_ / \_\_imatmul\_\_(self, other) | @ |
| \_\_divmod\_\_ /\_\_idivmod\_\_(self, other) | divmod() |
| \_\_pow\_\_ / \_\_ipow\_\_(self, other[, mod]) | \*\*, pow() |
| 按位运算 | \_\_lshift\_\_ / \_\_rshift\_\_(self, other) | <<, >> |
| \_\_and\_\_ / \_\_or\_\_ / \_\_xor\_\_(self, other) | &, |, ^ |

模块module：

模块包目录下，\_\_init\_\_.py定义导入模块包时的动作

|  |  |
| --- | --- |
| help("modules") | 读取模块列表 |
| from <mod> import \* | 导入\_\_all\_\_列表中的变量 |
| from <mod> import <obj> | 从模块<mod>中导出<obj> |
| import <mod> as <var> | 导入模块并命名为<var> |

二分bisect：

|  |  |  |  |
| --- | --- | --- | --- |
| 升序 | bisect\_left(array, x) | 二分法查找x | 已有x时 → x位置 |
| bisect(array, x) | 已有x时 → x右侧 |
| insort(array, x) | 二分法插入x | |

复数cmath：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 数值操作： | | | | | | | |
| pi | п | isnan(z) | 判断nan | isinf(z) | 判断inf | isfinite(z) | 是否有限 |
| e | e | nan | nan | inf | ∞ | isclose(a, b) | 是否相近 |

|  |  |
| --- | --- |
| 属性访问： | |
| z.real | 复数实部 |
| z.imag | 复数虚部 |
| z.conjugate() | 对应共轭复数 |
| abs(z) | 复数的模 |
| phase(z) | 复数相角 (-п, п] |

|  |  |  |  |
| --- | --- | --- | --- |
| 运算函数： | | | |
| rect(r, phi) | 极坐标 → 复数 | | |
| polar(z) | 复数 → 极坐标 (r, φ) | | |
| sqrt(z) / isqrt(z) | z ^ 0.5 | pow(z, a) | z ^ a |
| sin(z) / cos(z) / tan(z) | 三角函数 | 正运算 | |
| asin(z) / acos(z) / atan(z) | 逆运算 | |
| exp(z) / log(z, base=e) | 指数运算 | | |

容器collections：

|  |  |  |  |
| --- | --- | --- | --- |
| 计数器： | | | |
| Counter(var, \*\*kwargs) | | 实例化计数器 (dict子类) | |
| 实例方法 | elements() | 返回元素迭代器 | |
| most\_common(int=None) | 返回指定数量高频值 | |
| update(var, \*\*kwargs) | 更新计数器 | 加法 |
| subtract(var, \*\*kwargs) | 减法 |

|  |  |  |
| --- | --- | --- |
| 有序字典： | | |
| OrderedDict(\*kwargs) | | 实例化有序字典 (dict子类) |
| 实例方法 | move\_to\_end(key, last=True) | 移动元素到结尾 / 开头 |

|  |  |
| --- | --- |
| 结构体： | |
| namedtuple(typename, field\_names) | 实例化结构体 |
| type(name, bases, dict) |

|  |  |  |
| --- | --- | --- |
| 双队列： | | |
| deque(iter, maxlen) | | 实例化限长队列 |
| 实例方法 | append / appendleft(obj) | 入队 |
| extend / extendleft(iter) | 迭代入队 |
| pop / popleft() | 出队 |
| insert(i, obj) | 插入元素 |
| count(obj) | 返回元素出现次数 |
| index(obj, start, end) | 返回元素的位置 |

拷贝copy：

|  |  |
| --- | --- |
| copy(obj) | 浅拷贝变量 |
| deepcopy(obj) | 深拷贝变量 |

日期datetime：

|  |  |  |
| --- | --- | --- |
| 日期： | | |
| datetime(year, month, day, hour=0, minute=0) | | 实例化日期 |
| 类方法 | today() | 当前日期 |
| fromtimestamp(t) | 秒数 → 日期 |
| strptime(date\_string, format) | 字符串 → 日期 |
| 实例方法 | date() | 返回日期实例 |
| time() | 返回时间实例 |
| weekday() | 返回0 ~ 6 (Mon ~ Sun) |
| timetuple() | 返回时间元组 |
| timestamp() | 返回秒数 |
| replace(year, month, day, hour, minute) | 更新日期 |

|  |  |  |
| --- | --- | --- |
| 时间差： | | |
| timedelta(days, seconds, minutes, hours, weeks) | | 实例化时间差  同类可加减比较，可与int乘除 |
| 实例属性 | days | 天数 |
| seconds | 秒数 |

分数fractions：

|  |  |  |
| --- | --- | --- |
| Fraction(numerator, denominator) | | 实例化分数 |
| 实例属性 | numerator | 分子 |
| denominator | 分母 |

