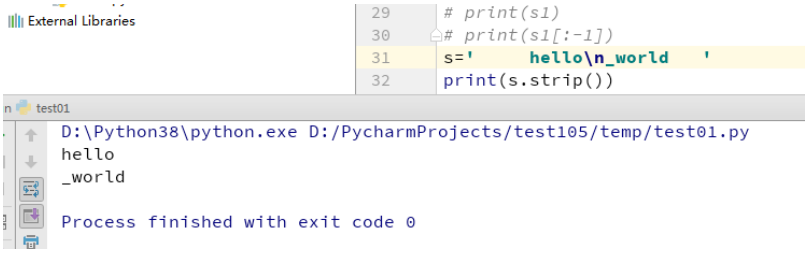
python1基础

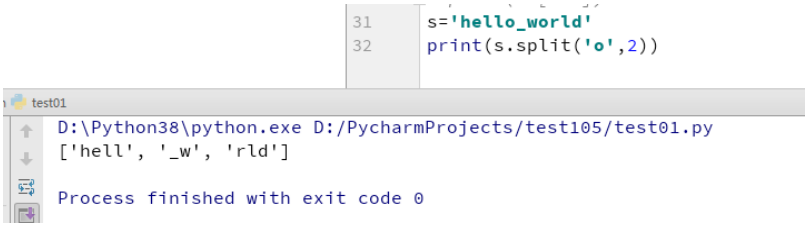
- 读代码不是读英文，而是读格式
- 一串字符加括号
  - 实例化一个类
  - 调用函数
- 标识符：给数据命名的符号
- 标识符的命名规则
  - *不要与系统关键字、类、库、函数等重名
  - 不要以数字开头或纯数字，可以是字符、下划线和数字的组合，例如：test_01
  - 不要以双下划线开头并结尾，例如：__abc__，避免系统歧义
  - 命名要有意义
  - 可以驼峰式或蛇形来命名
    - 驼峰式
      - 大驼峰：组成标识符的英文首字母都大写，例如：HelloWorld
      - 小驼峰：除了首个单词外，组成标识符的英文首字母都大写，例如：helloWorldPython
    - 蛇形：通过下划线拼接字符，例如：Hello_world、hello_world
-
- python常用的运算符
  - +、-、*、/
  - 特殊运算符
    - %：取余或取模，两数相除取余数
    - //：取商或取整，两数相除取商数
    - **：取幂或取指，以a ** b
      - 如果b为正数：相当于a的b次方
      - 如果b是以1为分子的分数：相当于a开b次方
      - 如果b是负数：先获取a的b次方，然后取倒数
-
- python常用的比较符：会得到True/False
  - <、>、<=、>=、!=
  - 特殊的比较符：==
-
- 赋值符：将赋值符右边的逻辑结果，赋予左边的变量的过程

- 变量：用于接受逻辑结果的标识符
  - ***变量有值才有意义
- *赋值的逻辑是先算右边，再给左边变量
  - 常见的赋值符：=
  - 特殊的赋值符：+=、-=、*=、/=、%=、//=、**=
    - 以 i += 1为例，等价于i = i +1
-
- python中的逻辑符
  - and、or、not
  - and连接的条件两边同时成立才成立
  - or连接的条件只需一边成立整体就成立
  - not修饰的条件，原命题成立反而会失败
-
- 判断成员是否在集合内：in、not in
-
- 代码的注释：被注释的代码不会被执行
  - 单行注释：代码行前加# 号
  - 多行注释：通过三引号将待注释代码引起来，例如："""待注释代码"""、'''待注释代码'''
  - 快捷方式：Ctrl+/
-
- python中的基础数据类型
  - 整数型 int、浮点型 float、字符串 str、布尔值 bool、列表 list、元祖 tuple、字典 dict
-
- 常用的系统函数和类
  - type()：判断数据的数据类型
  - len()：length的缩写，获取集合的长度（元素个数）
  - max()/min()：获取集合中的最大/最小值
  - input()：接受键盘的输入，返回字符串。input函数会形成阻塞
-
- 函数与类方法的区别
  - 函数是独立存在的逻辑功能语句组
  - 类方法是依附于某个数据类型的函数
    - 调用类方法的格式：标识符.标识符()
    - 调用类方法的前提：必须先满足是某种数据类型
- s='hello world'

