1. 基础语法
2. 基本数据类型：
3. 数字：整数（int），浮点数（float），复数（complex），布尔（bool）
4. 字符串（String）
5. 列表（list）：有序的可变序列
6. 元组（Tuple）：有序的不可变序列
7. 集合（Set）无序不重复集合
8. 字典（Dictionary）：无序键值对集合
9. 注释：
10. 单行注释：#
11. 多行注释：“”“”“” 即三个双引号
12. 变量：（1）和之前学的编程语言一致

（2）检测数据类型：type（要检查的变量）

4、命名规则：（1）标识符：只能是字母、数字、下划线、中文（不完善）

（2）大小写敏感

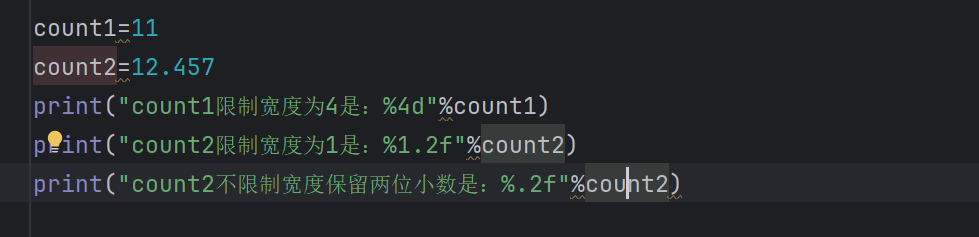
（3）不能使用关键字

5、字符串格式化：（1）语法：在双引号中放入占位符，后面进行对应即可



（2）种类：%s 字符串的占位 %d 整型的占位 %f 浮点型的占位

（3）数字的精度控制：当宽度限制小于数字本身不会生效



1. 快速格式化：缺点：不限制类型，不做精度控制

语法：图形用户界面, 文本

描述已自动生成

1. 对表达式格式化：即将乘除之类的不赋值，直接放入到字符串中

文本

描述已自动生成

6、数据的输入

（1）作用：用户输入信息

（2）语法：num1=input(“这里可以放入提示信息”)

