python文件的拓展名为.py

输出：print() 函数

括号中可以直接写字符串 print("hello world!")

也可以写变量名 message="hello world!"

print(massage)

变量的定义和使用：

只能包含字母、数字、下划线，且只能以字母或下划线开头

变量名不能包含空格，但可以以下划线分隔单词

不能使用关键字和函数名做变量名

（与C++原则相同）

报错：“名称错误” 一般是变量名没有赋值，或者输入的变量名拼写不正确。

注意python中没有类似C++中的变量声明（int n），声明一个变量必然对其赋值

字符串：

在python中，用引号引起来的都是字符串，引号可以是单引号，也可以是双引号（可以在字符串中包含引号和撇号）

字符串操作函数：



title()称为方法；圆点（.）让Python对变量st执行方法title()指定的操作；方法通常需要额外的信息来完成工作。这种信息是在圆括号内提供的，title()不需要额外的信息，所以括号是空的。



这两个函数通常用于：很多时候无法通过用户提供正确的大小写，因此可以先将其全部转化为大写或小写，之后再根据需求转化为合适的大小写方式。

在字符串中使用变量：

就是用已有的字符串给新的字符串赋值，可以在前引号前加上f（是是设置格式format的简写），再将要插入的字符串放在花括号内。

这样的字符串就称为f字符串。(可以类比格式化输入输出)

例如：



f字符串是3.6的版本引入的，如果是更早的版本，需要使用方法format()，在圆括号内列出字符串时用的变量。每一个变量都通过一对花括号来引用，然后按顺序将花括号的值替换为变量的值。

例如：



制表符、换行符：

\t \n

Python强调缩进格式

删除空白：

Python能够发现字符串中额外的空白，并认为它意义重大——除非你告诉它不是这样。

Python能够找出字符串开头和末尾的空格。

方法rstrip()可以去除字符串末尾的空白（但这种删除只是暂时的，接下来再次询问这个字符串的值的时候仍然会有末尾的空格）。

例如：



如果想将字符串末的空格完全删除，需要将字符串变量重新赋值。

例如：



此外还可以剔除字符串开头的空白，需要使用的方法为lstrip()

也可以同时剔除左右两边的空白，需要使用的方法为strip()

使用字符串时可能会出现的语法错误:

在用单引号括起的字符串中，如果字符串中仍包含撇号，就会报错。

这是因为Python会将第一个单引号和撇号之间的内容视为一个字符串，将余下的文本视为python代码而报错。

数

整数：

在python中，可以对整数直接执行+、-、\*、/和乘方(用两个乘号\*\*表示)的运算。

python支持运算次序，在一个表达式中执行多种运算，并依据优先级计算结果；当然也可以用括号修改运算顺序。

空格不影响计算表达式的方式。

浮点数：

计算机计算得到的结果包含的小数位数可能是不确定的（精度问题）。

暂时忽略多余的小数位数即可（在具体实现项目的学习中，会根据需要学习处理多余的小数位数）

整数和浮点数：

任意两个数相除时，得到的结果总是浮点数，即使这两个数都是整数。

在其他运算中，如果有操作数是浮点数，其他无论是整数或是浮点数，得到的结果总是浮点数。

数中的下划线：

书写很大的数字时，可以用下划线将数字分组，使其更清晰易读。

例如：

age=14\_000\_000\_000

打印这种使用下划线定义的数时，python不会打印其中的下划线。

例如：

print(age) //得到的结果为14000000000

这种表示法适用于整数和浮点数，但只有3.6及以上的版本支持。

常量：

Python内没有内置的常量类型。但可以使用全大写来指出某个变量应为常量；

例如：

NUM=15；

注释：

注释是非常重要的一个部分，在Python中，用#来标识注释。

列表：

列表由一系列按特定顺序排列的元素组成。相当于数组。元素之间可以没有任何关系。

python中用方括号[]表示列表，并用逗号分隔其中的元素。

例如：



访问列表元素：

要访问列表元素，可指出列表的名称，再指出元素的索引。

例如：



当然，可以对列表中的字符串元素使用方法，从而得到正确的格式。

特别注意，与C++中相同，列表的索引是从0开始而非从1开始。

python为访问列表最后一个元素提供了特殊的语法。通过将索引指定为-1，可以让Python返回最后一个列表元素。

这种约定同样也适用于其他索引，例如-2返回倒数第二个，-3返回倒数第三个，并依次类推。

修改、添加和删除元素：

要修改列表元素，可指定列表名和要修改的元素的索引，再指定该元素的新值。

（与C++类似）

例如：



在列表中添加元素：

在行列末尾添加元素：使用方法append()