- print(type(s))
- 字符串str：字符的有序集合
  - 标识：引号，单双都可以，python对引号不敏感
  - 有序：可以通过下标索引获取集合中的元素
  - 下标索引：集合中的元素，左起第一个对应下标为0，第二个对应为1，以此类推。右起为-1
  - *取下标的格式：标识符[整数]
-
- 字符串的常用方法
  - find()/index()：根据指定字符串在目标字符串中匹配，如果能匹配到，则返回第一次出现的下标索引，如果匹配不到，find返回-1，index报错
  - replace(old,new)：将字符串中old部分替换成new部分
  - strip()：去除字符串首尾指定部分，默认去除空格和换行符
    - 转义：在字符串中，转义符和特殊的字符组合，会有其他的意义
      - 转义符：\
      - 常用的转义：
        - \n：换行符
        - \t：制表符，4-8个空格



  - **split(flag,num)：根据指定字符串，切割目标字符串



    - flag：切割的标记
    - num：左起切多少次
  - count(element)：统计指定元素在目标字符串中出现的次数
  - 大小写相关
    - 判断组成字符串的字母是否都是大/小写：isupper()/islower()
    - 将字符串中的字母转换成大/小写：upper()/lower()
    - 将字符串的首个字母转换成大写：capitalize()
    - 将组成字符串的英文单词的首字母都大写：title()

- 判断字符串是否以指定字符开头/结尾：startswith()/endswith()
- 判断字符串是否由纯字母/纯数字组成：isalpha()/isdigit()
- 根据指定字符串，拼接有序集合：join()

```
27  s='hello_world'
28  s1='|'.join(s)
29  print(s1)
```

```
Run  test01
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py
h|e|l|l|o|_|w|o|r|l|d

Process finished with exit code 0
```

-
- 字符串的扩展知识
  - 字符串的二则运输：字符串可以和字符串相加，也可以和整数相乘
  - 字符串的切片
    - 切片的格式：标识符[x:y:z]
    - x、y、z分别代表起始值、终止值（不含）和步长。各种默认值为0、字符串的长度、1
    - 如果只有一个冒号
      - 标识符[num:]：从下标索引为num的字符开始切字符串，直到字符串的结尾。相当于去除字符串的前num个元素
      - 标识符[:num]：从首个字符开始，取字符串前num个字符
      - 标识符[:-num]：等价于标识符[:len(标识符)-num]，也等价于从右往左数num个元素。相当于去除字符串后num个元素
      - **特殊格式：标识符[::-1]，镜像输出字符串
-
  - 列表 list：数据的有序集合，元素之间通过逗号分开
  - 标识：[]
  - l=[1,2,4,1,4,5,1,2]
  - 列表的常用方法
    - 增加元素
      - **append()：在列表的末尾处追加一个元素
      - extend()：在列表的末尾处追加一个有序集合，并将集合中的元素一一展开，批量添加到列表中
      - insert(index,value)：在列表的指定位置插入数据
    - 删除元素
      - **pop(index)：根据下标索引删除元素。默认删除最后一个元素
      - remove(element)：根据指定元素在目标列表中匹配，如果能匹配到则删除第一次出现的元素
      - 通过系统关键字del删除元素：del 标识符[整数]

        - 修改元素
            - 重新给元素赋值：标识符[整数]=新的值
        - 查询元素
            - *index():根据指定元素在目标列表中匹配，如果能匹配到则返回该元素第一次出现的下标索引
            - count(element)：统计指定元素在目标列表中出现的次数
        - copy()/clear()：复制/清空列表
            - 深拷贝和浅拷贝
                - 深拷贝是在内存中新申请资源来存储值；浅拷贝只是通过新增指针的方式，类似于快捷方式
- 列表的扩展知识
    - 列表的二则运算：列表可以和列表相加，也可以和整数相乘
    - 列表的嵌套：列表中的元素，也是一个列表或其他集合
    - 列表也支持切片操作，同字符串
    - 纯数据列表：列表中的元素都是数值型或字母
        - 纯列表支持sort()排序
        - 支持max()/min()取最大/最小值
        - 支持sum()求和
    - 镜像输出列表：reverse()
-
- 元祖 tuple：数据的有序集合，元素之间通过逗号分开，元祖一旦被定义，则元素不能被修改
- 标识：()
- *如果元祖中只有一个元素，需要加一个逗号与四则运算区分歧义
- t=(1,[1,2,3])
- 元祖的常用方法
    - index()：根据指定元素在目标元祖中匹配，如果能匹配到则返回该元素第一次出现的下标索引
    - count(element)：统计指定元素在目标元祖中出现的次数
- **元祖与列表的区别
    - 列表支持元素的修改；元祖不支持
    - **列表在内存中是一种不稳定、不定长的状态；元祖则相对稳定
    - *列表通常用于存储少量数据；元祖通常用于函数间的参数传递
- 元祖的扩展知识
    - 定义元祖时可以不用加()
    - 多变量从元祖/列表中取值

