05. 万恶之源-基本数据类型(dict)

本节主要内容:

1. 字典的简单介绍
2. 字典增删改查和其他操作
3. 字典的嵌套

一. 字典的简单介绍

字典(dict)是python中唯一的一个映射类型.他是以{ }括起来的键值对组成. 在dict中key是唯一的. 在保存的时候, 根据key来计算出一个内存地址. 然后将key-value保存在这个地址中. 这种算法被称为hash算法, 所以, 切记, 在dict中存储的key-value中的key'必须是可hash的,
如果你搞不懂什么是可哈希, 暂时可以这样记, 可以改变的都是不可哈希的,
那么可哈希就意味着不可变. 这个是为了能准确的计算内存地址而规定的.

已知的可哈希(不可变)的数据类型: int, str, tuple, bool
不可哈希(可变)的数据类型: list, dict, set

语法 :

{key1: value1, key2: value2....}

注意: key必须是不可变(可哈希)的. value没有要求.可以保存任意类型的数据

```
# 合法
dic = {123: 456, True: 999, "id": 1, "name": 'sylar', "age": 18, "stu": ['帅哥', '美女'], (1, 2, 3): '麻花藤'}
print(dic[123])
print(dic[True])
print(dic['id'])
print(dic['stu'])
print(dic[(1, 2, 3)])
# 不合法
# dic = {[1, 2, 3]: '周杰伦'}   # list是可变的. 不能作为key
# dic = {{1: 2}: "哈哈哈"}      # dict是可变的. 不能作为key
dic = {{1, 2, 3}: '呵呵呵'}     # set是可变的, 不能作为key
```

dict保存的数据不是按照我们添加进去的顺序保存的. 是按照hash表的顺序保存的. 而hash表不是连续的. 所以不能进行切片工作. 它只能通过key来获取dict中的数据

二. 字典的增删改查和其他相关操作

1. 增加

```
dic = {}
dic['name'] = '周润发'      # 如果dict中没有出现这个key, 就会新增一个key-value的组合进dict
dic['age'] = 18
print(dic)

# 如果dict中没有出现过这个key-value. 可以通过setdefault设置默认值
dic.setdefault('李嘉诚')   # 也可以往里面设置值.
dic.setdefault("李嘉诚", "房地产")    # 如果dict中已经存在了. 那么setdefault将不会起作用

print(dic)
```

2. 删除

```
ret = dic.pop("jay")
print(ret)

del dic["jay"]
print(dic)

# 随机删除.
ret = dic.popitem()

# 清空字典中的所有内容
dic.clear()
```

3. 修改

```
dic = {"id": 123, "name": 'sylar', "age": 18}
dic1 = {"id": 456, "name": "麻花藤", "ok": "wtf"}
dic.update(dic1)    # 把dic1中的内容更新到dic中. 如果key重名. 则修改替换. 如果不存在key, 则新增.
print(dic)
print(dic1)
```

4. 查询

查询一般用key来查找具体的数据.

```
print(dic['name'])
# print(dic['sylar'])     # 报错
print(dic.get("ok"))
```

```
print(dic.get("sylar"))      # None
print(dic.get("sylar", "牛B"))   # 牛B
```

5. 其他相关操作

```
dic = {"id": 123, "name": 'sylar', "age": 18, "ok": "科比"}

print(dic.keys())   # dict_keys(['id', 'name', 'age', 'ok']) 不用管它是什么.当成list来用就行
for key in dic.keys():
    print(key)

print(dic.values())     # dict_values([123, 'sylar', 18, '科比']) 一样. 也当list来用
for value in dic.values():
    print(value)

print(dic.items())  # dict_items([('id', 123), ('name', 'sylar'), ('age', 18), ('ok', '科比')]) 这个东西也是list. 只不过list中装的是tuple
for key, value in dic.items():  # ?? 这个是解构
    print(key, value)

# 解构
a, b = 1, 2
print(a, b)

(c, d) = 3, 4
print(c, d)

e, f = [1, 2, 3]    # 解构的时候注意数量必须匹配
print(e, f)
```

三. 字典的嵌套

```
# 字典的嵌套
dic1 = {
    "name": "汪峰",
    "age": 18,
    "wife": {
        "name": '章子怡',
        "age": 28
    },
    "children": ['第一个毛孩子', '第二个毛孩子'],
    "desc": '峰哥不会告我吧. 没关系. 我想上头条的'
}

print(dic1.get("wife").get("name"))
```

```
print(dic1.get("children"))
print(dic1.get("children")[1])
```

练习:

```
dic1 = {
    'name':['alex',2,3,5],
    'job':'teacher',
    'oldboy':{'alex':['python1','python2',100]}
    }
1，将name对应的列表追加一个元素’wusir’。
2，将name对应的列表中的alex首字母大写。
3，oldboy对应的字典加一个键值对’老男孩’,’linux’。
4，将oldboy对应的字典中的alex对应的列表中的python2删除。
```