- Python笔记
  - 赋值运算符
  - Python的输入和输出
    - 字符串格式化
    - %表示占位符，后面跟的字母表示类型，如%s表示字符串类型
    - 所以我们可以将第一段代码优化为：
    - 换行
    - 输入
  - 选择流程
    - 单分支流程
    - 双分支流程
    - 多分支流程
      - 猜拳机小demo
      - if-else的嵌套使用
  - 循环流程
    - while循环
      - 输出1-100之间的数据
    - While的嵌套
      - 打印九九乘法表（正三角）
      - 打印倒三角
      - 打印等腰三角形
  - For循环
    - Range
    - break与continue
    - 用for打印乘法表
    - for---else
      - 输入密码限制次数
    - while--else
  - day2作业
      - 猜年龄小游戏，有三点要求
      - 计算IBM数值
  - 字符串操作
    - 获取第n个字符
    - 遍历字符串
    - 首字母变大写
    - 去除空格(左，右)
    - 复制字符串
    - 查找数据是否存在
    - 以...开头或者以...结尾
    - 大小写转换
    - 切片截取
  - 列表操作
    - 列表的定义
    - 列表追加元素
    - 列表插入
    - 批量增加
    - 修改
    - 删除
    - 查找
  - 元组
    - 元组的创建和遍历
    - 元组中的列表可以修改
    - 统计字符出现次数

Python笔记

赋值运算符

赋值运算符	作用描述	结果描述
=	赋值运算符	将右边的值赋给左边
+=	加法赋值运算符	c+=a等价于c=c+a
-=	减法赋值运算符	c-=a等价于c=c-a
*=	乘法赋值运算符	c*=a等价于c=c*a
/=	除法赋值运算符	c/=a等价于c=c/a
%+	取余赋值运算符	c%=a等价于c=c%a
**=	幂赋值运算符	c**=a等价于c=**a
//=	取整赋值运算符	c//=a等价于c=c//a

Python的输入和输出

字符串格式化

```
print('我是小A,我来自一班')
print('我是小B,我来自二班')
print('我是小C,我来自三班')
```

将以上代码优化，用到了字符串格式化

%表示占位符，后面跟的字母表示类型，如%s表示字符串类型

```
    name='A'
    class='海南大学'
    age=11
    print('我的名字是%s：来自【%s】，今年%d岁'%(name,class,age))


  '''
  输出结果：我的名字是A：来自【海南大学】，今年11岁
  '''
```

所以我们可以将第一段代码优化为：

```
name1,name2,name3='小A','小B','小C'
CLASS1,CLASS2,CLASS3='一班','二班','三班'
print('我是%s,我来自%s'%(name1,CLASS1))
print('我是%s,我来自%s'%(name2,CLASS2))
print('我是%s,我来自%s'%(name3,CLASS3))
```

换行

```
print('我可以\n换行吗')

'''
我可以
换行吗
'''
```

- 格式化符号

格式符号	转换
%c	字符
%s	通过str()字符串转换来格式化
%i	有符号的十进制整数
%d	十进制整数
%u	无符号的十进制整数
%o	八进制整数
%x	十六进制整数
%e	索引符号（小写e）
%E	索引符号（大写E）
%f	浮点实数
%g	%f和%e的简写
%G	%f和%E的简写

练习输出

```
name='老夫子'
QQ=666666
phone=15879209585
address='广东东莞'
print('姓名:%s\nQQ:%d\n手机号:%d\n地址:%d'%(name,QQ,phone,address))

#输出结果:
'''
姓名:老夫子
QQ:666666
手机号:15879209585
地址:广东东莞
'''
```

还可以这样输出

```
print("姓名: ",name)
```

还可以使用format,{}是format的占位符

```
>>>"{} {}".format("hello", "world")    # 不设置指定位置，按默认顺序
'hello world'

>>> "{0} {1}".format("hello", "world")  # 设置指定位置
'hello world'

>>> "{1} {0} {1}".format("hello", "world")  # 设置指定位置
```

```
'world hello world'



rint("网站名：{name}, 地址 {url}".format(name="菜鸟教程",
url="www.runoob.com"))

# 通过字典设置参数
site = {"name": "菜鸟教程", "url": "www.runoob.com"}
print("网站名：{name}, 地址 {url}".format(**site))

# 通过列表索引设置参数
my_list = ['菜鸟教程', 'www.runoob.com']
print("网站名：{0[0]}, 地址 {0[1]}".format(my_list))  # "0" 是必须的
```