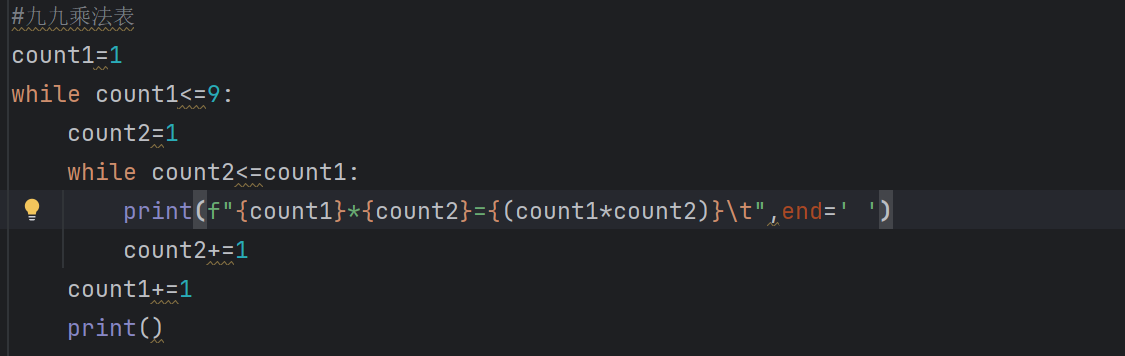
（3）注意：输入的信息都认为是字符串，如果要赋值给其他类型变量，需要进行强转

7、布尔类型和比价运算符：基本与Java和C++保持一致

8、判断结构：语法：if 判断标准: ……

elif 判断标准：……

else: ……

9、循环结构：（1）while循环：

（2）for循环：类似于Java中的增强for

for 临时变量 in 待处理的数据集（字符串、列表、元组等）

range语句：range（num）：获取一个从0开始，到num结束的数字序列（不包含num本身）

range（num1，num2），从num1开始到num2结束，不包含num2

range（num1，num2，step），step为步数，即每次递增的值，如2,4,6，步数就是2

（3）for循环的作用域：在Python中，是通过缩进来控制作用于的，即对齐的为统一作用域，如果不对齐，则更换作用域

（4）终结语句：break和continue，用法与Java和C++保持一致

10、函数

（1）语法：def 函数名（参数）：

函数体

return 返回值

（2）注意：当没有返回值的时候，返回的是None，在判断中相当于false

基本的结构等各方面和Java基本一致

11、数据容器

（1）定义：类似于Java中的集合和数组的结合

（2）List：就是Java中的单列集合List，可以存储不同的数据类型，可以嵌套

查询功能：列表.index（元素），返回该元素的下标

插入：列表.insert（位置索引,要插入的内容），此时插入内容的索引就是该位置索引，原来的在这索引之后的元素都自动向后移位

追加：方式一：列表.append（要加入的内容）

方式二：列表.extend（其它容器）

删除：方法一：del 列表[下标]

方法二：列表.pop（下标） 此方式会有返回值，即删除的元素

方法三：列表.remove（要删除的元素），会从前向后，删掉第一个出现的该元素

清空：列表.clear（）

统计某元素出现的个数：列表.count（元素）

容器中元素个数：len（列表）

（3）元组：定义：是一个容器，内容不可以被修改

语法：变量名=（元素1、元素2、元素3） //可以是不同的数据类型

变量名=（） //空元组

变量名=tuple（） //空元组

常用方法：index（） 查找某个元素，返回下标值

Count（） 统计出现某个元素的次数

Len（元组） 元组中元素的个数

特点：允许数据的重复，按照下标有序排列，不允许修改

1. 字符串（str）：

负下标：在Python中支持数组使用负下标，即假设长度为10的数组，正下标为0的元素，负下标为-10；此后一次递增，一直递增到-1

特点：无法修改，修改只会得到一个新的字符串

常见方法：

文本

描述已自动生成

文本

描述已自动生成

1. 序列

定义：指内容连续，有序，可以使用下标索引的一类数据容器

种类：列表、元组、字符串

常见方法：切片：序列[起始下标：结束下标：步长]