函数functools：

|  |  |
| --- | --- |
| partial(func, \*args, \*\*kwargs) | 返回部分应用给定参数的函数 |
| reduce(func, seq, init=None) | 返回序列值逐次二元运算的结果 (比itertools.accumulate快) |
| lru\_cache(maxsize=None) | 返回结果缓存修饰器 (记忆化DFS神器) |
| wraps(func) | 返回带原函数信息的装饰器，修改被装饰函数的信息 |

堆heapq：

# 针对有 \_\_lt\_\_ 方法的对象

|  |  |
| --- | --- |
| 小根堆： | |
| heapify(list) | 原地小根堆化 |
| heappush(heap, item) | 添加堆结点 |
| heappop(heap) | 弹出堆顶，并重排 |
| merge(\*sorted, key, reverse) | 合并有序数列 |
| nsmallest(n, iter, key) | 返回升序前n元素 |
| nlargest(n, iter, key) | 返回降序前n元素 |
| heapreplace(heap, item) | pop → push (高效) |
| heappushpop(heap, item) | push → pop (高效) |

迭代itertools：

|  |  |  |
| --- | --- | --- |
| 运算： | | |
| accumulate(iter, oper=int.\_\_add\_\_) | 返回前缀和 | |
| groupby(iter, key) | 返回分组结果 (dict) | |
| permutations(iter, k) | 返回全排列 | |
| combinations(iter, k) | 返回全组合 | 元素无重复 |
| combinations\_with\_replacement(iter, k) | 元素有重复 |

|  |  |  |
| --- | --- | --- |
| 过滤： | | |
| compress(iter, bool\_seq) | 返回压缩过滤序列 | |
| takewhile(filter, iter) | 筛选满足条件的值 | while - break |
| dropwhile(filter, iter) | 滤除满足条件的值 |

|  |  |  |
| --- | --- | --- |
| 迭代器： | | |
| count(start, step) | 无尽线性序列 | |
| product(\*iter, repeat=1) | 返回笛卡尔积 | |
| islice(iter, start, stop, step) | 返回切片迭代器 | |
| chain(\*iter) | 返回级联迭代器 | |
| cycle(iter) | 返回循环迭代器 | 级联迭代器 |
| repeat(obj, times=None) | 重复元素 |

日志logging：

|  |  |
| --- | --- |
| DEBUG / INFO / WARN / ERROR / CRITICAL | 返回等级常量 |
| basicConfig(filename, filemode, level, format) | 设置日志格式 |
| %( )s：asctime, name, levelname, message |
| disable(level) | 禁用指定等级以下日志 |

|  |  |  |
| --- | --- | --- |
| 日志记录器： | | |
| getLogger(\_\_name\_\_) | | 返回记录器 |
| 实例方法 | info / warning / error(msg) | 记录信息 |

实数math：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 数值操作： | | | | | | | |
| pi | п | isnan(x) | 判断nan | isinf(x) | 判断inf | isfinite(x) | 是否有限 |
| e | e | nan | nan | inf | ∞ | isclose(a, b) | 是否相近 |

|  |  |  |  |
| --- | --- | --- | --- |
| 运算函数： | | | |
| sqrt(x) / isqrt(x) | x ^ 0.5 | pow(x, a) | x ^ a |
| factorial(x) | x! | prod(iter) | 累乘 |
| perm(n, k) | 排列数，P = n! / (n - k)! | | |
| comb(n, k) | 组合数，C = P / k! | | |
| sin(x) / cos(x) / tan(x) | 三角函数 | 正运算 | |
| asin(x) / acos(x) / atan(x) | 逆运算 | |
| exp(x) / log(x, base=e) | 指数运算 | | |
| ceil(x) / floor(x) | 取整 | | |
| degrees(x) / radians(x) | 弧度 <-> 角度 | | |
| dist(p, q) | 欧式距离 | 点 → 点 | |
| hypot(\*coord) | 点 → 原点 | |
| gcd(a, b) / lcm(\*int) | 最大公约数 / 最小公倍数 | | |

依赖pkg\_resourse：

|  |  |
| --- | --- |
| require(str | iter) | 返回依赖包信息 |
| parse\_requirements(strs) | 解析依赖文件字符串 |

随机random：

|  |  |  |
| --- | --- | --- |
| float | random() | [0,1) |
| uniform(start, end) | [start, end] |
| int | randint(start, end) | [start, end] |
| randrange(start, end, step) | range(start, end, step) |
| getrandombits(bits) | k位二进制 |
| obj | choice(iter) | 单次选取 |
| choices(iter, weights, cum\_weights, k) | 按概率选取 |
| sample(iter, k) | 不放回选取 |
| list | shuffle(list) | 原地打乱 |

