一 课程回顾

```
# 1. 创建一个空字典my_dict。
my_dict = {}
print(my_dict)
# 2. 向字典my_dict中添加键值对'apple': 3、'banana': 6和'orange': 4。
my_dict['apple'] = 3
my_dict['banana'] = 6
my_dict['orange'] = 4
print(my_dict)
# 3. 获取字典my_dict中键为'banana'的值，并将其存储在变量value中。
# 字典名[键]
value = my_dict['banana']
print(value)

# 4. 将字典my_dict中键为'orange'的值增加2。
my_dict['orange'] += 2
print(my_dict)


# 5. 删除字典my_dict中键为'apple'的键值对。
del my_dict['apple']

# 6. 检查字典my_dict中是否存在键'grape'。
a = 'grape' in my_dict
print(a)
# 7. 获取字典my_dict中所有键，并将其存储在列表keys中。
keys = list(my_dict.keys())
print(keys)

# 8. 获取字典my_dict中所有值，并将其存储在列表values中。
values = list(my_dict.values())
print(values)


# 1.创建一个空集合my_set。
my_set = set()
print(my_set)
# 2.向集合my_set中添加元素'apple'、'banana'和'orange'。
my_set.add('apple')
my_set.add('banana')
my_set.add('orange')
print(my_set)

# 3.从集合my_set中删除元素'banana'。
my_set.remove('banana')
print(my_set)

# 4.创建另一个集合my_set，包含元素'kiwi'和'grape'。
my_set2 = {'kiwi','grape'}

print(my_set2)
```

```
# 5.获取两个集合my_set和my_set2的并集。
print(my_set | my_set2)

# 6.获取两个集合my_set和my_set2的交集。
print(my_set & my_set2)

# 7.获取两个集合my_set和my_set2的差集。
print(my_set - my_set2)

# 8.判断集合`my_set`是否是集合`my_set2`的子集。
print(my_set < my_set2)

c = my_set.issubset(my_set2)  # myset是否是my_set2的子集
print(c)
```

```
数据类型的总结：
总结容器： 统一管理数据

数值类型： int  float  bool
序列类型： str  list  tuple
散列类型： dict  set

可变类型： 列表  字典  集合
不可变类型： 数字  字符串  元祖
可变和不可变的概念：
可变：值发生改变时，内存地址不变，也就是id不变，证明在改变原来的值
不可变：值发生改变时，内存地址也发生改变，也就是id也改变，证明是没有再改原来的值，而是产生了新的值

序列： 相邻有序，定位灵活(索引，切片)
散列类型：分散无序，定位迅速(键)
```

二 函数基础 上

2.1 函数简介 function

现在的代码可复用性非常差

函数其实也是存数据的

```
总结函数的优点：
1.遇到重复功能的时候，直接调用就可以，减少代码量
2.提升代码，项目的结构性，分工明确，提高代码的可读性
3.遇到扩展功能时，修改比较方便
```

自定义函数，将一段有规律的可以重复使用的代码定义成函数，一次编写，多次调用

```
n = 0  # 存储累加的 容器
for i in 'Process finished with exit code 0':
    n += 1
print(n)

print(len('Process finished with exit code 0'))
```

2.2 函数的定义

```
定义函数：
def 函数名(形参1，形参2，形参n):
    代码块
```

```
def fn():       #  定义函数
    print('这是我的第一个函数')
```

2.3 函数的调用

```
语法：
    函数名()
   要先创建函数 再调用

def fn():       #  定义函数
    print('这是我的第一个函数')
    print('hello')
    print('world')
    print('再见')
fn()

print(fn)  # 打印函数名  就能得出函数内存地址

注意：
print(fn)     函数对象 也就是函数地址
fn()          调用函数

练习：
    定义一个login函数 功能是输入用户名和密码，验证是否正确 账号密码 ： root  123

def login():
    username = input('请输入用户名:')
    password = input('请输入密码:')
    if username == 'root' and password == '123':
        print('登陆成功')
    else:
        print('登录失败')
login()
```

2.4 函数的参数

2.4.1 形参和实参

定义一个函数，可以用来求任意两个数的和

```
def sum():
    print(1+1)
sum()

def sum():
    a = 55
    b = 200
    print(a+b)
sum()
```

函数的参数:在定义函数时，可以在函数名（）中定义数量不等的形参 注意：可以有也可以没有，多个形参之间使用逗号隔开

实参：实际参数，在函数定义时指定了形参，再调用的时候必须传递实参，实参会赋值给对应的形参，有几个形参就必须传递几个实参

```
def fn(a,b):
    print(a+b)
    print('a=',a)
    print('b=',b)
fn(13212,51531)
```

注意：定义了多少个形参就要传递多少个实参

2.4.2 默认值参数

```
def fn(a,b,c=10):  # c = 10就是默认值参数
    print(a)
    print(b)
    print(c)
fn(1,2,3)  # 实参优先级  大于 默认值参数


def fn(a,b=10,c=10):  # c = 10就是默认值参数
    print(a)
    print(b)
    print(c)
fn(1)  # 实参优先级  大于 默认值参数
```

2.4.3 位置参数和关键字参数

```
1. 位置参数就是将对应位置的实参赋值给对应位置的形参
def ms(年龄，姓名，手机号，QQ号，微信号，国家，省份，特长):
    pass
ms()
2.关键字参数：可以不按照形参定义的顺序来传递，而是直接根据参数名来传递参数
def fn(a,b,c):
```

```
    print('a=',a)
    print('b=',b)
    print('c=',c)
fn(b=1,c=3,a=2)
```

好处是什么：不需要记忆我们参数的一个顺序，只需要记住名字就可以了
位置参数和关键字参数可以混合使用
混合使用关键字参数和位置参数时，位置参数必须在关键字参数的前面
位置参数是所有参数里面优先级最高的

```
def fn(a,b,c):
    print('a=',a)
    print('b=',b)
    print('c=',c)
fn(1,c=2,b=5)
```

2.4.4 实参的类型

函数在调用的时候不会限制你的类型，实参可以传递任意类型

```
def fn(a):
    print('a=',a)
b = 123
fn(b)  #  b = 123
fn(123)


def fn(a):
    print('a=',a)
b = True
fn(b)  #  b = 123
fn(123)

def fn(a):
    print('a=',a)
b = 'hello'
fn(b)  #  b = 123
fn(123)

def fn(a):
    print('a=',a)
b = [1,2,34]
fn(b)  #  b = 123
fn(123)


def fn(a):
    print('a=',a)
b = fn
fn(b)  #  b = 123
fn(123)
```

取消缩进： 框选要取消的内容 按住shift + tab

作业

1.定义一个函数，可以用来求任意三个数的乘积

2.定义一个函数，可以根据不同的用户名显示欢迎信息

3.编写一个函数，功能是输出2个数中较大的一个数

编写一个函数，判断用户传入的列表长度是否大于2，如果大于2，就保留前两个 并输出出来

编写一个函数，接收一个列表(列表元素个数不限)，求列表数字的和

写一个函数，判断用户传入的对象（字符串、列表、元组）的元素是否为空

输入颜色（RGBA），打印描述信息，否则提示颜色不存在(这道题不是一定得用函数写)

```
"R" -->"红色"
"G" -->"绿色"
"B" -->"蓝色"
"A" -->"透明度"
```

python 集合跟数学有点关系 并集 交集 差集 子集