注意：使用字符串切片的时候，可以使用负下标，不过起始和结束下标要和正下标反过来

图形用户界面, 文本

中度可信度描述已自动生成

12、集合

（1）特点：去重、无序

（2）语法：变量名={元素1、元素2、元素3}

变量名=set（） #定义空集合

常见方法：文本, 聊天或短信

描述已自动生成

文本

描述已自动生成

13、字典

（1）特点：类似于Java中的双列集合，存在键值对，键除了字典外，可以是任意数据类型，值可以是任意数据类型

（2）语法：文本

中度可信度描述已自动生成

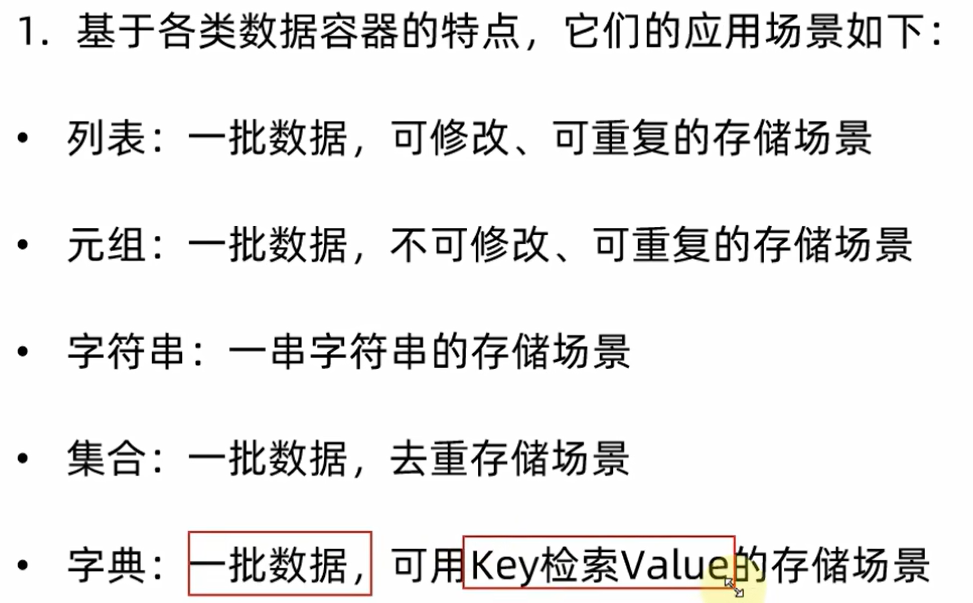
（3）常见方法：文本

描述已自动生成

文本

描述已自动生成

14、数据容器的总结：



15、数据容器的通用操作

（1）类型转化：转化后的容器类型（转化前的容器类型名）

图形用户界面

中度可信度描述已自动生成

（2）排序：默认为从大到小：sorted（排序容器名）

倒序排序：sorted（排序容器名，true）

16函数进阶

（1）多个返回值：图形用户界面, 应用程序

中度可信度描述已自动生成

注意：可以同时接收不同类型的返回值

（2）函数的多种参数使用方式：位置参数、关键字参数、缺省参数、不定长参数

位置参数：传入参数和定义的参数的顺序和个数必须一致

关键字参数：

缺省参数：

图形用户界面, 文本, 应用程序

描述已自动生成

不定长参数：

图片包含 文本

描述已自动生成

图形用户界面

低可信度描述已自动生成

（3）函数参数：

文本

描述已自动生成

注意：只能使用一行

17、文件操作

（1）打开文件：语法：open（name，mode，encoding）

name：文件名称（可以包含文件所在的具体路径）

mode：设置打开文件的模式（访问模式）：只读、写入、追加

encoding：编码格式

（2）mode常见的三种基础访问方式：r：只读方式代开文件。文件的指针将会放在文件的开头。

w：写入，如果文件已经存在，会删除原来的内容，将新的内容写入，如果没有则会进行创建

a：打开一个文件用于追加，如果该文件已经存在，新的内容会写在已有内容的后面，如果不存在，则会进行创建

（3）常用的读操作：

读取文件的所有字节：图形用户界面, 应用程序

描述已自动生成

读取所有行：

文本

描述已自动生成

读取一整行：

形状

低可信度描述已自动生成

for循环进行读取：

图片包含 文本

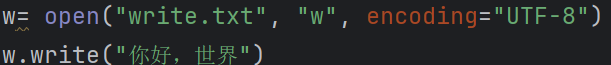
描述已自动生成

文件自动关闭：

文本

描述已自动生成

（4）文件的写操作

文件写入：

内容刷新：文本

描述已自动生成

（5）文件的追加：和文件的写入方法基本一致，区别在于，如果是已经存在的文件，则会在原有的内容上进行写入

18、异常

（1）异常的捕获（处理异常）：

语法：try：

可能错误的代码

except：

出现异常后执行的代码

多个异常：当指定异常进行捕获的时候，出现其它异常会直接报错，指定捕获，方便之后的处理。注意：异常是可以赋值给变量的

因此一般可以这样去捕获多个异常

文本

描述已自动生成

（2）else：当程序没有出现异常时，会执行else中的代码

（3）finally：无论是否出现异常都会执行，一般使用在关闭文件的操作中

（4）异常的传递性：当在对象中，或者是函数调用另一个函数的时候，异常会出现传递性，假如异常在函数一种没有捕获，会传递给另一个调用的函数，此时，函数2也是可以进行捕获的，假如都没有捕获，则会报错