输入

`Input`函数：

```
input('请输入你的名字')
#接下来就输入名字就好了，类似于C语言中的scanf
```

选择流程

单分支流程

if 条件表达式：

    代码指令

```
score=int(input('请输入你的成绩'))
if score>60:
 print('你及格了')
 pass                    #空语句
print('结束')
```

ps：要加int因为input默认是字符串类型

双分支流程

if 条件表达式：

    代码指令

else：

代码指令

```
score=int(input('WHAT IS YOUR SOCRE?'))
if score>60:
print('congratulation!')
else:
print('come on!see you next time')
```

多分支流程

if 条件:

　代码指令

elif 条件:

　代码指令

elif 条件:

　代码指令

else:

　代码指令

```
score=int(input('请输入你的成绩'))
if score>90:
 print('你的成绩为A等')
elif score>80:
 print('你的成绩是B等')
elif score>=60:
 print('你的成绩是C等')
elif 0<=score<60:
 print('你的成绩是D等，不合格，加油！')
else:
 print('输入不规范')
print('程序结束')
```

猜拳机小demo

```
import random
print('---------欢迎来到猜拳机器---------')
choice=input('请输入你的选择，剪刀还是石头还是布？')
choice2=random.choice(['剪刀','石头','布'])
if choice=='剪刀'and choice2=='布'or choice=='石头'and choice2=='剪刀'or
choice=='布'and choice2=='石头':
    print('YOU WIN!')
    a=0
elif choice=='剪刀'and choice2=='石头'or choice=='石头'and choice2=='布'or
choice=='布'and choice2=='剪刀':
    print('YOU LOSE!')
    a=1
else:
    print('平局！')
    a=2
if a==0:
```

```
    print('不愧是你！')
elif a==1:
    print('LOW!')
elif a==2:
    print('嗨，再来一把！')
else:
    pass
```

其中random函数是产生随机数的，具体用法如下：

```
import random

# 随机整数：
print random.randint(1,50)

# 随机选取0到100间的偶数：
print random.randrange(0, 101, 2)

# 随机浮点数：
print random.random()
print random.uniform(1, 10)

# 随机字符：
print random.choice('abcdefghijklmnopqrstuvwxyz!@#$%^&*()')

# 多个字符中生成指定数量的随机字符：
print random.sample('zyxwvutsrqponmlkjihgfedcba',5)

# 从a-zA-Z0-9生成指定数量的随机字符：
ran_str = ''.join(random.sample(string.ascii_letters + string.digits, 8))
print ran_str

# 多个字符中选取指定数量的字符组成新字符串：
print ''.join(random.sample(['z','y','x','w','v','u','t','s','r','q','p','o','n','m','l','k','j','i','h','g','f','e','d','c','b','a'], 5))

# 随机选取字符串：
print random.choice(['剪刀', '石头', '布'])

# 打乱排序
items = [1, 2, 3, 4, 5, 6, 7, 8, 9, 0]
print random.shuffle(items)
```

if-else的嵌套使用

话不多说直接上例子：

```
love=input('DO YOU LOVE ANIME?[YES,NO]')
level=input('your love to anime[A,B,C]')
if love=='YES':
    if level=='A':
```

```
        print('YES YOU ARE FUCXING LOVER')
    elif level=='B':
        print('OH YOU ARE TRUE LOVER')
    elif level=='C':
        print('YOU ARE LOVER')
    else:
        print('invade!!!!!!!')
elif love=='NO':
    print('FUCX YOU!')
else:
    print('INVADE1!!!!!!')
```

循环流程

while循环

while 条件表达式:

    代码指令

语法特点:

有初始值

条件表达式

自增或者自减

输出1-100之间的数据

```
i=1
while i<100:
    i+=1      #python中没有i++和i--
    print(i)
```

While的嵌套

打印九九乘法表（正三角）

```
i=1
while i<=9:
    j=1
    while j<=i:
        print('{}*{}={}'.format(i,j,i*j),end='  ')
        j+=1
    print('\n')
    i+=1
```

print默认是打印一行，结尾加换行。end=' '意思是末尾不换行，加空格。

打印倒三角

```
i=9
while i>=1:
    j=1
    while j<=i:
        print('{}*{}={}'.format(i,j,i*j),end='  ')
        j+=1
    print('\n')
    i-=1
```

打印等腰三角形

```
i=1
j=4
while i<=4:
    print(' '*(j-i)+(2*i-1)*'*',end='')
    print()   #没有内容默认就是换行
    i+=1
```

For循环

for 临时变量 in 容器:

代码

```
tags='我是一个中国人'  #字符串本来就是一个字符类型集合
for item in tags:
    print(item)

'''
我
是
一
个
中
国
人
'''
```

这里这个item已经被定义了，后续也可以用

Range

`range`：此函数可以生成一个数据集合列表

`range(起始值，结束，步长）`:

输出从1到99

```
for i in range(1,100):
print(i)
#左包含右不包含
```

输出100到200的偶数

```
for j in range(100,201):
    if j%2==0:
        print(j,end=' ')
```

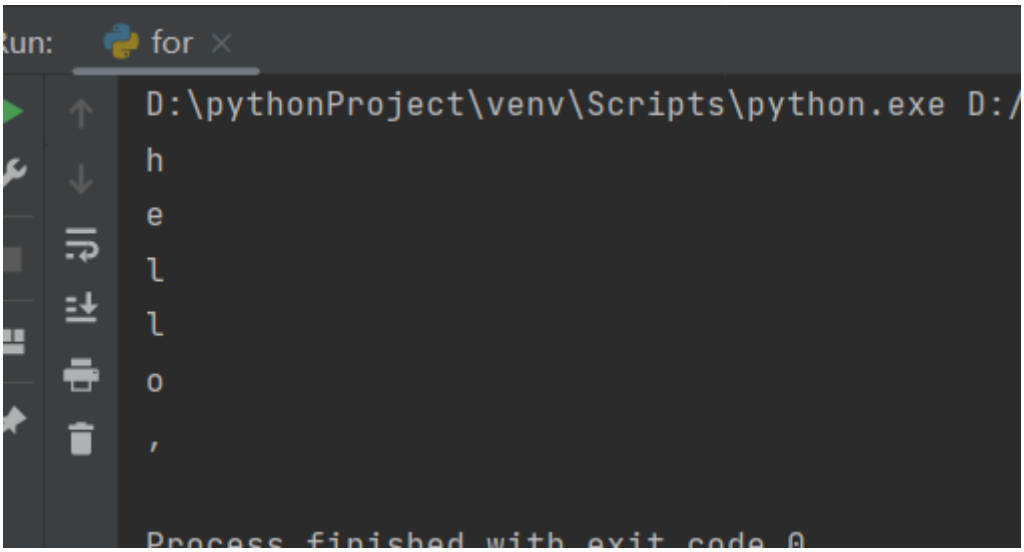
break与continue

break：表示中断本层循环

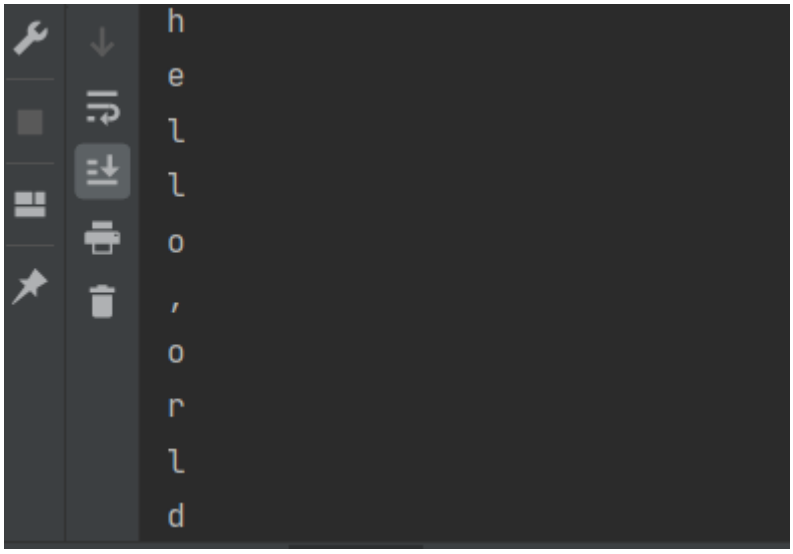
continue：表示结束本次循环，继续进行下次循环

```
exp='hello,world'
for i in exp:
    if i=='w':
        break
    print(i)
```

结果:



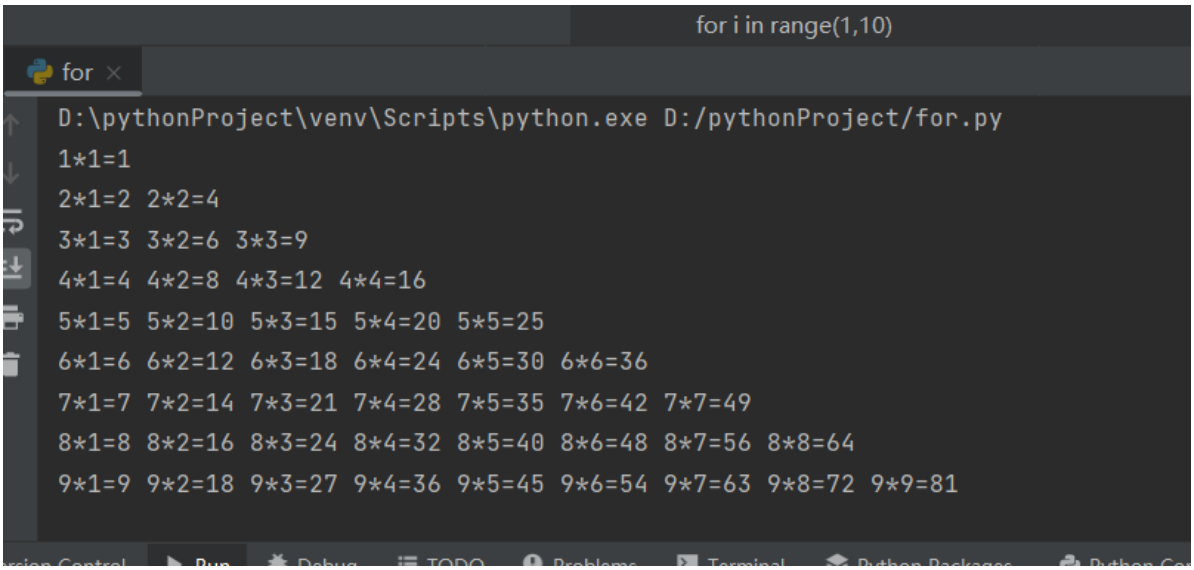

```
exp='hello,world'
for i in exp:
    if i=='w':
        continue
    print(i)
```



break和continue后面的语句都不执行

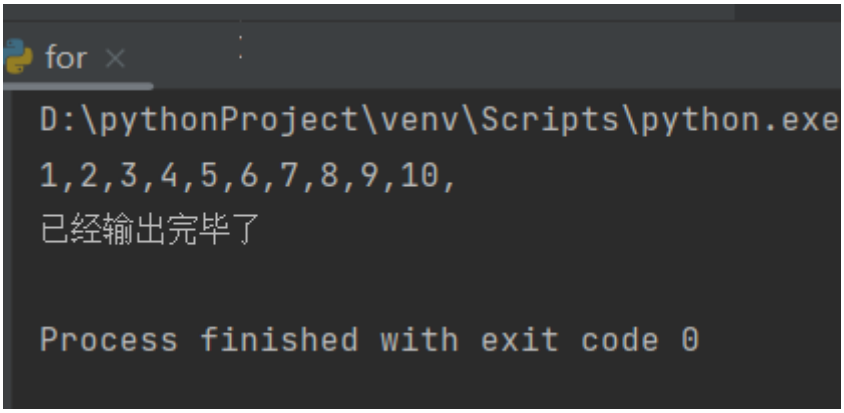
用for打印乘法表

```
for i in range(1,10):
    for j in range(1,i+1):
        print('{}*{}={}'.format(i,j,i*j),end=' ')
    print()  #换行，print默认会换行
```



for---else

```
for i in range(1,11):
    print(i,end=',')
else:
    print()
    print('已经输出完毕了')
```




```python
for i in range(1,11):
    print(i,end=',')
    if i>=5:
        break
else:
    print('上面的循环中，只要出现了break，else的代码不执行')
```

输入密码限制次数

```python
passwd='412'
for i in range(1,4):
    if passwd==input('请输入您的密码'):
        break
else:
    print('次数用完，账号已
```

while--else

和for-else的用法一致

day2作业

猜年龄小游戏，有三点要求

1. 允许用户最多尝试三次
2. 每尝试三次后，如果还没猜对，就询问是否还想继续玩，如果回答Y或者y，就在让他玩三次，以此往复，如果回答N或者n，就退出程序
3. 如果猜对了，就直接退出游戏

```python
import random
sign=0
a=''
while sign==0 or a=='y' or a=='Y':
    print('游戏开始')
    count = 0
    age = random.randint(12, 22)
    sign=1
    while count<3:
        guess_age=int(input('你猜猜我多少岁？'))
        if guess_age>age:
            print("呜呜呜，我有这么老吗？")
```

```
            count+=1
        if guess_age==age:
            print('恭喜你猜对了！')
            break;
        if guess_age<age:
            print('原来我这么年轻')
            count+=1
        if count==3:
            a=input('你已经输了，是否要继续[Y/y],[N/n]')
    if a=='n' or a=='N':
        break
```

