马哥教育

IT人的高薪职业学院

Python内置数据结构

讲师：Wayne

从业十余载，漫漫求知路

字典dict

- key-value键值对的数据的集合
- **可变的**、**无序的**、key**不重复**

马哥教育
IT人的高薪职业学院

字典dict定义 初始化

- d = dict() 或者 d = {}
- dict(**kwargs) 使用name=value对初始化一个字典
- dict(iterable, **kwarg) 使用可迭代对象和name=value对构造字典，不过可迭代对象的元素必须是一个二元结构
  - d = dict(((1,'a'),(2,'b'))) 或者 d = dict(([1,'a'],[2,'b']))
- dict(mapping, **kwarg) 使用一个字典构建另一个字典
- d = {'a':10, 'b':20, 'c':None, 'd':[1,2,3]}
- 类方法dict.fromkeys(iterable, value)
  - d = dict.fromkeys(range(5))
  - d = dict.fromkeys(range(5),0)

字典元素的访问

- d[key]
  - 返回key对应的值value
  - key不存在抛出KeyError异常
- get(key[, default])
  - 返回key对应的值value
  - key不存在返回缺省值，如果没有设置缺省值就返回None
- setdefault(key[, default])
  - 返回key对应的值value
  - key不存在，添加kv对，value为default，并返回default，如果default没有设置，缺省为None

字典增加和修改

- d[key] = value
  - 将key对应的值修改为value
  - key**不存在添加新的**kv**对**
- update([other]) -> None
  - 使用另一个字典的kv对更新本字典
  - key不存在，就添加
  - key存在，覆盖已经存在的key对应的值
  - 就地修改

```
d.update(red=1)
d.update((('red',2),))
d.update({'red':3})
```

字典删除

- pop(key[, default])
  - key存在，移除它，并返回它的value
  - key不存在，返回给定的default
  - default未设置，key不存在则抛出KeyError异常
- popitem()
  - 移除并返回一个任意的键值对
  - 字典为empty，抛出KeyError异常
- clear()
  - 清空字典

字典删除

- del语句

```
a = True
b = [6]
d = {'a': 1, 'b': b, 'c': [1,3,5]}
del a
del d['c'] # 删除了一个对象[1,3,5] ?
del b[0]
c = b
del c
del b
b = d['b']
```

- del a['c'] 看着像删除了一个对象，本质上减少了一个对象的引用，del 实际上删除的是名称，而不是对象

字典遍历

- for ... in dict
  - 遍历key

```
for k in d:
    print(k)

for k in d.keys():
    print(k)
```


马哥教育
IT人的高薪职业学院

字典遍历

- for ... in dict
  - 遍历value

```
for k in d:
    print(d[k])

for k in d.keys():
    print(d.get(k))

for v in d.values():
    print(v)
```

字典遍历

- for ... in dict
  - 遍历item，即kv对

```
for item in d.items():
    print(item)
for item in d.items():
    print(item[0], item[1])
for k,v in d.items():
    print(k, v)
for k, _ in d.items():
    print(k)
for _ ,v in d.items():
    print(v)
```

字典遍历

- 总结
  - Python3中，keys、values、items方法返回一个类似一个生成器的可迭代对象，不会把函数的返回结果复制到内存中
    - Dictionary view对象
    - 字典的entry的动态的视图，字典变化，视图将反映出这些变化
  - Python2中，上面的方法会返回一个新的列表，占据新的内存空间。所以Python2建议使用iterkeys、itervalues、iteritems版本，返回一个迭代器，而不是一个copy

字典遍历和移除

- 如何在遍历的时候移除元素

错误的做法

```
d = dict(a=1, b=2, c='abc')
for k,v in d.items():
    d.pop(k) # 异常

while len(d): # 相当于清空，不如直接clear()
    print(d.popitem())
```

正确的做法

```
d = dict(a=1, b=2, c='abc')
keys = []
for k,v in d.items():
    if isinstance(v, str):
        keys.append(k)

for k in keys:
    d.pop(k)
print(d)
```

字典的key

- key的要求和set的元素要求一致
  - set的元素可以就是看做key，set可以看做dict的简化版
  - hashable 可哈希才可以作为key，可以使用hash()测试
  - d = {1 : 0, 2.0 : 3, "abc" : None, ('hello', 'world', 'python') : "string", b'abc' : '135'}

defaultdict

- collections.defaultdict([default_factory[, ...]])
  - 第一个参数是default_factory，缺省是None，它提供一个初始化函数。当key不存在的时候，会调用这个工厂函数来生成key对应的value

```
import random

d1 = {}
for k in 'abcdef':
    for i in range(random.randint(1,5)):
        if k not in d1.keys():
            d1[k] = []
        d1[k].append(i)

print(d1)
```

```
from collections import defaultdict
import random

d1 = defaultdict(list)
for k in 'abcdef':
    for i in range(random.randint(1,5)):
        d1[k].append(i)

print(d1)
```

OrderedDict

- collections.OrderedDict([items])
  - key并不是按照加入的顺序排列，可以使用OrderedDict记录顺序

```
from collections import OrderedDict
import random
d = {'banana': 3, 'apple': 4, 'pear': 1, 'orange': 2}
print(d)
keys = list(d.keys())
random.shuffle(keys)
print(keys)
od = OrderedDict()
for key in keys:
    od[key] = d[key]
print(od)
print(od.keys())
```

OrderedDict

- 有序字典可以记录元素插入的顺序，打印的时候也是按照这个顺序输出打印
- 3.6版本的Python的字典就是记录key插入的顺序（IPython不一定有效果）

- 应用场景：
  - 假如使用字典记录了N个产品，这些产品使用ID由小到大加入到字典中
  - 除了使用字典检索的遍历，有时候需要取出ID，但是希望是按照输入的顺序，因为输入顺序是有序的
  - 否则还需要重新把遍历到的值排序

字典练习

- 用户输入一个数字
  - 打印每一位数字及其重复的次数
- 数字重复统计
  - 随机产生100个整数
  - 数字的范围[-1000, 1000]
  - 升序输出所有不同的数字及其重复的次数
- 字符串重复统计
  - 字符表'abcdefghijklmnopqrstuvwxyz'
  - 随机挑选2个字母组成字符串，共挑选100个
  - 降序输出所有不同的字符串及重复的次数

谢谢