图形用户界面, 文本, 应用程序

描述已自动生成

19、Python模块

（1）定义：Python中模块相当于Java中的API，是一个工具包，以.py结尾，提供类、函数、变量等

（2）导入：文本

描述已自动生成

（3）注意：一般直接调用里面的方法，如果没有导包，软件也会帮你进行导包

图片包含 图表

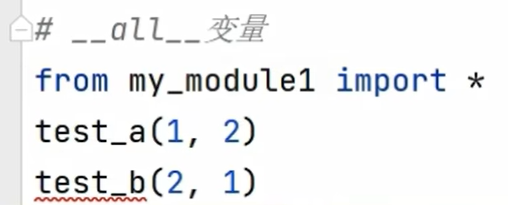
描述已自动生成

（4）自定义模块：当需要频繁使用一些功能时，就可以将其模块化，即自定义模块

（5）\_ \_main\_ \_：图形用户界面

描述已自动生成

（6）\_ \_all\_ \_：当使用\*导入的时候，如果使用\_ \_all\_ \_时，则可以选择性的导入一部分内容



文本

中度可信度描述已自动生成

注意：该效果只作用在利用\*进行导模块的操作中

20、Python包

（1）定义：表面上看，是一个文件夹；本质仍然是一个包

（2）作用：管理模块文件

（3）自定包：图形用户界面, 应用程序