    - 元祖可以嵌套到其他集合，但不负责其他集合的子元素不被修改
-
- 布尔值bool：判断逻辑结果是否为真的依据
- 常见的布尔值：True/False
- 特殊的布尔值：存在即合理
    - 合理的值：非空集合、非零的数
    - 不合理的值：空集合、零、None
-
- 字典dict：键值对的无序集合，元素之间通过逗号分开，每个元素都有一个冒号，冒号左边为键，右边为值
- 无序：不能通过下标索引取值
- 标识：{}
- 键值对：成对出现的数据，类似于a=1、b=2，在字典中表示为a:1、b:2
- 字典中的元素通过键取值，字段中键不能重复
    - 字典通过键取值的格式：标识符[字符串]
- d={**'name'**:**'张三'**,**'age'**:18,**'hobby'**:**'钓鱼'**,**'info'**:{**'a'**:1,**'b'**:2,**'c'**:[1,2,3,4,5]}}
- 字典的常用方法
    - 增加元素
        - update(dict)：相当于融合两个字典的元素
        - 新增键值对：标识符[新的键]=值
    - 删除元素
        - pop(key)：根据键删除指定元素
        - 通过del关键字删除元素：del 标识符[新的键]
    - 修改元素
        - 给键重新赋值：标识符[键]=新的值
    - 查询元素
        - key()/values():获取字典的所有键/值
        - 可以双变量for循环迭代items()

    - copy()/clear()：同列表

- **字典与json的区别
  - json：严格的键值对数据模型，通常在各语言、各模块中充当信使参数
  - 字典可以理解成一个不严谨的json
-
- 字典的扩展知识
  - 字典的嵌套：字典中元素的值可以是一个字典或其他集合
  - set集合：可以将set集合理解成一个没有值的字典
    - set集合的标识也是{}
-
- 数据类型的强制转换：在处理不同类型数据时，通常需要将 数据转换成同一类型进行统一处理
- *需要哪一种数据类型，使用该数据类型加括号来转换其他数据类型
  - 例如：int()、float()、str()、list()、tuple()、set()
- 数据类型强制转换的规则
  - 所有的数据类型都可以转换成字符串，并不是所有的字符串都能转换成其他数据类型
  - 整数可以无损转换成浮点数，但浮点数转换成整数时会丢失小数点后的数据
  - 列表和元祖可以相互无损转换
  - 字典由于结构特殊，通常不参与数据类型的强制转换。字典可以转换成列表但会丢失值
  - 布尔值只作为判断逻辑结果的依据，不参与数据类型的强制转换
-
- 字符串的格式输出：按照一定的制式标准，输出字符串
- 占位符：可以在字符串中通过占位符来暂时占位，等待后续数据的传入
  - 常用的占位符
    - %s:s是string的缩写，即将传入的数据是一个字符串
    - %d:d是digit的缩写，即将传入的数据是一个整数型
    - %f:f是float的缩写，即将传入的数据是一个浮点数
      - 可以通过%.xf的方式来精确小数点后的位数。原则上是四舍五入，但实际会因为浮点数精度问题出现误差
        - x必须是正整数
- 通过元祖实现字符串的格式化输出
- print('我来自%s,名叫%s,今年%d岁,身高%.100f厘米' %('广州','李四',18,174,35))
- 通过字典实现格式化输出
- d={**'city'**:**'上海'**,**'name'**:**'老刘'**,**'age'**:49,**'hight'**:178.36,**'hehe'**:0}
- print(**'我来自%(city)s,名叫%(name)s,今年%(age)d岁,身高%(hight).1f厘米'** %d)

```python
d={'city':'上海','name':'老刘','age':49,'hight':178.36,'hehe':0}
print('我来自%(city)s,名叫%(name)s,今年%(age)d岁,身高%(hight).1f厘米' %d)
```