计算IBM数值

输入身高和体重，请根据BMI公式(体重除以身高的平方，单位m，kg)帮小王计算他的BMI指数

低于18.5：过轻

18.5-25：正常

25-28：过重

28-32：肥胖

高于32：严重肥胖

```
height=float(input('请输入你的身高，单位m'))
weight=float(input('请输入你的体重，单位kg'))
BMI=weight/height**2
if 0<BMI<18.5:
    print('too thin')
elif 18.5<=BMI<=25:
    print('Normal')
elif 25<BMI<=28:
    print('too fat')
elif 28<BMI<=32:
    print('obesity')
elif BMI>32:
    print('serious obesity')
else:
    print('invalid value')
```

字符串操作

序列：一组按照顺序排列的值【数据集合】

在python中存在三中内置的序列类型：

- 字符串
- 列表
- 元组

获取第n个字符

```
test='python'
print('获取第一个字符,%s'%test[0])
print('获取第二个字符,%s'%test[1])
```

遍历字符串

```
test='python'
for i in test:
    print(i,end=' ')
```

首字母变大写

capitalize()

```
test='python'
print('首字母变大写%s'%test.capitalize())
```

去除空格(左，右)

strip，lstrip，rstrip

```
test='   python   '
print(test.strip())
print(test.lstrip()) #去除左边空格
print(test.rstrip()) #去除右边空格
```

复制字符串

```
a='python'
b=a # 在这里把a的内存地址给了b，也就是说id(a)=id(b)
```

查找数据是否存在

find和index都可以，index如果没找到会报错，find没找到返回-1

```
data='i love yoyu'
print(data.find('y')) #返回值为7，就是对应的下标值,返回的是第一个出现的
print(data.find('M'))#返回-1，没找到就是-1
```

以...开头或者以...结尾

startswith和endswith

```
data='i love python'
print(data.startswith('i'))#返回true
print(data.endswith('p'))#返回false
```

大小写转换

lower，upper

```
data='I LOVE PYTHON'
print(data.lower())
data2='i love python'
print(data.upper())
```

切片截取

[start:.end:step] 左闭右开

```
test='pythonasd'
print(test[2:5])#返回tho,不包括5
print(test[2:])#从第三个开始取到最后
print(test[::-1])#倒序输出
```

列表操作

列表的定义

```
list=[1,2,3,'你好','1.5','true']
```

len()表示字符串的长度，len(list)=6

print(list[0])输出第一个元素也就是1

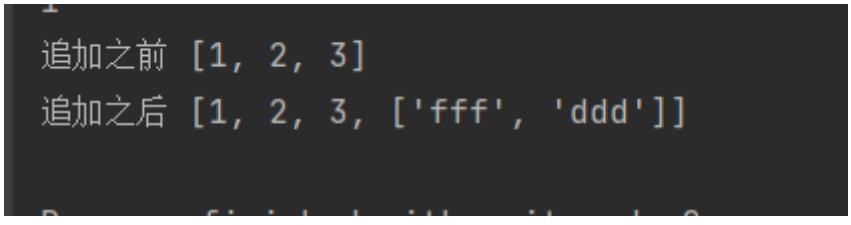
print(list[1:3])从第二个开始到第三个，也就是[2,3]，返回值也是列表

print(list*2)等于输出两次，返回一个列表[1,2,3,'你好','1.5','true'，1,2,3,'你好','1.5','true']

列表追加元素

append每次只可以追加一个元素

```
listA=[1,2,3]
print('追加之前',listA)
listA.append(['fff','ddd']) #可以追加一个列表
listA.append('fff','ddd')  #会报错一次只可以追加一个元素
print('追加之后',listA)
```



列表插入

```
listA=[1,2,3]
listA.insert(1,'第二个位置插入')
```

```
[1, '第二个位置插入', 2, 3]

Process finished with exit code 0
```

批量增加

```
listB=[1,2,3,4]
DATA=range(10)
print(DATA)
DATA2=list(range(10))
listB.extend(DATA2)#等价于listB.extend([0,1,2,3,4,5,6,7,8,9])
print(listB)
```

```
range(0, 10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[1, 2, 3, 4, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

修改