时间time：

|  |  |
| --- | --- |
| 时间操作： | |
| time() | 获得当前秒数 |
| sleep(secs) | 睡眠指定时间 |

|  |  |  |
| --- | --- | --- |
| **时间元组struct\_time：** | | |
| **tm\_sec** | **%S** | **0~61** |
| **tm\_min** | **%M** | **0~59** |
| **tm\_hour** | **%H** | **0~23** |
| **tm\_mday** | **%d** | **1~31** |
| **tm\_mon** | **%m** | **01~12** |
| **%b** | **Jan ~ Dec** |
| **%B** | **January ~ December** |
| **tm\_year** | **%Y** | **2001** |
| **tm\_wday** | **%a** | **Mon ~ Sun (0 ~ 6)** |
| **%A** | **Monday ~ Sunday** |
| **tm\_yday** | **年中第几天** | **1 ~ 366** |
| **tm\_isdst** | **夏令时真值** | **0,1,-1 (代表夏令时)** |

|  |  |  |  |
| --- | --- | --- | --- |
| 时间转换： | | | |
| 秒数 | ctime(seconds=None) | 秒数 → 字符串 | %a %b %d %H:%M:%S %Y |
| gmtime(seconds=None) | 秒数 → 时间元组 | 格林威治时间 |
| localtime(seconds=None) | 当地时间 |
| 时间元组 | mktime(t\_tuple) | 时间元组 → 秒数 | |
| strftime(format, t\_tuple) | 时间元组 → 字符串 | |
| 字符串 | strptime(str, format) | 字符串 → 时间元组 | |

计时timeit：

|  |  |  |
| --- | --- | --- |
| Timer(stmt, setup) | 计时器对象 | stmt：代码段字符串 |
| timeit(stmt, setup, number) | 运行耗时 | setup："from \_\_main\_\_ import <object>" |
| repeat(stmt, setup, repeat, number) | 运行耗时列表 | number：代码执行次数 |

进度tqdm：

from tqdm.notebook import tqdm

|  |  |  |
| --- | --- | --- |
| tqdm(iterable, desc, total, file) | | 进度条装饰的迭代器 |
| 实例方法 | set\_describption(desc) | 设置进度条前缀格式 |
| set\_postfix(dict) | 属性补充 |
| reset(total) | 重置进度条 |
| update(n) | 步进进度条 |

回溯traceback：

|  |  |
| --- | --- |
| print\_exc() | 输出报错信息 |

类型typing：

|  |  |
| --- | --- |
| 类型声明： | |
| Any | 任意 |
| TypeVar(str) | 变量 |
| Optional[type] | type | None |
| Union[\*type] | type | ... |
| List / Tuple / Dict / Set / Sequence | iterable |
| Callable[[\*args], result] | function |
| Type[father] | class |

警告warnings：

|  |  |
| --- | --- |
| filterwarnings("ignore") | 忽略所有警告 |
| warn(msg) | 输出警告 |

命令行cmd：

环境conda：

|  |  |  |
| --- | --- | --- |
| conda create -n <envs> python==3.8.0 | 克隆 | 环境 |
| conda activate <envs> | 激活 |
| conda base | 关闭 |
| conda info --envs | 查看虚拟环境列表 | |
| conda remove -n <envs> --all | 删除虚拟环境 | |
| conda upgrade --all | 升级全部包 | |
| conda clean -y --all | 清理安装包 | |
| conda install -y python==3.7 anaconda=custom | 安装指定Python版本 | |
| conda config --show | 查看配置 | |

笔记jupyter：

|  |  |
| --- | --- |
| notebook.auth | |
| passwd(str) | 字符串 → 密文 |

|  |  |  |
| --- | --- | --- |
| 命令行： | | |
| jupyter notebook | | 启动jupyter |
| jupyter notebook --generate-config | | 生成配置文件 |
| 配置  文本 | c.NotebookApp.ip = "\*" | 对外开放的IP |
| c.NotebookApp.allow\_remote\_access = True | 允许远程访问 |
| c.NotebookApp.password | 设置密文 |
| c.NotebookApp.port = 8888 | 开放端口 |

|  |  |
| --- | --- |
| 代码： | |
| %%time | 输出执行时间 |

|  |  |  |  |
| --- | --- | --- | --- |
| 快捷键： | | | |
| shift + Tab | 函数文档 | Tab | 补全建议 |

安装pip：

|  |  |  |
| --- | --- | --- |
| pip install --upgrade <package> -f <url> | 升级包 | |
| pip freeze > requirements.txt | 导出 | 依赖包文件 |
| pip install -r requirements.txt | 安装 |

运行python：

|  |  |
| --- | --- |
| python -V | 查看当前Python版本 |
| python <main.py> | 直接运行程序 |
| python -m <main.py> | 以模块形式运行程序 |