描述已自动生成 第一个\_init\_必须存在，不然不是包

文本

描述已自动生成

21、第三方包：

（1）著名的第三方包：

文本, 信件

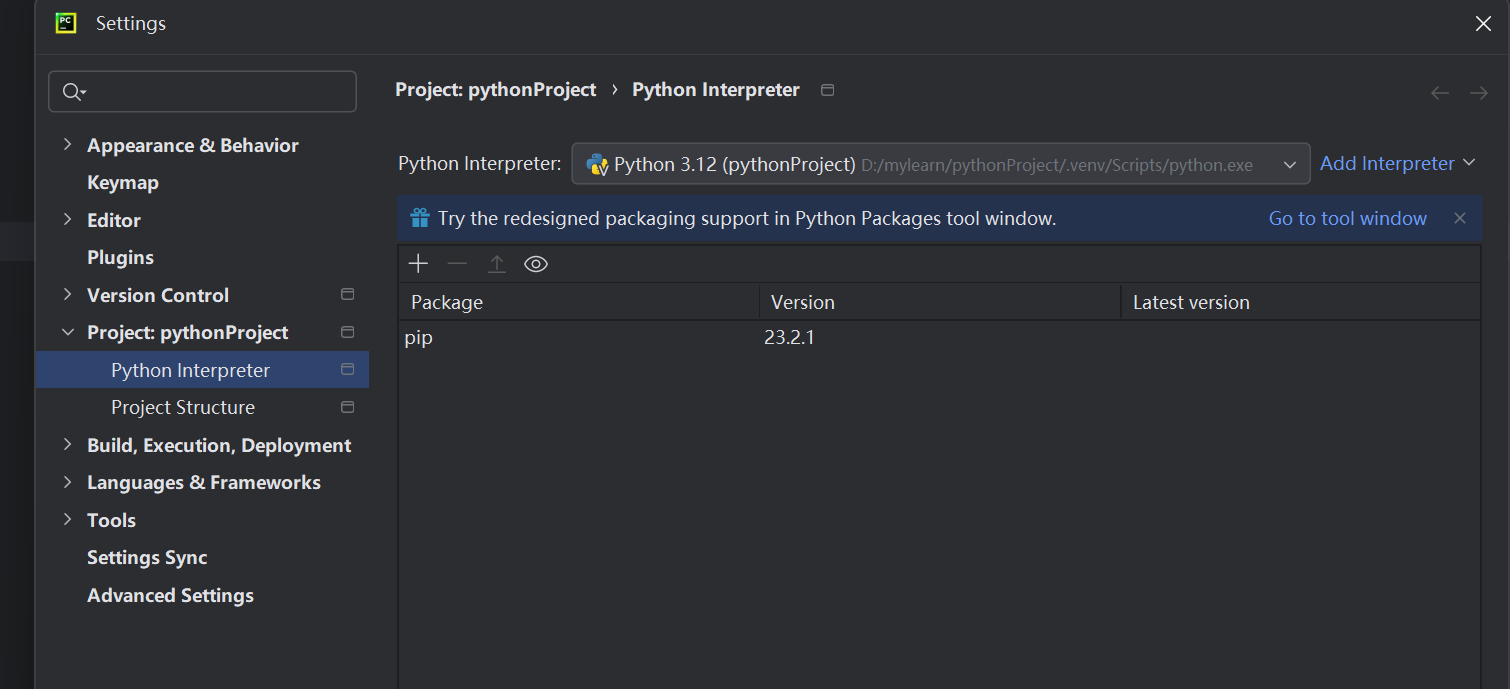
描述已自动生成

（2）下载第三方包：Python中自带pip，可以使用cmd来执行pip命令，进行下载

文本

描述已自动生成

（3）PyCharm中：在其中也可以下载第三方包



网络不好时：可以使用镜像

文本

描述已自动生成

22、Python基础综合------折线图可视化

（1）技术支持：Echarts：百度开源的数据可视化工具

（2）JSON数据格式：一种数据交互格式，可以按照JSON指定的格式去组织和封装数据。本质是一个带有特定格式的字符串

常用方法：dumps（数据）：将数据转化为JSON格式

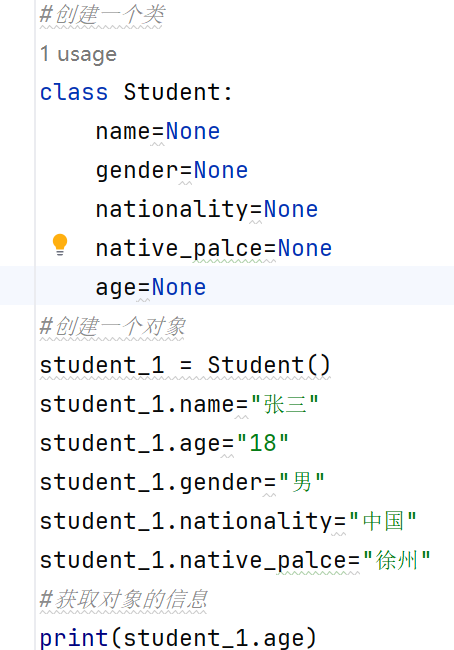
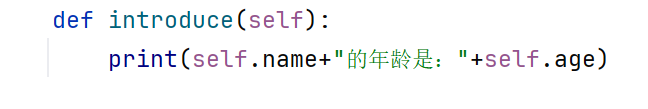
Loads（）：将JSON数据转化为一般数据

