

python

The Python logo, consisting of two interlocking snakes, one blue and one yellow, is positioned below the word "python".

```
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")

for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
    turtle.fd(40)
    turtle.circle(16, 180)
    turtle.fd(40 * 2/3)
```

Python语言程序设计

字典类型及操作



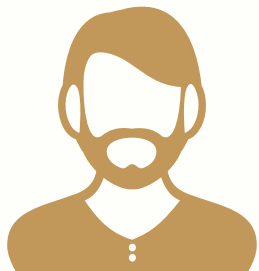
嵩 天
北京理工大学





单元开篇

字典类型及操作



- 字典类型定义
- 字典处理函数及方法
- 字典类型应用场景





字典类型定义

字典类型定义

理解“映射”

- 映射是一种键(索引)和值(数据)的对应



字典类型定义

理解“映射”

- 映射是一种键(索引)和值(数据)的对应

内部颜色：蓝色

外部颜色：红色



"streetAddr" : "中关村南大街5号"
"city" : "北京市"
"zipcode" : "100081"

字典类型定义

理解“映射”

- 映射是一种键(索引)和值(数据)的对应

["python", 123, ".io"]



0



1



2



内部颜色：蓝色

外部颜色：红色

序列类型由0..N整数作为数据的默认索引

映射类型则由用户为数据定义索引

字典类型定义

字典类型是“映射”的体现

- 键值对：键是数据索引的扩展
- 字典是键值对的集合，键值对之间无序
- 采用大括号{}和dict()创建，键值对用冒号: 表示

{<键1>:<值1>, <键2>:<值2>, ... , <键n>:<值n>}

字典类型的用法

在字典变量中，通过键获得值

$\langle \text{字典变量} \rangle = \{ \langle \text{键1} \rangle : \langle \text{值1} \rangle, \dots, \langle \text{键n} \rangle : \langle \text{值n} \rangle \}$

$\langle \text{值} \rangle = \langle \text{字典变量} \rangle [\langle \text{键} \rangle] \quad \langle \text{字典变量} \rangle [\langle \text{键} \rangle] = \langle \text{值} \rangle$

[] 用来向字典变量中索引或增加元素

字典类型定义和使用

```
>>> d = {"中国":"北京", "美国":"华盛顿", "法国":"巴黎"}
```

```
>>> d
```

```
{'中国': '北京', '美国': '华盛顿', '法国': '巴黎'}
```

```
>>> d["中国"]
```

```
'北京'
```

```
>>> de = {} ; type(de)
```

```
<class 'dict'>
```

type(x)

返回变量x的类型



字典处理函数及方法

字典类型操作函数和方法

函数或方法	描述
<code>del d[k]</code>	删除字典d中键k对应的数据值
<code>k in d</code>	判断键k是否在字典d中，如果在返回True，否则False
<code>d.keys()</code>	返回字典d中所有的键信息
<code>d.values()</code>	返回字典d中所有的值信息
<code>d.items()</code>	返回字典d中所有的键值对信息

字典类型操作

```
>>> d = {"中国":"北京", "美国":"华盛顿", "法国":"巴黎"}
```

```
>>> "中国" in d
```

```
True
```

```
>>> d.keys()
```

```
dict_keys(['中国', '美国', '法国'])
```

```
>>> d.values()
```

```
dict_values(['北京', '华盛顿', '巴黎'])
```

字典类型操作函数和方法

函数或方法	描述
d.get(k, <default>)	键k存在，则返回相应值，不在则返回<default>值
d.pop(k, <default>)	键k存在，则取出相应值，不在则返回<default>值
d.popitem()	随机从字典d中取出一个键值对，以元组形式返回
d.clear()	删除所有的键值对
len(d)	返回字典d中元素的个数

字典类型操作

```
>>> d = {"中国":"北京", "美国":"华盛顿", "法国":"巴黎"}
```

```
>>> d.get("中国","伊斯兰堡")
```

```
'北京'
```

```
>>> d.get("巴基斯坦","伊斯兰堡")
```

```
'伊斯兰堡'
```

```
>>> d.popitem()
```

```
('美国', '华盛顿')
```


字典功能默写

■ 定义空字典d

```
>>> d = {}
```

■ 向d新增2个键值对元素

```
>>> d["a"] = 1; d["b"] = 2
```

■ 修改第2个元素

```
>>> d["b"] = 3
```

■ 判断字符"c"是否是d的键

```
>>> "c" in d
```

■ 计算d的长度

```
>>> len(d)
```

■ 清空d

```
>>> d.clear()
```



字典类型应用场景

字典类型应用场景

映射的表达

- 映射无处不在，键值对无处不在
- 例如：统计数据出现的次数，数据是键，次数是值
- 最主要作用：表达键值对数据，进而操作它们

字典类型应用场景

元素遍历

```
for k in d :
```

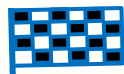
＜语句块＞



单元小结

字典类型及操作

- 映射关系采用键值对表达
- 字典类型使用{}和dict()创建，键值对之间用:分隔
- d[key] 方式既可以索引，也可以赋值
- 字典类型有一批操作方法和函数，最重要的是.get()





小花絮



冈仁波齐：每个人心中都有一个神圣的目标，你的是什么？