```
listA=[1,2,3,4]
listA[0]=2#修改第一个值
```

删除

```
listB=list(range(10,30))
print(listB)
del listB[0]
print(listB)
```

```
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
[11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
```

- 批量删除

```
listB=list(range(10,30))
print(listB)
del listB[1:3]
print(listB)
```

```
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
[10, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
```

- 移除指定元素

```
listB=list(range(10,30))
print(listB)
listB.remove(20)#只可以移除一项
print(listB)
```

```
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 24, 25, 26, 27, 28, 29]
```

```
listB=list(range(10,30))
print(listB)
b=listB.pop(2)#也是移除，移除指定选项，你可以b=listB.pop(2)来接收
print(listB)
```

```
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
12
[10, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
```

查找

```
listB=list(range(10,30))
print(listB.index(19,20,25))#在下标20-25中查找数值为19的元素
```

元组

元组：是一种不可变的序列，在创建之后不能被修改，用（）创建，也可以是任何类型，用逗号分割数据项。

元组的创建和遍历

```
tupleA=('abcd',89,91.1,'kino',[11,12,13])
print(type(tupleA))
print(tupleA)
#遍历
for i in tupleA:
    print(i,end=' ')
print()
print(tupleA[2:4])
```

```
<class 'tuple'>
('abcd', 89, 91.1, 'kino', [11, 12, 13])
abcd 89 91.1 kino [11, 12, 13]
(91.1, 'kino')
```

元组每次数值发生变化，内存地址都会变

元组中的列表可以修改

```
tupleA=('abcd',89,91.1,'kino',[11,12,13])
print(tupleA[4])
tupleA[4][0]=555
print(tupleA[4])
```

```
[11, 12, 13]
[555, 12, 13]

Process finished with exit code 0
```

- 当元组中只有一个数据必须在后面加逗号

  `tupleA=(1,)`，不加逗号就是单纯整数型的1

统计字符出现次数

count

```
tupleA='aaaaweweeessssszzzx'
print(tupleA.count('a'))  #返回值是4
```

字典操作

字典：由键名->值组成的数据，可以增删修改，没有下标概念不能索引，用{}表示字典对象，每个键值对用逗号分隔，每个键都是唯一的，如果存在重复的键，后者覆盖前者

创建字典

```
dictA={}
print(type(dictA))
dictA['name']='奇诺'
dictA['age']=16
dictA['gender']='girl'
print(dictA)
print(dictA['name'])
#或者用下述方法也是一样的
dictA={'name':'奇诺','age':16,'gender':'girl'}
```

```
<class 'dict'>
奇诺
{'name': '奇诺', 'age': 16, 'gender': 'girl'}
```

字典的长度

```
dictA={'name':'奇诺','age':16,'gender':'girl'}
print(len(dictA))  #返回值为3
```

获取所有的键，所有的值，所有的数据项

```
dictA={'name':'奇诺','age':16,'gender':'girl'}
print(dictA.keys())
print(type(dictA.keys()))
print(dictA.values())
print(type(dictA.values()))
print(dictA.items())
print(type(dictA.items()))
```

```
dict_keys(['name', 'age', 'gender'])
<class 'dict_keys'>
dict_values(['奇诺', 16, 'girl'])
<class 'dict_values'>
dict_items([('name', '奇诺'), ('age', 16), ('gender', 'girl')])
<class 'dict_items'>
```

遍历字典

```python
dictB={'name':'奇诺','age':'16','gender':'girl'}
for x,y in dictA.items():
    print(x,end=',')
    print(y,end=',')
    print('%s==%s' % (x, y),end=',')
    print()
```

```
D:\pythonProject\venv\Scripts\python.exe D:/python
name,奇诺,name==奇诺,
age,16,age==16,
gender,girl,gender==girl,

Process finished with exit code 0
```

字典修改

```python
dictA={'name':'奇诺','age':16,'gender':'girl'}
print(dictA)
#  dictA.update({'age':18},{'height':1.61})这个会报错，update一次最多修改一个
dictA.update({'age':18})
dictA.update({'height':1.61})
print(dictA)
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/字典.py
{'name': '奇诺', 'age': 16, 'gender': 'girl'}
{'name': '奇诺', 'age': 18, 'gender': 'girl', 'height': 1.61}

Process finished with exit code 0
```

字典删除

```
dictA={'name':'奇诺','age':16,'gender':'girl'}
print(dictA)
del dictA['name'] #或者dictA.pop('name')
print(dictA)
```

```
{'name': '奇诺', 'age': 16, 'gender': 'girl'}
{'age': 16, 'gender': 'girl'}
```

字典排序

- 按照key(键名)排序