（3）配置选项：第一种：全局配置

第二种：系列配置

（4）全局配置：可以使用set\_global\_opts方法进行实现

（5）数据处理：

1. 面相对象
2. 初始对象：
3. 成员方法：

这里的self表示的是类对象本身

1. 构造方法：使用\_ \_init\_ \_（self，其它属性）

图形用户界面, 文本, 应用程序

描述已自动生成

1. 其它内置方法（魔术方法）：

图示

描述已自动生成

1. 字符串方法：相当于Java中的toString方法

图片包含 图形用户界面

描述已自动生成

1. 大小比较方法：相当于Java中的compartor（）方法

图片包含 文本

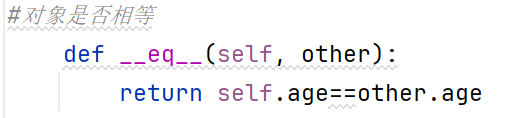
描述已自动生成

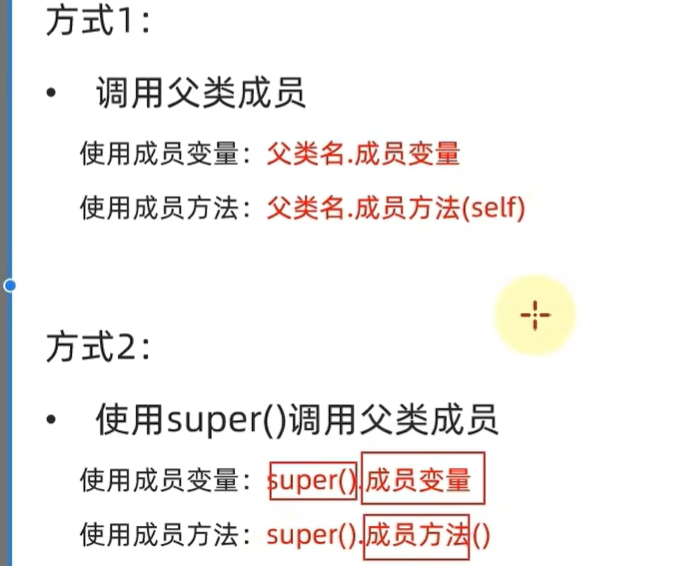
1. 大于（小于）等于：可以进行大于等于和小于等于的比较

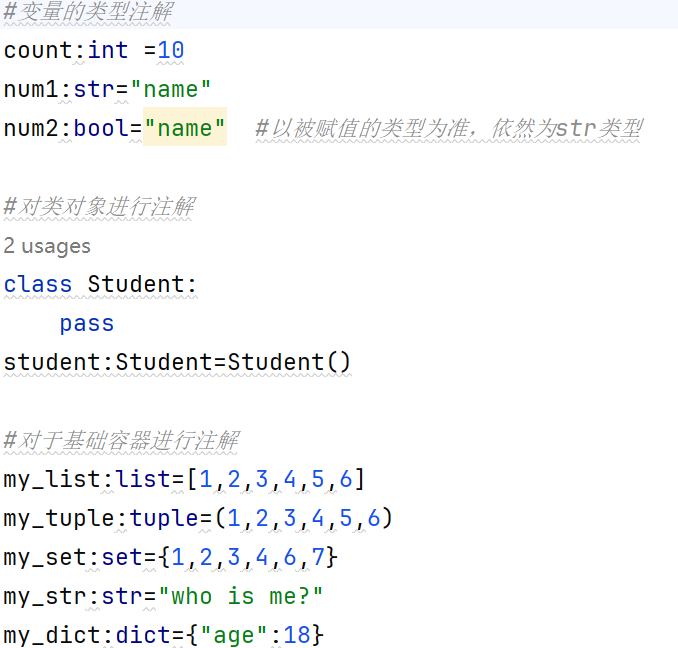
注意：这里在对象中也只能使用>=或<=进行比较，不能混用<或>

文本

描述已自动生成

1. 等于的判断：
2. 封装性
3. 私有成员：在私有成员和私有方法前加上\_ \_，即可变为私有。成员方法可以访问私有
4. 继承
5. 单继承：语法：class 类名（父类名）
6. 多继承：语法：class 类名（父类1，父类2，父类3）
7. pass关键字：补全代码的作用，当子类继承父类后，不想要再进行更多的添加，便可以使用pass进行跳过，在if语句、for循环中都可以使用
8. 优先级：在多继承中，括号内从左往右，优先级依次递减，当出现同名方法、变量名时，会优先继承优先级高的
9. 特殊情况：当重写了父类的成员方法时，如果想调用父类的，有两种方式



