# 一、函数

1.求绝对值的 my_abs 函数

```
def my_abs(x):
    if x >= 0:
        return x
    else:
        return -x
```

2.（函数参数）计算'jshdjkshkdhsk'的个数

```
def fun(s):
    count=0
    for i in s:
        count+=1
    return count
#函数调用
str=fun('jshdjkshkdhsk')
print(str)
```

3.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，只保留前两个，并将新内容返回给调用者

```
def func(l):
     if len(l)>2:
         l = l[0:2]
         return l
     else:
         return l
print(func([1,2,3,4]))
print(func([1,2]))

#结果：[1, 2]
#      [1, 2]
```

# 二、类与对象

1.定义一个学生 student 类。有下面的类属性：

(1)姓名 name

(2)年龄 age

(3)成绩 score（语文，数学，英语) [每课成绩的类型为整数]

类方法：

(1)获取学生的姓名：get_name() 返回类型:str

(2)获取学生的年龄：get_age() 返回类型:int

(3)返回 3 门科目中最高的分数。get_course() 返回类型:int

写好类以后，可以定义 2 个同学测试下:

zm = student('zhangming',20,[69,88,100])

返回结果：

zhangming

20

100

```python
class student():
    # 构造函数
    # 对当前对象的实例的初始化
    def __init__(self, name, age, score):
        self.name = name
        self.age = age
        self.score = score

    #   isinstance函数判断一个对象是否是一个已知的类型，类似type
    def get_name(self):
        if isinstance(self.name, str):
            return self.name

    def get_age(self):
        if isinstance(self.age, int):
            return self.age

    def get_course(self):
        a = max(self.score)
        if isinstance(a, int):
            return a


zm = student('zhangming', 20, [69, 88, 100])
print(zm.get_name())
print(zm.get_age())
print(zm.get_course())
```

2.定义一个字典类：dictclass。完成下面的功能：

dict = dictclass({你需要操作的字典对象})

(1)删除某个 key

del_dict(key)

(2)判断某个键是否在字典里，如果在返回键对应的值，不存在则返回"not found"

get_dict(key)

(3)返回键组成的列表：返回类型:(list)

get_key()

(4)合并字典，并且返回合并后字典的 values 组成的列表。返回类型:(list)

update_dict({要合并的字典})

```python
class dictclass():

    # 构造函数
    # 对当前对象的实例的初始化
    def __init__(self, class1):
        self.classs = class1

    def del_dict(self, key):
        if key in self.classs.keys():
            del self.classs[key]
            return self.classs
        return "不存在这个值，无需删除"

    def get_dict(self, key):
        if key in self.classs.keys():
            return self.classs[key]
        return "not found"

    def get_key(self):
        return list(self.classs.keys())

    def update_dict(self, dict1):
        # 方法1
        # self.classs.update(dict1)
        # 方法2,对于重复的key，b会覆盖a
        a = dict(self.classs, **dict1)
        return a


a = dictclass({"姓名": "张三", "年龄": "18", "性别": "男"})
print(a.del_dict("年龄"))
print(a.get_dict("姓名"))
print(a.get_key())
print(a.update_dict({"年薪": 0}))
```

# 三、魔法方法

1.构建一个 Vector 类，具有以下功能：

(1)获取长度

(2)获取指定位置的元素

(3)设定指定位置的数值

(4)进行加法并以指定格式输出。（提示：print 函数可以通过改写 __str(self）__ 方法）

示例：

a=Vector([1,2,3])

b=Vector([2,3,4])

print(a+b)的结果为<3,5,7>

```
class Vector :
    def __init__(self, d):
        if isinstance(d, int):
            self._coords = [0] * d
        else:
            try: # we test if param is iterable
                self._coords = [val for val in d]
            except TypeError:
                raise TypeError('invalid parameter type')
    def __len__(self):
        """ Return the dimension of the vector . """
        return len(self._coords)
    def __getitem__(self , j):
        """ Return jth coordinate of vector . """
        return self._coords[j]
    def __setitem__(self, j, val):
        """ Set jth coordinate of vector to given value . """
        self._coords[j] = val
    def __add__(self ,other ):
        """ Return sum of two vectors . """
        if len(self) != len(other): # relies on __len__ method
            raise ValueError('dimensions must agree')
        result = Vector(len(self)) # start with vector of zeros
        for j in range(len(self)):
            result[j] = self[j] + other[j]
        return result
```

```
a=Vector([1,2,3,4,5])
b=Vector([1,3,5,7,9])
c=Vector([1,2])
len(a)
print(a)
a[2]
a[3]=5
print(a)
print(a+b)
print(a+c)
```

```
<1, 2, 3, 4, 5>
<1, 2, 3, 5, 5>
<2, 5, 8, 12, 14>
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-1-112861a8ea34> in <module>
     38 print(a)
     39 print(a+b)
---> 40 print(a+c)

<ipython-input-1-112861a8ea34> in __add__(self, other)
     20         """ Return sum of two vectors . """
     21         if len(self) != len(other): # relies on __len__ method
---> 22             raise ValueError('dimensions must agree')
     23         result = Vector(len(self)) # start with vector of zeros
     24         for j in range(len(self)):

ValueError: dimensions must agree
```

2. 构建一个类 TimeTest:

(1)在不构造实例的情况下可以通过 showTime 函数获取当前时间。

(2)构建函数 whatTime，通过输入小时、分钟、秒，获取“小时：分钟：秒”格式的时间。

```
import time
class TimeTest(object):
    def __init__(self, hour, minute, second):
        self.hour = hour
        self.minute = minute
        self.second = second

    @staticmethod
    def showTime():
        return time.strftime("%H:%M:%S", time.localtime())

    def whatTime(self):
        print(self.hour, self.minute,self.second,sep=':')
print(TimeTest.showTime())
t = TimeTest(2, 10, 10)
t.whatTime()
```

```
21:58:11
2:10:10
```