```
dictA={'b':3,'a':2,'c':1,'d':6}
print(sorted (dictA.items(), key=lambda x:x[0]))
#dictA.items()=[('b',3),('a',2),('c',1),('d',6)]
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/字典.
[('a', 2), ('b', 3), ('c', 1), ('d', 6)]

Process finished with exit code 0
```

- 按照value（值）排序

```
dictA ={'b':3,'a':2,'c':1,'d':6}
print(sorted (dictA.items(), key=lambda x:x[1]))
#dictA.items()=[('b',3),('a',2),('c',1),('d',6)]
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/字典.py
[('c', 1), ('a', 2), ('b', 3), ('d', 6)]

Process finished with exit code 0
```

这上面的x可以是任意的东西，换成任意字符都可以,dictA.items()是一个列表，里面有四个元组，上面的x:x[]意思是以第几个字段排序，以第0个就是key，第1个就是value

共用操作

操作	描述	适用
合并操作：+	两个对象相加操作，会合并两个对象	字符串，列表，元组
赋值：*	对象自身按指定次数进行+操作	字符串，列表，元组
in判断元素是否存在	判断指定元素是否存在与对象中	字符串，列表，元组，字典

```
print('人生苦短'+'我用python')
list1=list(range(1,10))
list2=list(range(10,20))
print(list1+list2)
tuple1=tuple(range(10,15))
tuple2=tuple(range(15,20))
print(tuple1+tuple2)
```

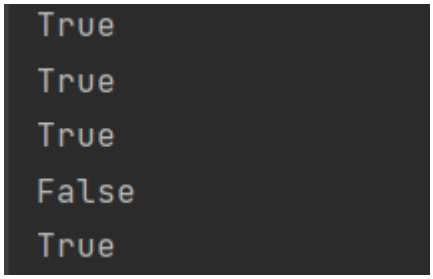
```
人生苦短我用python
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
(10, 11, 12, 13, 14, 15, 16, 17, 18, 19)

Process finished with exit code 0
```

```
print('二次元'*3)
list1=list(range(1,10))
print(list1*3)
tuple1=tuple(range(10,15))
print(tuple1*3)
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/公用操作.py
二次元二次元二次元
[1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 1, 2, 3, 4, 5, 6, 7, 8, 9]
(10, 11, 12, 13, 14, 10, 11, 12, 13, 14, 10, 11, 12, 13, 14)
```

```
str1='我是二次元'
list1=['我','是','二次元']
tuple1=('我','是','二次元')
dict1={1:'我',2:'是',3:'二次元'}
print('二次元'in str1)
print('二次元'in list1)
print('二次元'in tuple1)
print('二次元'in dict1)
print(1 in dict1)
```



特别注意，字典判断是判断key值到底存不存在

函数初识

def 函数名（参数）：

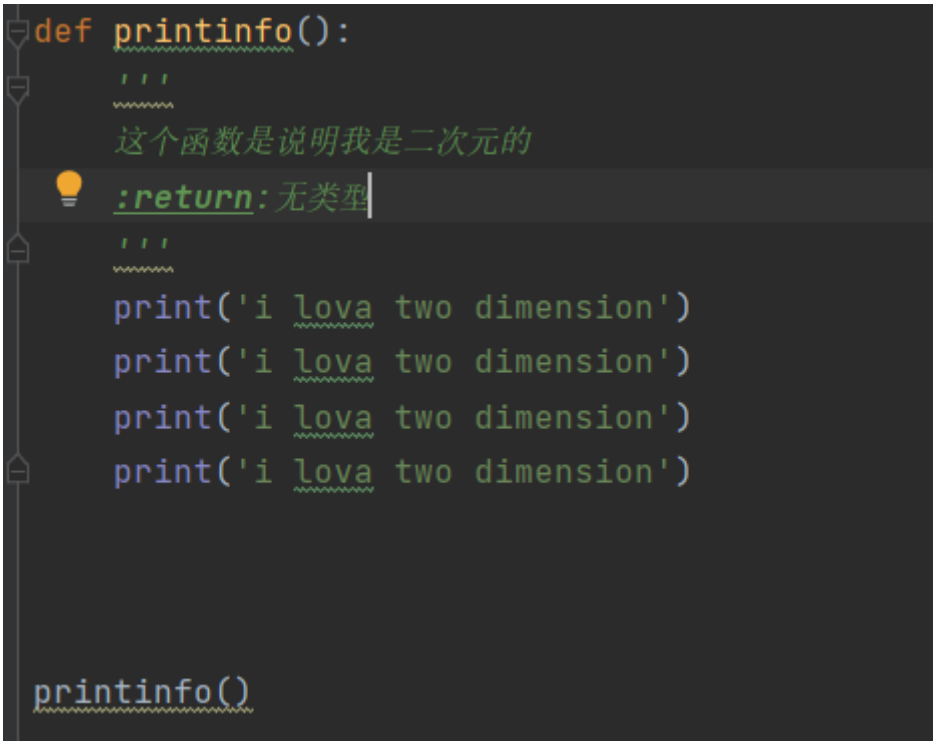
代码块

函数创建和使用

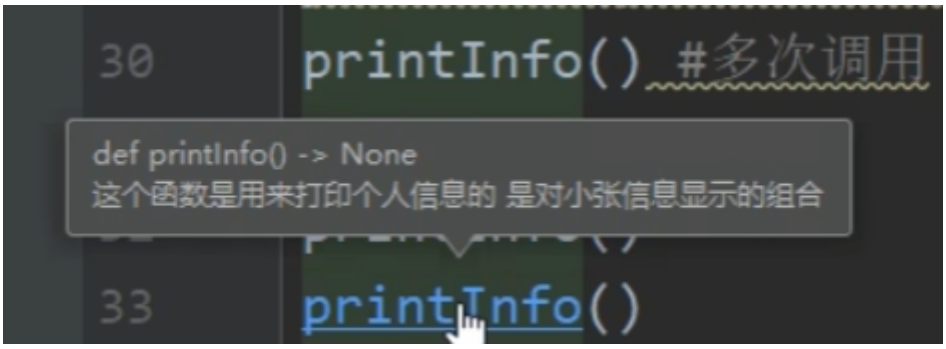
```
def printinfo():
    print('i lova two dimension')
    print('i lova two dimension')
    print('i lova two dimension')
    print('i lova two dimension')