1. 类型注解：
2. 作用：在python中，编译器是无法自动识别参数的类型，因此需要使用类型注解，帮助进行识别
3. 支持：变量的类型注解、函数（方法）形参列表和返回值的类型注解
4. 变量的类型注解：

文本, 信件

描述已自动生成

1. 函数的类型注解：文本, 信件

   描述已自动生成
2. Union类型：适用情况：当出现多种类型的多个变量，不好一一标注的时候可以使用

图形用户界面, 文本, 应用程序

描述已自动生成

1. 多态

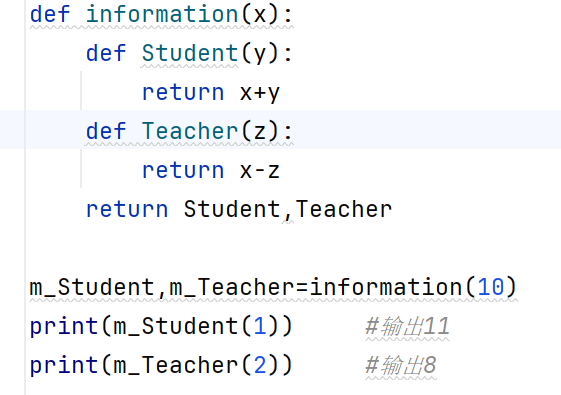
图形用户界面, 应用程序

中度可信度描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

Python高级技巧

1. 全局变量：在Python中，在函数和类外的变量称为全局变量，可以在整个模块中被访问和修改。但是在函数和类中如果要对于全局变量进行修改，要在函数内部声明变量是全局变量。否则，相当于在函数体内部创建了一个同名的局部变量
2. 闭包
3. 作用：有些全局变量，仅仅是为了一个函数设置的，此时，由于是全局变量，存在被其他函数和类修改的风险，因此，闭包可以很好的解决这个问题，在类中也可以实现闭包的功能
4. 解析：在嵌套函数中，内函数调用外函数中的变量，而外函数返回内函数，其中内函数可以存在多个
5. 注意：当闭包中存在多个内函数的时候，在调用的时候要创建多个函数去一一对应，否则不能够实现闭包，因为，编译器无法识别到底返回的是哪一个内函数。
6. 代码演示：
7. 缺点：会导致一步分的内存空间不能够被释放，一直占用内存
8. Nonlocal：在内函数中修改外函数的变量值
9. 装饰器
10. 作用：在不破坏函数原有代码和功能的前提下，为函数增加新的功能
11. 本质：闭包的一种，只是将外部函数的变量改为要进行功能增加的函数
12. 错误演示：

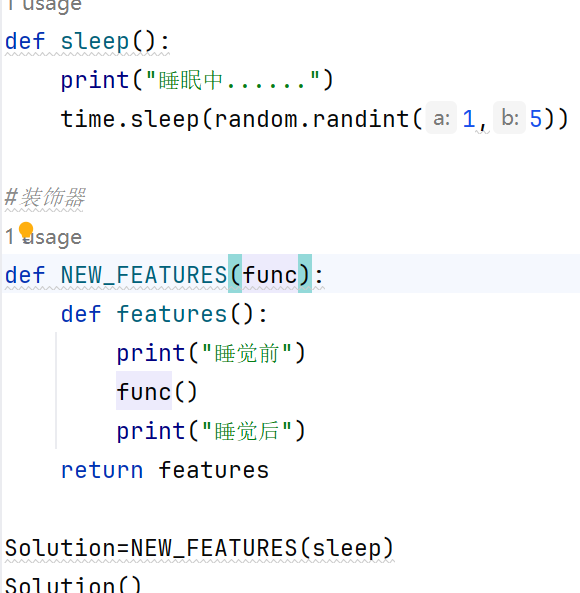
图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

