Python 算术优先级：

乘方**➔一个数自身+-➔数之间乘除 * / % // ➔数之间加减➔左右移<< >>➔And➔OR➔比大小➔相等== 不等!= ➔赋值

/是浮点除；//是向下取整(负数更小)；X %= 10➔X = X % 10

每个+操作符不改变数的正负性，而每个-操作符会翻转数的正负性。

逻辑：

空字符：False；非空为 True

所有值都有真假性，非空即为真

None != False，但在逻辑运算中会被当作 False。

优先级：not – and - or

Not 0 == True; Not 1 == False

True + 1 == True

_ or _:若前面为真，则停止，不会读到后面；_ and _: 若前面为假，则直接停止，不会读到后面

多重比较,从左到右进行评估：e.g. False == True == False，其实是 False == True and True == False

any(iterable)： 只要迭代器中有一个值为 True，返回 True。all(iterable)： 只有当迭代器中所有值都为 True，返回 True。

Slicing: 1ist[start 闭:end 开: step]

1.start >end ➔ 返回空字符；

2.start > len(list) ➔ 从最后开始；

3.end<0➔从右边开始算，end 负出 start➔到最左端 start 停止。Step 为负，从右到左切片

4.list[1:0]第二个到最后，list[:]整个列表

5.list[-1:]最后一个, list[:1]第一个

用 list index 的时候，最后一定要记住% len(list)，可以使 index 限定在 len 中，不超索引。

Str, list, tuple 相加➔拼接，相减会报错。不同 type 不能直接相加

字符串操作包括**大小写转换（upper/lower）**、**分割（split）**、**去除空格或指定字符（strip）**、**替换（replace）**、查找子字符串（find 找不到返回-1 或 index 抛异常）、判断前缀后缀（startswith/endswith）。

字符串比大小，只比第一个不同的 ASCII 值，小写字母的 ASCII 比大写的大。

列表元素之间要有逗号

简化创建列表，如[x**2 for x in range(5) if x > 2]。

list.sort(): 原地排序; sorted(list):返回排序后的新列表。

Sqrt 等需调包的函数，没包就报错

嵌套的 **Lambda** 函数，从左到右，由内到外地把 lambda 消去。

Lambda 函数的输入如果在 for loop 中，只会捕获 for loop 的最终值。例如 lambda x: i + x for i in range(3)，此处的 i 只会是 3。

集合: &:交集；|:并集；-:差集；^:对称差集(symmetric)

List index 超出 len 后会返回[]，空列表布尔值为 False

Map(function, iterable) iterable 可以是任何类型数据，数据依次进入 function 中。

filter(function, iterable) iterable 依次进入可以判断布尔值的函数。若 function 的返回值为 0 或 False，则该元素会被过滤掉。

sorted( ) 会将输入的字符串按字典顺序排序，返回包含排序后字符的列表。例：'abracadabra' 排序后为：['a', 'a', 'a', 'a', 'a', 'b', 'b', 'c', 'd', 'r', 'r']。

reduce(function, iterable)将 iterable 中的前两个元素传递给 function，并计算结果，将结果与序列中的下一个元素再次传递给 function。

isinstance(object, classinfo)（需检查的对象，需比较的类或类型）会返回一个布尔值，如果 object 是指定类型 classinfo 则返回 True。

‘’.join( )将散的拼成一个字符串

round()当一个数正好位于两个整数之间（例如 2.5，位于 2 和 3 之间），舍入到最近的偶数。

List.append( ):在最后加入，List.extend( ): 塞进来 list

d[i] = d.get(i, 0): 从 dict 中获得 key i 的 value，若无 key i，则返回 0

Read File: r：只读模式。**rb：**以**二进制只读**模式。**r+：读写模式**。**rb+：**以**二进制读写**模式。**w：只写**模式，。**wb：**以**二进制只写**模式。**w+：**以**读写**模式。**wb+：二进制读写**模式。**a：追加**模式。**ab：二进制追加**模式。**a+：读写追加**模式。**ab+：二进制读写追加**模式

Type	iterable	mutable	indexable	hashable
int	-	-	-	Yes
flo	-	-	-	Yes
str	Yes	-	Yes	Yes
tuple	Yes	-	Yes	Yes
list	Yes	Yes	Yes	-
set	Yes	Yes	-	-
dic	Yes	Yes	Yes	-

Dict 需要键值对，set 不能嵌套 set，不能包含 list。list(dict) 将字典转换为列表，只会返回字典的键。

在 dict 中，如果插入的 key 已存在，则会覆盖该键原有的值。设置键值对（**d[k] = v**）、获取键值（**d[k]**）、删除键值对（**del**）以及检查键是否存在（**in**）的复杂度均为 **O(1)**。计算字典大小（**len**）也是 O(1)。而获取所有键（keys）、所有值（values）或所有键值对（items）的复杂度为 O(n)。将字典转为列表（如 list(dict)）时，只会提取字典的键并转化为列表。

Debug：try 块中的代码尝试执行，如果没有异常发生，代码顺利运行。如果发生异常（例如语法错误、运行时错误等），程序会跳转到 except 块。

Except 可以同时捕获多个异常类型，例如 except (TypeError, ValueError)。**Finally**:无论是否发生异常，finally 中的代码都会被执行，用于清理资源。

NameError：在赋值前使用变量名；TypeError：函数传参类型错误；ValueError：参数值无效（如 log(-1)）；ZeroDivisionError：除数为零；StopIteration：迭代器无更多项；IndexError：序列索引超范围；KeyError：请求不存在的键；IOError：I/O 操作失败（如文件打开失败）；EOFError：文件或控制台输入到达结尾；AttributeError：访问未定义的对象属性。

Algorithm: Bubble Sort: n ~ n**2; Merge Sort: nlogn; BFS: 1 ~ V+E, (V for node num, E for edge num); DFS: 1 ~ V+E; Dijkstra: V**2.

Deep 递归最关键在中间一步，中间可以替换成其他函数，比如 deepmap，上下两步基本思路一致。

```
def deepcount(seq):
        if seq == []:
                return 0
        elif type(seq) != list:
                return 1
        else:
                return deepcount(seq[0]) + deepcount(seq[1:])
```

```
def deepMap(func,seq):
        if seq == []:
                return seq
        elif type(seq) != list
                return func(seq)
        else:
                return [deepSquare(func,seq[0])] + deepSquare(func,seq[1:])
```

OOP：子 class 在 init 时，可用 super( ).__init__(x)继承父 class 中 init(self, x)的内容。

双父 class 时，越近的优先级越高。

可以在 class 中的函数名前面加”__”，让这个函数不能被继承 e.g.def __name(self)

多继承中的 **MRO** 方法解析顺序：当类有多个父类时，MRO 规则：**子类优先**，先调用当前类中的方法；**深度优先**，在多个父类中，按从左到右的继承顺序依次查找；**避免重复**，确保每个类只被调用一次。（做这种题需要**画出继承图**，把继承顺序搞清楚，追踪确定第一个输出结果来自于哪个 class，然后按照调用栈，倒推回来依次确定打印结果）