例如：



在列表中插入元素：

使用函数insert()可以在列表的任何位置插入元素。为此，需要指定新元素的索引和值。

例如：



在列表中删除一个元素：

要删除列表中的元素，可以使用del语句。使用del可以删除列表中的任意一个元素，前提是知道元素的索引。

例如：

 使用del将元素从列表中删除后，就再也无法访问它了。

使用方法pop()删除元素：

有时候需要将元素从列表中删除并接着使用它的值，需要使用方法pop().

方法pop()删除列表末尾的元素，并让你能够继续使用它。

列表相当于一个栈，而删除栈顶元素相当于弹出栈顶元素。

例如：



实际上可以使用pop()删除列表中的任意元素，只需要在圆括号内指定要删除元素的索引即可。

要注意使用pop后，被弹出的元素就不在列表中了。

根据值删除元素：

如果不知道需要删除的元素的位置，仅知道要删除元素的值，可以使用方法remove().

例如：



注意方法remove()只删除第一个指定的值。如果要删除的元素在列表中出现多次，就需要使用循环来确保每个值都被删除。

另外如果指定删除的值在列表中不存在，编辑器会直接报错。

对列表中元素进行排列：

使用方法sort() 对列表进行永久排序：

sort()可以使列表中的元素按升序排序（如果括号内不加参量）

例如列表lists，对其进行sort排序的指令为：

lists.sort()

与C++类似，想要使sort按照降序排序，只需要向sort()方法传递参数 reverse=true即可。

sort()对数组元素的排列是永久性的。

使用函数sorted()对列表进行临时排序：

函数sorted()能够使列表中的元素按特定顺序排列，同时不影响它们在原始列表中的排列顺序。

同样，如果想要降序排序，像函数传递参数reverse=true

注意，使用上述两种排序方法时，列表中的元素最好是“同类型”。比如列表中都是小写；当列表中的值含有大写时，决定排列顺序有多种对大写字母的解读方式，要指定排列顺序可能要复杂的多。

倒着打印列表：

要反转列表中元素的排列顺序，可以使用方法reverse() .

方法reverse()永久性的修改列表中元素的顺序，但是要回到原来的顺序，只需要再次调用reverse()即可。



对列表和循环时使用的列表元素变量进行适当的命名有助于理解代码操作。

for循环中可以包含多条语句，执行多个操作。

python对缩进的要求十分严格，在一个for下执行的语句应该有相同的缩进。

for循环之后执行的语句应该与for循环有相同的缩进。

缩进可能在语法和逻辑上发生错误：

语法上发生错误可能是灭有找到应该被缩进后执行的语句，例如for循环后没有语句缩进。或者缩进了不该缩进的语句，例如仅有顺序结构时，将某些语句缩进。

逻辑上发生错误可能是缩进了不该缩进的语句，导致这个语句在函数中执行发生错误。或者应该在函数中执行的语句没有缩进。

语法错误编辑器会直接报错，逻辑错误可能一找一个小时（哭哭）

以及不要忘记冒号。

创建数值列表：

即创建一个用于存储数字集合的列表。

 

使用range()函数几乎可以创建所有的列表元素，例如将每个数的平方加入数集。

例如：

//在python中，用两个星号\*\*表示乘方运算

 //或者为了使代码更简洁，可以不使用中间变量square

使用**列表解析**创建列表：

列表解析将for循环和创建新元素的代码合成一行，并自动附加新元素。

 这种语法的格式为：首先指定一个描述性的列表名；然后，指定一个左方括号，并定义一个表达式，用于生成要存储到列表中的值；接下来，用for循环给表达式提供值，再加上右方括号。

对数字列表执行简单的统计操作：

min() //得到列表的最小值

max() //得到列表的最大值

sum() //对数字列表求和

使用列表的一部分内容，**切片**



//也就是说第一个参数是起始索引，第二个参数是结束的索引的下一位，第三个参数（可有可无）是步长

//实际运行可知索引超界不会报错，输出到实际结束的地方

//第三个参数num，就是索引每次加num