1. 正确代码：

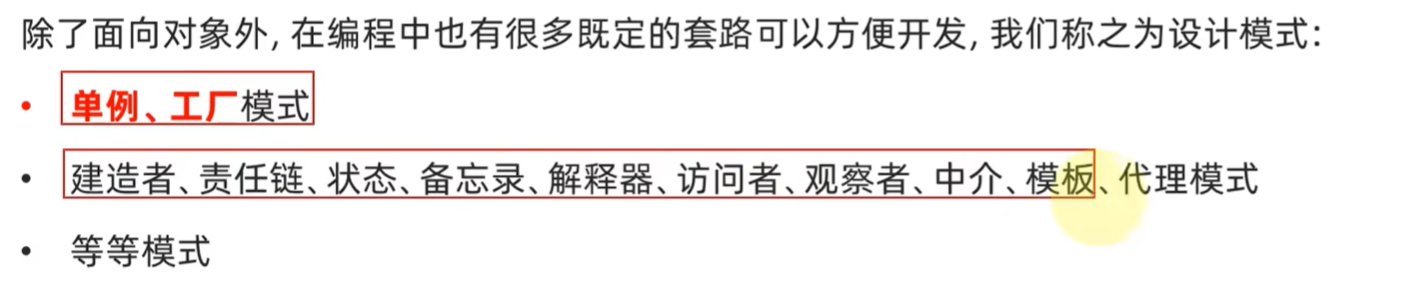


1. 注意：在Python中，函数的参数列表是无法指定声明参数到底是哪一个函数，但是如果是类的话则是可以的



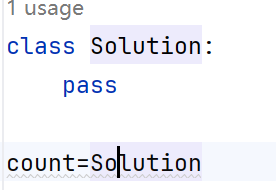
1. 原因：Python不同于Java、C++等静态语言类型，是一种动态语言类型

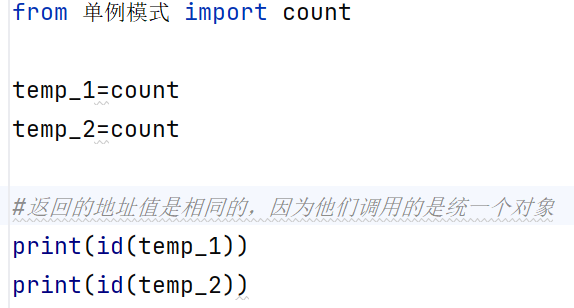
动态语言和静态语言：简单来说，就是变量的类型是否是可变的，例如，在Java中定义一个变量，int num=1，那么num的类型就是已经确定下来了，之后对其修改，也只能更改为整型，或浮点型等，而在Python中，a=1，a在之后可以更改为字符串型，布尔型都是可以的

1. 简便写法：
2. 设计模式
3. 概念：相当于一种套路，可以极大的方便代码的编写，最简单的设计模式就是面向对象
4. 常见的模式：
5. 单例模式：定义：保证一个类只有一个实例，并提供一个访问它的全局访问点

作用：可以节省内存的开销

常见的应用场景：在一个工具类文件中创建一个类，并创建对应的唯一对象，在另一个模块中导入该对象，便实现了单例模式。类似于Java中的Scanner类

代码展示：



1. 工厂模式：定义：当需要大量构建一个类的类对象的时候

优点（作用）：大量的创建对象，有统一的入口，方便代码的维护，当需要大量的进行修改的时候，只需要对于工厂进行修改即可

1. 多线程：

代码实现：



1. 网络编程
2. Socket：定义：负责进程之间的网络数据传输
3. 分类：服务端和客户端
4. 服务端的编程：

图形用户界面, 文本, 应用程序, 聊天或短信

描述已自动生成

1. 客户端的开发：图形用户界面, 文本, 应用程序, 聊天或短信

   描述已自动生成
2. 正则表达式：
3. 在Python中，使用re模块，并基于re模块的三个基础方法来做正则匹配
4. 基本方法：match、search、findall
5. match用法: re.match(匹配规则，被匹配的字符串)