printinfo()
```

在自定义函数中敲三引号会自动生成函数的备注注释信息：



同时按住ctrl，把鼠标拖到printinfo()上会显示备注信息



函数参数简单使用

```
def printinfo(a,b,c):
    '''
    这个函数是说明我是二次元的
    :return:无类型
    '''
    print('myname is %s,my score is %d,my age is %d'%(a,b,c))

name='kino'
score=99
age=19
printinfo(name,score,age)
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/函数初识.py
myname is kino,my score is 99,my age is 19

Process finished with exit code 0
```

函数参数

参数的分类：必选参数，默认参数，可选参数，关键字参数

```python
def sum(a=10,b=20):  #a,b只是形式参数，不占用内存空间，这里设置a的默认值是10，b的是20
    sum=a+b
    print(sum)

sum(20,30)
sum()#使用默认值
```

```
函数初识
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/函数初识
50
30

Process finished with exit code 0
```

```python
def sum1(a,b=40):
    print('默认参数使用=%d'%(a+b))
    pass
# 默认参数调用
sum1(10) #在调用的时候如果未赋值，就会用定义函数时给定的默认值
sum1()
```

```
Run:  函数参数
E:\PythonPro\Python-Func\venv\Scripts\python.exe E:/PythonPro/Python-Func/函数参数.py
默认参数使用=50

Process finished with exit code 0
```

可变参数

```python
def count(*args):
    result=0
    for i in args:
        result+=i
    print('最终和为%d'%result)

count(1)
count(1,2,5)
count(1,2,3,4,5,6,7,8,9,10)
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/函数初识.p
最终和为1
最终和为8
最终和为55

Process finished with exit code 0
```

关键字可变参数

在函数体内，参数关键字是一个字典类型，key是一个字符串

```python
def keyFunc(**kino):
    print(kino)

dictA={'name':'奇诺','age':16,'gender':'girl'}
#keyFunc(dictA) #这样仍然会报错
keyFunc(**dictA)#或者keyFunc(name='奇诺',age=16,gender='girl')
#可以不穿参数输出空字典
keyFunc()
```

```
D:\pythonProject\venv\Scripts\python.exe D:/pythonProject/函数初识.
{'name': '奇诺', 'age': 16, 'gender': 'girl'}
```

这里假如用dictA前面一定要加两个**

参数混合使用

```python
ef mix(*a,**b):
    print(a)
    print(b)
    pass

mix(1,2,3,4)
mix(1,2,3,4,name='kino',gender='girl')
mix(name='kino',gender='girl')
```

可选参数必须放到关键字参数之前